Information Retrieval

Ling573
NLP Systems and Applications
April 26, 2011
Roadmap

- Problem:
  - Matching Topics and Documents

- Methods:
  - Classic: Vector Space Model

- Challenge: Beyond literal matching
  - Relevance Feedback
  - Expansion Strategies
Matching Topics and Documents

- Two main perspectives:
  - Pre-defined, fixed, finite topics:
    - “Text Classification”
Matching Topics and Documents

- Two main perspectives:
  - Pre-defined, fixed, finite topics:
    - “Text Classification”
  - Arbitrary topics, typically defined by statement of information need (aka query)
    - “Information Retrieval”
    - Ad-hoc retrieval
Information Retrieval Components

- Document collection:
  - Used to satisfy user requests, collection of:
Information Retrieval Components

- Document collection:
  - Used to satisfy user requests, collection of:
  - Documents:
    - Basic unit available for retrieval
Information Retrieval Components

- **Document collection:**
  - Used to satisfy user requests, collection of:
  - **Documents:**
    - Basic unit available for retrieval
      - Typically: Newspaper story, encyclopedia entry
Information Retrieval Components

- Document collection:
  - Used to satisfy user requests, collection of:
  - Documents:
    - Basic unit available for retrieval
      - Typically: Newspaper story, encyclopedia entry
      - Alternatively: paragraphs, sentences; web page, site
Information Retrieval Components

- **Document collection:**
  - Used to satisfy user requests, collection of:
  - **Documents:**
    - Basic unit available for retrieval
      - Typically: Newspaper story, encyclopedia entry
      - Alternatively: paragraphs, sentences; web page, site

- **Query:**
  - Specification of information need
Information Retrieval Components

- Document collection:
  - Used to satisfy user requests, collection of:
  - Documents:
    - Basic unit available for retrieval
    - Typically: Newspaper story, encyclopedia entry
    - Alternatively: paragraphs, sentences; web page, site

- Query:
  - Specification of information need

- Terms:
  - Minimal units for query/document
Information Retrieval Components

- Document collection:
  - Used to satisfy user requests, collection of:
  - Documents:
    - Basic unit available for retrieval
      - Typically: Newspaper story, encyclopedia entry
      - Alternatively: paragraphs, sentences; web page, site

- Query:
  - Specification of information need

- Terms:
  - Minimal units for query/document
    - Words, or phrases
Information Retrieval Architecture

1. Query
2. Query Processing
3. Indexing
4. Search (vector space or probabilistic)
5. Ranked Documents
Vector Space Model

- Basic representation:
  - Document and query semantics defined by their terms
Vector Space Model

- Basic representation:
  - Document and query semantics defined by their terms
  - Typically ignore any syntax
    - Bag-of-words (or Bag-of-terms)
      - Dog bites man == Man bites dog
Vector Space Model

• Basic representation:
  • Document and query semantics defined by their terms
  • Typically ignore any syntax
    • Bag-of-words (or Bag-of-terms)
      • Dog bites man == Man bites dog

• Represent documents and queries as
  • Vectors of term-based features
Vector Space Model

- Basic representation:
  - Document and query semantics defined by their terms
  - Typically ignore any syntax
    - Bag-of-words (or Bag-of-terms)
      - Dog bites man == Man bites dog

- Represent documents and queries as
  - Vectors of term-based features
  - E.g. $\vec{d}_j = (w_{1,j}, w_{2,j}, ..., w_{N,j})$; $\vec{q}_k = (w_{1,k}, w_{2,k}, ..., w_{N,k})$
  - $N$:
**Vector Space Model**

- Basic representation:
  - Document and query semantics defined by their terms
  - Typically ignore any syntax
    - Bag-of-words (or Bag-of-terms)
      - Dog bites man == Man bites dog

- Represent documents and queries as
  - Vectors of term-based features
  - E.g. \( \vec{d}_j = (w_{1,j}, w_{2,j}, \ldots, w_{N,j}) \); \( \vec{q}_k = (w_{1,k}, w_{2,k}, \ldots, w_{N,k}) \)
  - \( N \):
    - # of terms in vocabulary of collection: Problem?
Representation

• Solution 1:
  • Binary features:
    • $w = 1$ if term present, 0 otherwise

• Similarity:
  • Number of terms in common
  • Dot product
    \[
    \text{sim}(\vec{q}_k, \vec{d}_j) = \sum_{i=1}^{N} w_{i,k} w_{i,j}
    \]

• Issues?
VSM Weights

• What should the weights be?

