Information Retrieval and Related Applications

Adam Meyers
New York University
2018
Outline

• Information Retrieval and (Question Answering)
  – Introduction to Vector-based Similarity
  – Retrieving documents that are “similar” to query

• Document Classification and Sentiment Analysis
  – Using similarity to group “similar” documents

• Terminology Extraction
  – Finding the significant terms from a set of document
  – Text Mining based on units larger than the word

• Summary and Related Applications
Re-occurring Themes

• Document → Vector representing its parts
• What are its parts? Words, N-grams, Chunks?
• How do we measure each part's importance?
• Vector Similarity measures similarity between
  – 2 documents
  – A question and a document;
  – Documents within a genre
  – Documents with the same “sentiment”
• Evaluation Methodology
Ad Hoc Information Retrieval (IR)

• Given:
  – a collection of documents: a set of recipes
  – a query: “Chicken soup with noodles”

• Find documents that “best” match the query.

• Assumptions:
  – Each document is a “bag of terms”
  – Each query is a “bag of terms”

• Bag of terms = unordered set of words (“chicken”) or word sequences (“ice cream”, “frying pan”)
Ad Hoc IR – More Details

- Model of document = unordered set (bag) of terms contained in that document
  - Term = word, bigram (2 consec words), trigram (3 consec words, noun group (sequence of adjectives & nouns), other units
- Query = user input, typically a set of terms
- Collection = set of documents
- Goal: find documents in collection “closest” to query
- Web search has IR component
  - IR component determines relevance of page to query
  - Other component (Google's PageRank) determines prominence on web (how many links to page)
  - Other components: question answering, etc.
Vector Representations

- Represent query and documents as vectors
  - each value represents a “score” or “weight” for a word.

- **Query: “Chicken Soup with Noodles”**

```
<table>
<thead>
<tr>
<th></th>
<th>bean</th>
<th>chicken</th>
<th>lemon</th>
<th>noodle</th>
<th>stew</th>
<th>onion</th>
<th>soup</th>
<th>rice</th>
<th>coconut</th>
<th>sugar</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.4</td>
<td>0</td>
<td>2.7</td>
<td>0</td>
<td>0</td>
<td>5.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

- **Recipe for chicken rice stew**

```
<table>
<thead>
<tr>
<th></th>
<th>bean</th>
<th>chicken</th>
<th>lemon</th>
<th>noodle</th>
<th>stew</th>
<th>onion</th>
<th>soup</th>
<th>rice</th>
<th>coconut</th>
<th>sugar</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>16.2</td>
<td>0</td>
<td>0</td>
<td>3.1</td>
<td>2.7</td>
<td>0</td>
<td>10.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

- **Recipe for chicken noodle soup with coconut**

```
<table>
<thead>
<tr>
<th></th>
<th>bean</th>
<th>chicken</th>
<th>lemon</th>
<th>noodle</th>
<th>stew</th>
<th>onion</th>
<th>soup</th>
<th>rice</th>
<th>coconut</th>
<th>sugar</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10.8</td>
<td>0</td>
<td>27</td>
<td>1.1</td>
<td>2.7</td>
<td>10.2</td>
<td>0</td>
<td>14.1</td>
<td>5.1</td>
<td></td>
</tr>
</tbody>
</table>
```

- **Recipe for black bean soup with onion**

```
<table>
<thead>
<tr>
<th></th>
<th>bean</th>
<th>chicken</th>
<th>lemon</th>
<th>noodle</th>
<th>stew</th>
<th>onion</th>
<th>soup</th>
<th>rice</th>
<th>coconut</th>
<th>sugar</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>5.4</td>
<td>10.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```
Intuitions About Previous Slide

• Query is “close” to chicken noodle soup with coconut recipe
  – Mostly the same positions with zero values
  – Smaller difference between non-zero values
  – More “important” positions tend to be similar

• How should we determine the scores?
• How should we measure similarity?
TFIDF = Common Weight for Vector

- Term Frequency – number of times term $t$ occurs in document

- Inverse Document Frequency: Reciprocal of portion of large document set that contain term $t$, normalized with log function:
  \[
  \log \left( \frac{\text{NumberOfDocuments}}{\text{NumberOfDocumentsContaining}(t)} \right)
  \]

- TFIDF($t$) = TF($t$) $\times$ IDF($t$)
  - Scores terms highly that occur frequently in a document or query
  - Scores terms highly that are infrequent in collection

- TFIDF($t$) is high if $t$ is more frequent in document $d$ than $t$ is in most documents, i.e., if $t$ is characteristic of the document $d$
Example: \textit{noodle} vs. \textit{tablespoon}

- \textit{noodle}
  - occurs \(\sim 3\) times in chicken noodle soup with coconut recipe
    - Term frequency = 3
  - occurs in 4 out of 10,000 documents in collection
  - inverse document frequency = \(\log(10000/4) = \log(2500) = 7.82\)
  - TFIDF = \(3 \times 7.82 = 23.46\)

- \textit{tablespoon}
  - occurs 4 times in chicken and noodle soup with coconut recipe
    - Term frequency = 4
  - occurs in 1200 out of 10,000 documents in corpus
  - inverse document frequency = \(\log(10000/1200) = \log(8.33) = 2.12\)
  - TFIDF = \(4 \times 2.12 = 8.48\)

- \textit{noodle} is more highly weighted for recipes than \textit{tablespoon}

- Note: Suitability of query term may depend on the nature of the collection
  - Is this a collection of recipes? – \textit{tablespoon} not good query term
  - Is collection diverse: instructions, news, …? – \textit{tablespoon} may be good query term
Cosine Similarity: Common Similarity Score

\[ \text{Similarity}(A, B) = \frac{\sum_i a_i \times b_i}{\sqrt{\sum_i a_i^2 \times \sum_i b_i^2}} \]

- Cosine of the Angle Between the Vectors
- Numerator = Dot Product
- Denominator = Square root of product of squares
- If a query is A and a document is B
  - Cosine similarity high if values of a and b are similar
  - Maximum = 1, if a = b (i.e., numerator = denominator)
Cosine Similarity: Measures Closeness of Vectors

- Vectors are represented graphically as dimensions
  - 3 words → 3d space, 10 words → 10d space, ...
- 2 Vectors form an angle
- High Cosine if smaller angle (more similar)
Example

- Vector dimensions correspond to terms:
  - potato chip, chicken, sesame seed, coconut milk, ground beef

- 2 Queries
  - Q1 chicken, coconut milk: (0,5,0,5,0)
  - Q2 potato chip, ground beef: (4,0,0,0,7)

- 2 Documents
  - D1 Chicken and Coconut Soup Recipe: (0,7,0,9,0)
  - D2 Hamburger Recipe: (3,0,2,0,9)

- Cosine similarities

\[
\text{Similarity}(Q1, D1) = \frac{(0 + 35 + 0 + 45 + 0)}{\sqrt{5^2 + 0^2 + 5^2 + 0^2} \times (0^2 + 7^2 + 0^2 + 9^2 + 0^2)} = .992
\]

\[
\begin{array}{|c|c|c|}
\hline
 & Q1 & Q2 \\
\hline
D1 & .992 & 0 \\
\hline
D2 & 0 & .959 \\
\hline
\end{array}
\]
Other Factors

- Many terms (possibly thousands) represented in each vector
- Other similarity measures and weight functions
- Lists of “stop words”, e.g., the, a, in, to, does, …
- Stemming procedures used to make equivalence classes
  - \([cat, cats] \rightarrow cat\)
  - \([analyze, analyzes, analyzed, analysis, analyse,...] \rightarrow analyze\)
- Identifying other similar words, e.g., synonyms
  - query expansion, term clustering, …
- Systems identify word sequences as terms: N-grams or chunking
- Methods for removing dimensions, e.g., Latent Dirichlet Allocation
- Other methods for deriving vectors, e.g., Deep Learning
Evaluation Metrics: Precision, Recall, F-measure

- System Output = answers from a system
- Answer Key = correct answers from humans
- Correct = length (System Output ∩ Answer Key)
- Precision = Correct ÷ length/System Output
- Recall = Correct ÷ length/Answer Key
- F-measure = \[ \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} \]

Example:
- System: 1, 2, 4, 5, 7
- Answer Key 1, 2, 3, 4
- Correct = 1, 2, 4
- Precision = 3/5 = .6