• “Aboutness”
  • To what degree is this term what document is about?
  • Within document measure
  • Term frequency (tf): # occurrences of t in doc j

• Examples:
  • Terms: chicken, fried, oil, pepper
  • D1: fried chicken recipe: (8, 2, 7,4)
  • D2: poached chick recipe: (6, 0, 0, 0)
  • Q: fried chicken: (1, 1, 0, 0)
Vector Space Model (II)

- Documents & queries:
  - Document collection: term-by-document matrix

\[
A = \begin{pmatrix}
8 & 6 \\
2 & 0 \\
7 & 0 \\
4 & 0 \\
\end{pmatrix}
\]

- View as vector in multidimensional space
  - Nearby vectors are related

- Normalize for vector length
Vector Space Model

- Document k is further from query
- Document j (fried chicken recipe)
- Document k (poached chicken recipe)

Query ('fried chicken')

Dimension 1: 'fried'
Dimension 2: 'chicken'
Vector Similarity Computation

- **Normalization:**
  - Improve over dot product
    - Capture weights
    - Compensate for document length
Vector Similarity Computation

- Normalization:
  - Improve over dot product
  - Capture weights
  - Compensate for document length
- Cosine similarity

\[
sim(\tilde{q}_k, \tilde{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2} \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]
Vector Similarity Computation

- **Normalization:**
  - Improve over dot product
  - Capture weights
  - Compensate for document length

- **Cosine similarity**

\[
sim(\vec{q}_k, \vec{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2} \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]

- Identical vectors:
Vector Similarity Computation

- Normalization:
  - Improve over dot product
  - Capture weights
  - Compensate for document length
- Cosine similarity

\[
sim(\vec{q}_k, \vec{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2} \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]

- Identical vectors: 1
- No overlap:
Vector Similarity Computation

- Normalization:
  - Improve over dot product
  - Capture weights
  - Compensate for document length
- Cosine similarity

\[
\text{sim}(\vec{q}_k, \vec{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2} \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}
\]

- Identical vectors: 1
- No overlap:
Vector Similarity Computation

- **Normalization:**
  - Improve over dot product
  - Capture weights
  - Compensate for document length

- **Cosine similarity**

\[
\text{sim}(\tilde{q}_k, \tilde{d}_j) = \frac{\sum_{i=1}^{N} w_{i,k} w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,k}^2 \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}}
\]

- Identical vectors: 1
- No overlap: 0
Term Weighting Redux

- “Aboutness”
  - Term frequency (tf): # occurrences of t in doc j
Term Weighting Redux

- “Aboutness”
- Term frequency (tf): # occurrences of t in doc j
  - Chicken: 6; Fried: 1 vs Chicken: 1; Fried: 6
Term Weighting Redux

• “Aboutness”
  • Term frequency (tf): # occurrences of t in doc j
    • Chicken: 6; Fried: 1 vs Chicken: 1; Fried: 6

• Question: what about ‘Representative’ vs ‘Giffords’?
Term Weighting Redux

• "Aboutness"
  • Term frequency (tf): # occurrences of t in doc j
    • Chicken: 6; Fried: 1 vs Chicken: 1; Fried: 6

• Question: what about ‘Representative’ vs ‘Giffords’?