- Recall = 3/4 = .75
- F-measure = 2 ÷ (1.33 + 1.67) = .67
Tasks can Favor Precision or Recall

• Voice Recognition of Helicopter Commands
  – Precision is much more important than Recall
  – An error is a disaster, better to produce no output and let pilot keep the control

• Finding Lost Children
  – Recall is much more important than Precision
  – Missing a child is a disaster, better to question some children who are not lost
Precision/Recall Curve: Parameters may trade precision for recall
Evaluation for Ranked IR retrieved Documents

- Precision/Recall tradeoff curves based on numbers in the ranking
  - Typically, precision goes down and recall goes up as more documents in the ranking are considered
  - Systems are sometimes measured by the area under the curve (the area is greater for better systems)

- Output of IR = A Ranked List of Documents
  - Ranking makes relevant/irrelevant distinction subtle
  - Error in high-ranked documents “worse” than error in low-ranked

- Too Expensive to Create Gold Standard Manually
  - Collections can be millions or billions of documents
  - Precision can be approximated by taking samples of the text or evaluating the top N ranked terms manually.
  - Recall can also be approximated by some sort of sampling, e.g., only manually evaluating a subset of the collection
Mean Average Precision is used to Score IR output

- Problem: How do we score a ranked list?
- MAP = compute precision at several intervals and average.
  - If high ranked items in list tend to be better, the score is higher
- Example
  - Answer Key: 20 correct answers
  - Different precision assuming progressively higher recall
    - 2/first 5 correct (2/5 precision at 10% recall)
    - 2/next 10 correct (4/15 precision at 20% recall)
    - 2/next 7 correct (6/22 precision at 30% recall)
    - ...
    - 20 correct/top 200 (20/200 precision at 100% recall)
  - MAP = average of these 10 intervals
    - \[0.195 \approx \frac{2/5 + 4/15 + 6/22 + 8/40 + 10/55 + 12/85 + 14/105 + 16/120 + 18/150 + 20/200}{10}\]
Homework 4

- http://cs.nyu.edu/courses/spring18/CSCI-UA.0480-009/homework4.html
- Information Retrieval Task
- Due February 28
Question Answering Task can be Similar to IR

- Given a query and set of documents
- Find the paragraph (not a document) that is closest to the query.
- This assumes the answer to the question will be found in the paragraph
- Essentially same problem as IR, but looking for most similar paragraph, instead of most similar document
(Supervised) Document Classification

• Given:
  – N sets (categories, genres, topics, etc) of seed documents
  – A set of unclassified documents

• Goal
  – Automatically assign new documents to “closest” category
Is Doc 11 a Recipe or a Sports Document?

<table>
<thead>
<tr>
<th>Doc</th>
<th>bean</th>
<th>chick</th>
<th>lime</th>
<th>salt</th>
<th>ball</th>
<th>stick</th>
<th>helmet</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4.3</td>
<td>5.1</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>7.8</td>
<td>10.2</td>
<td>2.4</td>
<td>0.6</td>
<td>2.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>5.1</td>
<td>4.5</td>
<td>1.5</td>
<td>0</td>
<td>4.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>4.3</td>
<td>5.1</td>
<td>2.4</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>9.6</td>
<td>0</td>
<td>4</td>
<td>0.6</td>
<td>0</td>
<td>4.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Ave</td>
<td>5.2</td>
<td>5.1</td>
<td>2.7</td>
<td>2.1</td>
<td>0.5</td>
<td>1.7</td>
<td>0</td>
<td>.2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Doc</th>
<th>bean</th>
<th>chick</th>
<th>lime</th>
<th>salt</th>
<th>ball</th>
<th>stick</th>
<th>helmet</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
<td>1.2</td>
<td>9.2</td>
<td>0</td>
<td>7.8</td>
<td>5.8</td>
</tr>
<tr>
<td>7</td>
<td>1.5</td>
<td>1.1</td>
<td>0</td>
<td>0</td>
<td>10.2</td>
<td>5.1</td>
<td>4.3</td>
<td>6.1</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5.4</td>
<td>4.3</td>
<td>5.7</td>
<td>8.1</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1.1</td>
<td>0</td>
<td>5.4</td>
<td>4.5</td>
<td>5.1</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7.2</td>
<td>9.0</td>
<td>4.5</td>
<td>0</td>
</tr>
<tr>
<td>Ave</td>
<td>0.3</td>
<td>0.44</td>
<td>0.2</td>
<td>0.26</td>
<td>6.4</td>
<td>4.8</td>
<td>5.4</td>
<td>6.8</td>
</tr>
</tbody>
</table>

**• Recipes**

**• Sports**

**• Doc 11**

(Unclassified)
Document Classification: Method

• Each document $\rightarrow$ vector of TF-IDFs of list of terms
• Each category $\rightarrow$ average of category doc vectors
  – Each dimension $d =$ average of scores for $d$
• Classification of New doc $\rightarrow$ most “similar” category
• Similarity scores may be ranked by best to worst fit
  – Allow each doc to be part of multiple categories?
    • maximum number of categories
    • a cut-off similarity score
    • etc.
Examples of Document Classification

• Given science papers divided by grade level
• System classified unclassified documents by grade level
• Wanjing Ma – Current CS /Education Major – undergrad
  – Education conference paper based on class project
• Other Examples:
  – Classify documents (tweets, news, etc.) by political persuasion (undergrad NLP students did this)
  – Classify newspaper articles by genre: sports, business, politics, etc.
  – Classify medical articles by subject matter: cancer, genetics, etc. (colleague did this)
Evaluating Document Clustering

• Set aside some test documents
  – already assigned classes by human beings
  – not seed documents

• Accuracy = correct ÷ number of documents
  – Precision and Recall are only used if the system can produce a different number of results

• Example: 50 correct out of 100 = 50% accuracy

• It is customary to set aside 2 sets of testing documents
  – dev set: use to design your system
  – test set: use to report final results
Unsupervised Document Clustering

- **Goal:** Partition set of unclassified documents
- **Given:**
  - Set of unclassified documents and their vectors
  - Each document is its own cluster
- **Repeat until some “stopping point”**
  - Merge the 2 most similar clusters
    - The vector for the new cluster is the average of the document vectors for the cluster
- **Stopping points:**
  - There are only N remaining clusters (e.g., N = 10)
  - The closest clusters have a similarity of less than X (e.g., X = .5)
Examples

• Automatically Group all final papers into 5 categories
  – Professor can grade similar papers together and have basis of comparison

• Divide 1000 news articles automatically into 10 topics
  – Based on sets of words associated with each topic, try to manually assign categories
More on Unsupervised Document Clustering

• Clusters are not guaranteed to be natural classes of documents (possible criterion for scoring results)

• Topic Modeling = A popular clustering technique
  – Methods to reduce the dimensions of the vectors
    • e.g., Latent Dirichlet allocation
  – Dimensions may not refer to particular words, but vectors represent topics (which are sets of words)
  – Topic is hierarchical and documents can be members of more than one cluster
  – Words representing documents are used to provide user with an idea of what each topic is “about”
Sentiment Analysis

• Document Classification
  – Classes based on opinions or sentiments

• Examples:
  – Positive vs Negative vs Neutral views about
    • Companies or their stock prices
    • Political Candidates
    • Products
  – Star Ratings or Scales from 1 to 5
    • Reviews of movies, products, etc.
  – Happy vs Sad vs Angry vs …. – Really difficult

• Sentiment Task Available: predict stars from reviews:
  – https://www.kaggle.com/c/sentiment-analysis-on-movie-reviews
  – Good choice for Final Project on Sentiment Analysis
Sentiment: Special Considerations

• Negation: not, no, never, doesn't, ….  
  – Bag of words model problematic  
  – “I don't like Senator Smith”  
    • Negative sentiment in spite of “like”

• Same terms linked to positive/negative in different contexts  
  – low, high, small, large, thin, thick, visible, loud, soft, …  
    • high/low quality, high/low interest, high/low resolution
Terminology Extraction

• Given
  – set of documents about a topic (foreground)
  – set of documents about diverse topics (background)

• Find ranked list of terms (words, n-grams, etc.)
  – that are more characteristic of foreground than background

• Uses:
  – terms for previously described tasks, search terms, terms for glossary, terms to track for technology forecasting (predicting technological emergence), etc.
Termolator  🕶

- Open Source Terminology Extraction for Chinese and English
  - [http://nlp.cs.nyu.edu/termolator/](http://nlp.cs.nyu.edu/termolator/)

- Created under a government contract as part of the Foresight and Understanding from Scientific Exposition (FUSE)

- Collaborators at NYU: Zachary Glass, Ralph Grishman, Yifan He, Giancarlo Lee, Shasha Liao, Angus Grieve-Smith and John Ortega
What is Terminology?

• Webster’s II New Collegiate Dictionary Definition