• "Specificity"
  • How surprised are you to see this term?
  • Collection frequency
  • Inverse document frequency (idf):

\[
idf_i = \log\left(\frac{N}{n_i}\right)
\]
Term Weighting Redux

• “Aboutness”
  • Term frequency (tf): # occurrences of t in doc j
    • Chicken: 6; Fried: 1 vs Chicken: 1; Fried: 6

• Question: what about ‘Representative’ vs ‘Giffords’?

• “Specificity”
  • How surprised are you to see this term?
  • Collection frequency
  • Inverse document frequency (idf):

\[
idf_i = \log\left(\frac{N}{n_i}\right) \quad w_{i,j} = tf_{i,j} \times idf_i
\]
Tf-idf Similarity

- Variants of tf-idf prevalent in most VSM

\[
\text{sim}(q,d) = \sum_{w \in q,d} tf_{w,q} \cdot tf_{w,d} \cdot (idf_w)^2
\]

\[
\sqrt{\sum_{q_i \in q} (tf_{q_i,q} \cdot idf_{q_i})^2} \cdot \sqrt{\sum_{d_i \in d} (tf_{d_i,d} \cdot idf_{d_i})^2}
\]
Term Selection

- Selection:
  - Some terms are truly useless
Term Selection

- Selection:
  - Some terms are truly useless
  - Too frequent:
    - Appear in most documents
Term Selection

- Selection:
  - Some terms are truly useless
    - Too frequent:
      - Appear in most documents
      - Little/no semantic content
Term Selection

- Selection:
  - Some terms are truly useless
    - Too frequent:
      - Appear in most documents
      - Little/no semantic content
    - Function words
      - E.g. the, a, and,...
Term Selection

- **Selection:**
  - Some terms are truly useless
    - Too frequent:
      - Appear in most documents
    - Little/no semantic content
    - Function words
      - E.g. the, a, and,...
  - Indexing inefficiency:
    - Store in inverted index:
      - For each term, identify documents where it appears
      - ‘the’: every document is a candidate match
Term Selection

- Selection:
  - Some terms are truly useless
    - Too frequent:
      - Appear in most documents
    - Little/no semantic content
    - Function words
      - E.g. the, a, and,...
  - Indexing inefficiency:
    - Store in inverted index:
      - For each term, identify documents where it appears
      - ‘the’: every document is a candidate match

- Remove ‘stop words’ based on list
  - Usually document-frequency based
Term Creation

- Too many surface forms for same concepts
Term Creation

• Too many surface forms for same concepts
  • E.g. inflections of words: verb conjugations, plural
    • Process, processing, processed
    • Same concept, separated by inflection
Term Creation

- Too many surface forms for same concepts
  - E.g. inflections of words: verb conjugations, plural
    - Process, processing, processed
    - Same concept, separated by inflection

- Stem terms:
  - Treat all forms as same underlying
    - E.g., ‘processing’ -> ‘process’; ‘Beijing’ -> ‘Beije’

- Issues:
Term Creation

• Too many surface forms for same concepts
  • E.g. inflections of words: verb conjugations, plural
    • Process, processing, processed
    • Same concept, separated by inflection

• Stem terms:
  • Treat all forms as same underlying
    • E.g., ‘processing’ -> ‘process’; ‘Beijing’ -> ‘Beije’

• Issues:
  • Can be too aggressive
    • AIDS, aids -> aid; stock, stocks, stockings -> stock
Evaluating IR

- Basic measures: Precision and Recall
Evaluating IR

- Basic measures: Precision and Recall
- Relevance judgments:
  - For a query, returned document is relevant or non-relevant
    - Typically binary relevance: 0/1
Evaluating IR

- Basic measures: Precision and Recall

- Relevance judgments:
  - For a query, returned document is relevant or non-relevant
    - Typically binary relevance: 0/1
  - T: returned documents; U: true relevant documents
  - R: returned relevant documents
  - N: returned non-relevant documents
Evaluating IR