  – *The vocabulary of technical terms and usages appropriate to a particular field, subject, science, or art.*

• Operational Definitions:
  – Keyword sequences for Information Retrieval (IR)
    • Need not be technical, e.g., *wheat, barley, white mouse*, in genetics
  – Items to define in Technical Glossaries
  – Items to track for Technology Forecasting (TF)

• Noun Terminology:
  – Technical word sequence headed by noun
  – Vast majority of all terminology
  – Non-noun terminology exists, but not included in this research
Examples of Terminology

- Juggling: *cascade pattern, Mills mess, full shower*
- Real Estate: *balloon mortgage, title search, full shower*
- Computer Science: *hidden markov model, genetic algorithm, top-down search*
- Knitting: *gobelin stitch, half-treble, corner scallop*
- Biology: *myosin-ii, plasminogen activator, antizyme*
Termolator Work Flow

Foreground Text
A semiconductor device which includes a semi...

Background Text
A cascade pattern differs from a shower pattern ...

1 In-Line Term Extractor

Foreground Terms
semiconductor device semiconductor chip bin-sort algorithm ...

2 Distributional Comparison Using Statistical Metrics

Background Terms
Newton's 2nd Law cascade pattern correlation coefficient...

3 WF and Relevance Filters

Ranked Output
transfer electrode semiconductor face micropattern

Final ReRanked Output: transfer electrode; semiconductor face; micropattern, through-connection, wavelength conversion chip, ...
The Termolator: 2 Main Subsystems

• **In-Line Term System**: Finds instances of terms (tokens)
  • Finds noun/adjective sequences that obey constraints
  • Identifies term tokens, instances of terms in sentences
    • 500 term tokens occur in document X
    • 50 are instances of *biotrophic effector models*
  • Limited previous work in this area

• **Distributional Term System**: Finds term types
  • Counts instances of term types
    • 30 term types occur in document X
    • *biotrophic effector models* occurred 500 times
  • Ranks term types by characteristic-ness to a particular topic
  • Top N term types are kept, the rest are discarded
  • Uses metrics similar to TF-IDF discussed in previous slides
Our In-line Term Extraction System

- Manual Rule Based “Chunker”
  - Identifies sequences of nouns and adjectives
    - part of speech tagger output
    - dictionaries
  - Technical words identified:
    - Out-of-Vocabulary (OOV) words – words not in dictionaries
      - semiconductor, biotrophic, gobelin
    - Technical Adjectives – based on endings (-ic,-cal,-ous ) and dictionaries
      - algebraic, amphibious, umbilical
    - Nominalizations – based on endings (tion, etc.) and NOMLEX dictionary
      - conduction, vulcanization, accelerator

- Well-formedness filter
  - eliminates ill-formed (too short, bad characters, etc.)
  - eliminates terms without OOV or technical words
  - eliminates words detected to be names of people or places
Example Identification of Technical Noun Adjective Sequences

- A semiconductor device which includes: a semiconductor chip bonded to a surface of a solid device; and a stiffener surrounding the periphery of the semiconductor chip.

Rules group yellow words (below) together resulting in blue sequences (above).

IR and Related Applications 2018
Filters Remove Unlikely Candidate Terms

- Accepts Terms which contain an Out-of-vocabulary (OOV) word
  - semiconductor/O-NOUN device
  - semiconductor/O-NOUN chip (2 instances)
- Accepts Terms containing technical adjectives or nominalizations
  - thermal/TECH-ADJ stress
  - fabrication/NOM process
- Rejects Terms because they contain no technical words
  - surface
  - device
  - stiffener
  - periphery
- Other Non-Terms removed for other reasons
  - T
  - 212-345-8888
  - No.
  - New York
Supplementary patterns for identifying Terms

- Arguments of Abbreviation relations
  - Not organizations or places
  - Aligns words before parentheses with word in parentheses
    - *already been chewed (ABC)*
    - XML (Extensible Markup Language)
    - *third variable loop (V3)*
    - *D. melanogaster gene Muscle LIM protein at 84B (abbreviated as Mlp84B)*
  - Schwartz and Hearst (2003)

- Terms Matching Regexp Patterns
  - Gene Sequences: *AACAAGGTGGCGCAGTT*
  - Chemical Formulas: *Ag2CrO4*
Evaluation of Inline Term System

- 2 Annotators Manually Annotated Inline terms in 3 documents
- Adjudicated the Results
- Scored annotators against adjudicated annotation
- Scored system against adjudicated annotation
- Compared annotator vs system performance
Annotation

• Setup
  • 2 annotators annotated the same three documents
  • Annotator 2 Adjudicated
  • Annotator 1’s score against Adjudicated results may be a good Upper Bound for evaluating the Automatic System (assumes the adjudication is biased in favor of Annotator 2).

• Defining Inline Term for Annotator
  • Single or multi-word nominal expression specific to technical discipline
  • It can be conventionalized by defining or abbreviating it early in the document and by reusing the term
  • Is term specific to technical discipline, i.e., is it obscure enough?
    • Would a naïve adult (like Homer Simpson) know the term?
    • Is it found in the Juvenile subcorpus of the Corpus of Contemporary American English (http://corpus.byu.edu/coca/)?
Corpora and Systems Tested

• Corpora
  • A Speech Recognition Patent (SRC)
  • A Sun Screen Patent (SUP)
  • A Journal Article about a Virus Vaccine (VVA)

• Systems Tested
  • Base 1: assume all noun groups minus determiners are terms
    • use MEMM chunker with Genia (Kim et al 2003) features
  • Base 2: baseline 1 system, but filtered by only keeping those Noun Groups that end with an O-NOUN
  • System without Filter: Chunking system as described, but without the filter
  • Final System

• Matching Criteria
  • Strict Match – The test term and answer key term are the same
  • Sloppy Match – The test term and answer key term overlap in extent.
## Inter Annotator Agreement

<table>
<thead>
<tr>
<th>Annotator 1</th>
<th>Doc</th>
<th>Terms</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRP</td>
<td>1131</td>
<td>798</td>
<td></td>
<td>70.8%</td>
<td>70.6%</td>
<td>70.7%</td>
<td>1041</td>
<td>92.5%</td>
<td>92.0%</td>
<td>92.2%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>1809</td>
<td></td>
<td>87.5%</td>
<td>83.5%</td>
<td>85.5%</td>
<td>1992</td>
<td>96.3%</td>
<td>92.0%</td>
<td>94.1%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>713</td>
<td></td>
<td>90.9%</td>
<td>77.6%</td>
<td>83.7%</td>
<td>762</td>
<td>97.2%</td>
<td>82.9%</td>
<td>89.5%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Annotator 2</th>
<th>Doc</th>
<th>Terms</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRP</td>
<td>1131</td>
<td>960</td>
<td></td>
<td>98.4%</td>
<td>84.9%</td>
<td>91.1%</td>
<td>968</td>
<td>99.2%</td>
<td>85.6%</td>
<td>91.9%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>1999</td>
<td></td>
<td>95.5%</td>
<td>92.3%</td>
<td>93.8%</td>
<td>2062</td>
<td>98.5%</td>
<td>95.2%</td>
<td>96.8%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>838</td>
<td></td>
<td>97.4%</td>
<td>91.2%</td>
<td>94.2%</td>
<td>855</td>
<td>99.4%</td>
<td>93.0%</td>
<td>96.1%</td>
</tr>
</tbody>
</table>

Annotator 1 scores may be upper bounds for system results
## Baseline Systems

<table>
<thead>
<tr>
<th></th>
<th>Doc</th>
<th>Terms</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base 1</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRP</td>
<td>1131</td>
<td>602</td>
<td></td>
<td>24.3%</td>
<td>53.2%</td>
<td>33.4%</td>
<td>968</td>
<td>44.2%</td>
<td>96.8%</td>
<td>60.7%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>1367</td>
<td></td>
<td>36.5%</td>
<td>63.1%</td>
<td>46.2%</td>
<td>1897</td>
<td>50.6%</td>
<td>87.6%</td>
<td>64.2%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>576</td>
<td></td>
<td>28.5%</td>
<td>62.7%</td>
<td>39.2%</td>
<td>887</td>
<td>44.0%</td>
<td>96.5%</td>
<td>60.4%</td>
</tr>
<tr>
<td><strong>Base 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRP</td>
<td>1131</td>
<td>66</td>
<td></td>
<td>24.9%</td>
<td>5.8%</td>
<td>9.5%</td>
<td>151</td>
<td>57.0%</td>
<td>13.4%</td>
<td>21.6%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>771</td>
<td></td>
<td>52.3%</td>
<td>35.6%</td>
<td>42.4%</td>
<td>1007</td>
<td>68.4%</td>
<td>46.5%</td>
<td>55.3%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>270</td>
<td></td>
<td>45.8%</td>
<td>29.4%</td>
<td>35.8%</td>
<td>392</td>
<td>66.5%</td>
<td>42.6%</td>
<td>51.9%</td>
</tr>
</tbody>
</table>

- Base 1 (all noun groups): results in high recall/low precision
- Base 2 (must end in O-NOUN): too severe a filter.
## System Results

<table>
<thead>
<tr>
<th></th>
<th>Doc</th>
<th>Terms</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
<th>Matches</th>
<th>Pre</th>
<th>Rec</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Strict</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No Filter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRP</td>
<td>1131</td>
<td>932</td>
<td></td>
<td>39.0%</td>