- Basic measures: Precision and Recall

- Relevance judgments:
  - For a query, returned document is relevant or non-relevant
    - Typically binary relevance: 0/1
  - $T$: returned documents; $U$: true relevant documents
  - $R$: returned relevant documents
  - $N$: returned non-relevant documents

\[
\text{Precision} = \frac{|R|}{|T|}; \quad \text{Recall} = \frac{|R|}{|U|}
\]
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
    - In first 10 positions?
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
    - In first 10 positions?
    - In last 10 positions?
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
    - In first 10 positions?
    - In last 10 positions?
    - Score by precision and recall – which is better?
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
    - In first 10 positions?
    - In last 10 positions?
    - Score by precision and recall – which is better?
      - Identical !!!
      - Correspond to intuition? NO!
Evaluating IR

- Issue: Ranked retrieval
  - Return top 1K documents: ‘best’ first
  - 10 relevant documents returned:
    - In first 10 positions?
    - In last 10 positions?
    - Score by precision and recall – which is better?
      - Identical !!!
      - Correspond to intuition? NO!

- Need rank-sensitive measures
## Rank-specific P & R

<table>
<thead>
<tr>
<th>Rank</th>
<th>Judgment</th>
<th>Precision$^{\text{Rank}}$</th>
<th>Recall$^{\text{Rank}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>R</td>
<td>1.0</td>
<td>.11</td>
</tr>
<tr>
<td>2</td>
<td>N</td>
<td>.50</td>
<td>.11</td>
</tr>
<tr>
<td>3</td>
<td>R</td>
<td>.66</td>
<td>.22</td>
</tr>
<tr>
<td>4</td>
<td>N</td>
<td>.50</td>
<td>.22</td>
</tr>
<tr>
<td>5</td>
<td>R</td>
<td>.60</td>
<td>.33</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
<td>.66</td>
<td>.44</td>
</tr>
<tr>
<td>7</td>
<td>N</td>
<td>.57</td>
<td>.44</td>
</tr>
<tr>
<td>8</td>
<td>R</td>
<td>.63</td>
<td>.55</td>
</tr>
<tr>
<td>9</td>
<td>N</td>
<td>.55</td>
<td>.55</td>
</tr>
<tr>
<td>10</td>
<td>N</td>
<td>.50</td>
<td>.55</td>
</tr>
<tr>
<td>11</td>
<td>R</td>
<td>.55</td>
<td>.66</td>
</tr>
<tr>
<td>12</td>
<td>N</td>
<td>.50</td>
<td>.66</td>
</tr>
<tr>
<td>13</td>
<td>N</td>
<td>.46</td>
<td>.66</td>
</tr>
<tr>
<td>14</td>
<td>N</td>
<td>.43</td>
<td>.66</td>
</tr>
<tr>
<td>15</td>
<td>R</td>
<td>.47</td>
<td>.77</td>
</tr>
<tr>
<td>16</td>
<td>N</td>
<td>.44</td>
<td>.77</td>
</tr>
<tr>
<td>17</td>
<td>N</td>
<td>.44</td>
<td>.77</td>
</tr>
<tr>
<td>18</td>
<td>R</td>
<td>.44</td>
<td>.88</td>
</tr>
<tr>
<td>19</td>
<td>N</td>
<td>.42</td>
<td>.88</td>
</tr>
<tr>
<td>20</td>
<td>N</td>
<td>.40</td>
<td>.88</td>
</tr>
<tr>
<td>21</td>
<td>N</td>
<td>.38</td>
<td>.88</td>
</tr>
<tr>
<td>22</td>
<td>N</td>
<td>.36</td>
<td>.88</td>
</tr>
<tr>
<td>23</td>
<td>N</td>
<td>.35</td>
<td>.88</td>
</tr>
<tr>
<td>24</td>
<td>N</td>
<td>.33</td>
<td>.88</td>
</tr>
<tr>
<td>25</td>
<td>R</td>
<td>.36</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Rank-specific P & R

- \( \text{Precision}_{\text{rank}} \): based on fraction of reldocs at rank
- \( \text{Recall}_{\text{rank}} \): similarly
Rank-specific P & R