<td>82.4%</td>
<td>53.0%</td>
<td>1121</td>
<td>46.9%</td>
<td>99.1%</td>
<td>63.7%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>1475</td>
<td></td>
<td>39.7%</td>
<td>68.1%</td>
<td>50.2%</td>
<td>1962</td>
<td>52.8%</td>
<td>90.6%</td>
<td>66.7%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>629</td>
<td></td>
<td>27.8%</td>
<td>68.4%</td>
<td>39.5%</td>
<td>900</td>
<td>39.8%</td>
<td>97.9%</td>
<td>56.6%</td>
</tr>
<tr>
<td><strong>Final System</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRP</td>
<td>1131</td>
<td>669</td>
<td></td>
<td>69.0%</td>
<td>59.2%</td>
<td>63.7%</td>
<td>802</td>
<td>82.8%</td>
<td>70.9%</td>
<td>76.4%</td>
</tr>
<tr>
<td>SUP</td>
<td>2166</td>
<td>1193</td>
<td></td>
<td>64.7%</td>
<td>55.1%</td>
<td>59.5%</td>
<td>1526</td>
<td>82.8%</td>
<td>70.5%</td>
<td>76.1%</td>
</tr>
<tr>
<td>VVA</td>
<td>919</td>
<td>581</td>
<td></td>
<td>62.1%</td>
<td>63.2%</td>
<td>62.7%</td>
<td>722</td>
<td>77.2%</td>
<td>78.6%</td>
<td>77.9%</td>
</tr>
</tbody>
</table>

Final System gets the highest F-score
Distributional Term System

• Find In-line Terms for Foreground Corpus (or sample)
• Find In-line Terms for Background Corpus (or sample)
• Count instances of the same lemma as instances of the same term
  • singular/plural, -ing endings (stemming)
  • \textit{speech recognizers} → \textit{speech recognizer}
• abbreviation/full-form
  • \textit{html} → \textit{hypertext markup language}
• Noun mod alternations:
  • \textit{Recognition of Speech} → \textit{Speech Recognition}
• Rank by Statistical Metrics similar to TF-IDF
  • finds terms more characteristic of foreground than background
• Rerank terms using Relevance Metric, based on a Yahoo Websearch
• Take Top N terms (e.g., N = 5000)
Statistical Metrics for Ranking Terms

- A linear combination of 3 Measures comparing the distribution of terms in the foreground (For) vs background (Bac)
- Term Frequency Inverse Document Frequency (TFIDF)
  \[ TFIDF(t) = \frac{Freq_{For}(t)}{Freq_{Bac}(t)} \times \log \left( \frac{NumBacDocuments}{NumBacDocsContain(t)} \right) \]
- Document Relevance Document Consensus (DRDC)
  \[ DRDC(t) = \frac{Freq_{For}(t)}{Freq_{Bac}(t)} \times \sum_{d \in \text{Foreground}} \frac{freq_{Bac}(t,d)}{freq_{For}(t)} \times \log \left( \frac{freq_{For}(t)}{freq_{Bac}(t,d)} \right) \]
  - Doc Relevance (1\textsuperscript{st} factor) favors representative terms (like TFIDF)
  - Doc Censensus (2\textsuperscript{nd} factor) favors terms found in many documents
- Kullback Leibler Divergence (KLD)
  \[ KLD(t) = \log(freq_{For}(t)) - \log(freq_{Bac}(t)) \times freq_{For}(t) \]
  - Compares Probability of term occurs in Foreground vs. Background
Filters on Distributional Output

- 2 Filters that can be applied to our system or output of other term generation systems
  - In FUSE, they were applied to MITRE and BBN output
- Both scores are between 0 and 1, they are combined by multiplication
- Well-Formedness Filter
  - Many of the constraints are built into our chunker
    - Most terms have a score of 1
  - However, component of distributional System adds some common substrings of terms to output, some of which are ill-formed
- Relevance Filter
  - We use a Yahoo search result and heuristics to score terms more highly if they are used in articles or patents
Well-Formedness Filter

• A term is well-formed if it is:
  • An abbreviation
  • A set of words that is abbreviated somewhere in the corpus
  • A single out of vocabulary word
  • Matches a regular expression that finds chemical names, DNA sequences or paths (urls, bio paths, etc.) – although URLs can be documents, rather than terms.

• A term is also well formed if it obeys noun group rules (a sequence of adjectives and nouns ending in a noun) AND it contains at least one out-of-vocabulary word, nominalization or technical adjective

• The degree of ill-formededness is not so important as scores below 1 rarely apply to accepted terms. (Sometimes favors terms with OOV words over terms with other technical words and no OOVs)

• This filter is more important when applied to term lists not created by The Termolator (Mitre and BBN term lists in FUSE)
Relevance Filter

- Run on each term below some cutoff (typically 30K)
  - Time consuming (about .75 seconds per term)
- Yahoo search (Bing) for exact match of term
- Relevance = $H^2T$
  - $H = 0$ to 1 score based on number of hits
    - $\min\left(\log_{10}(\text{numberHits}), 10\right)$
    - Minimized for non-hits (0 hits counts as 500 hits)