- Precision\textsubscript{rank}: based on fraction of reldocs at rank
- Recall\textsubscript{rank}: similarly
- Note: Recall is non-decreasing; Precision varies

\[
\text{Int\ Precision}(r) = \max_{i \geq r} \text{Precision}(i)
\]
Rank-specific P & R

- $\text{Precision}_{\text{rank}}$: based on fraction of reldocs at rank
- $\text{Recall}_{\text{rank}}$: similarly
- Note: Recall is non-decreasing; Precision varies
- Issue: too many numbers; no holistic view
Rank-specific P & R

- Precision_{rank}: based on fraction of reldocs at rank
- Recall_{rank}: similarly
- Note: Recall is non-decreasing; Precision varies
- Issue: too many numbers; no holistic view
  - Typically, compute precision at 11 fixed levels of recall
  - Interpolated precision:
    \[
    \text{Int\ Precision}(r) = \max_{i \geq r} \text{Precision}(i)
    \]
    - Can smooth variations in precision
# Interpolated Precision

<table>
<thead>
<tr>
<th>Interpolated Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>1.0</td>
<td>0.10</td>
</tr>
<tr>
<td>.66</td>
<td>0.20</td>
</tr>
<tr>
<td>.66</td>
<td>0.30</td>
</tr>
<tr>
<td>.66</td>
<td>0.40</td>
</tr>
<tr>
<td>.63</td>
<td>0.50</td>
</tr>
<tr>
<td>.55</td>
<td>0.60</td>
</tr>
<tr>
<td>.47</td>
<td>0.70</td>
</tr>
<tr>
<td>.44</td>
<td>0.80</td>
</tr>
<tr>
<td>.36</td>
<td>0.90</td>
</tr>
<tr>
<td>.36</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Comparing Systems

- Create graph of precision vs recall
  - Averaged over queries
  - Compare graphs
Mean Average Precision (MAP)

- Traverse ranked document list:
  - Compute precision each time relevant doc found
Mean Average Precision (MAP)

- Traverse ranked document list:
  - Compute precision each time relevant doc found
    - Average precision up to some fixed cutoff
    - $R_r$: set of relevant documents at or above $r$
    - $\text{Precision}(d)$: precision at rank when doc $d$ found
      \[
      \frac{1}{|R_r|} \sum_{d \in R_r} \text{Precision}_r(d)
      \]
Mean Average Precision (MAP)

- Traverse ranked document list:
  - Compute precision each time relevant doc found
    - Average precision up to some fixed cutoff
    - \(R_r\): set of relevant documents at or above \(r\)
    - \(\text{Precision}(d)\): precision at rank when doc \(d\) found
      \[
      \frac{1}{|R_r|} \sum_{d \in R_r} \text{Precision}_r(d)
      \]
  - Mean Average Precision: 0.6
    - Compute average over all queries of these averages
Mean Average Precision (MAP)

- Traverse ranked document list:
  - Compute precision each time relevant doc found
    - Average precision up to some fixed cutoff
    - $R_r$: set of relevant documents at or above $r$
    - Precision($d$) : precision at rank when doc $d$ found
      \[
      \frac{1}{|R_r|} \sum_{d \in R_r} \text{Precision}_r(d)
      \]
  - Mean Average Precision: 0.6
    - Compute average of all queries of these averages
    - Precision-oriented measure
Mean Average Precision (MAP)

- Traverse ranked document list:
  - Compute precision each time relevant doc found
    - Average precision up to some fixed cutoff
    - $R_r$: set of relevant documents at or above $r$
    - Precision($d$): precision at rank when doc $d$ found
  \[
  \frac{1}{|R_r|} \sum_{d \in R_r} \text{Precision}_r(d)
  \]
  - Mean Average Precision: 0.6
    - Compute average of all queries of these averages
    - Precision-oriented measure
- Single crisp measure: common TREC Ad-hoc