  - $T = \text{Percent of top 10 hits that are articles or patents}$
    - Based on key word search in title, url & summary
      - Key words = \{patent, article, proceedings, journal, dissertation, abstract, ...\}
Evaluation of Termolator Output

- Foreground Corpus: 2500 patents about optical systems
- Background Corpus: 2500 randomly selected patents
- Years: 1997-2007
- Took the top 30K out of 219K terms and reranked using:
  - Percentile X Well-Formedness X Relevance
- Manually evaluated 100 terms sampled from top 5000 terms
- A term was judged correct if
  - valid keyword
  - not missing crucial modifier
  - did not contain any spurious word.
- The system achieved 86% Precision
- Recall difficult to measure, but also produces more high-quality terms
- Competitive with other systems (main innovation is: inline terms)
Evaluation Details

• Sample Correct (sampled from the first 5000):
  – *stimulable phosphor, ion beam profile, x-ray receiver, wavelength-variable, quadrupole lens, proximity correction, dfb laser, asymmetric stress, panoramagram, single-mode optical fiber, total reflection plane, photosensitive epoxy resin*

• Sample Incorrect
  – *irradiation time t*
    • A variable, not a term (without t, it would be a term)
  – *evolution*
    • This word has entered the common vocabulary
  – *crystal adjacent*
    • This word sequence includes two words at a constituent boundary
      – a noun phrase followed by a modifying adjective phrase, e.g.,
        – *[a liquid crystal] [adjacent to the lower alignment layer]*
Informal Observations about Recall

• Recall or coverage is difficult to measure without an exhaustive amount of human annotation.
• The distributional system gets roughly the same precision for Noun Group input as Inline Term Group Input for the top $N$ terms, where $N$ is a small number.
• Using Inline Terms as input, we generate many more terms with high scores and thus seem to improve Recall by a large amount (at least a factor of 2).
  • But this is hard to measure.
• **Rationale**: Garbage In $\rightarrow$ Garbage Out
  • High F-scores for inline terms (vs NGs or N-grams)
  • Higher Quality terms are being ranked and so the high-ranked items are more likely to be correct.
The Termolator for Chinese 🕶️

- Work by Yifan He
- Distributional System is the Same as English
- Uses Noun Group Chunker for input terms
- Accessor-Variety Filter (Feng et al., 2004)
  - Score Based on the Number of distinct words that appear before and after a particular term type
  - Low Scores indicate unlikely Chinese words
- 1100 terms extracted from 2000 speech recognition patents
  - 78% precision on top 50 terms
  - 85% precision on top 20 terms
Example of Chinese Term Filtering

• Examples for Access Variety based filtering
  • 尔科夫模型 (Markov model, with the first Chinese character 马 missing) is probably a boundary error
  • [Pic on left] 尔科夫模型 has the same character 马 on its left boundary thus its Left AV=1
  • [Pic on left] A correct term 声学 (acoustics) will have Left AV>=3
Open Source Distribution

- Open Source release of The Termolator
  - NYU’s Website: http://nlp.cs.nyu.edu/termolator/
  - Github:
    - English: https://github.com/AdamMeyers/The_Termolator
    - Chinese: https://github.com/ivanhe/termolator/
- English system for UTF-8 (including ASCII) & ISO-8859-1
- Tested on Public Domain Texts
  - Google Patents
  - Project Gutenberg
  - Open American National Corpus
Examples from Public Domain Texts

- Gutenberg: Chapters in a Book about knitting vs Other Docs
  - *open-work insertion, fine mesh, transverse stitching, empty scallop*

- Open American National Corpus (OANC) – Biology documents versus random documents
  - *myosin-ii, hsn3, intron, migration defect, sparc-null mice*

- Google Patents: Surgery patents (US Patent Class 606) vs Random Patents:
  - *fluid manifold, dissector arm, pedicle punch, balloon catheter*
Possible Final Project Topics
Connected to Termolator

• Apply Termolator to another Language
  – Example: Spanish Termolator
• Further Develop Chinese Termolator
• Error Analysis and Improvements to Applying Termolator to Legal Domain
Summary of Techniques

• Text Mining Applications use Vectors with weights represent queries, paragraphs, documents, or sets of similar documents
  – TF-IDF and similar measures weigh terms more highly if they occur more frequently in selected context

• Cosine can measure similarity among bags of terms
  – 2 documents, a query and a document, etc.

• TF-IDF measures can also be used to find “special” terms that characterize particular sets of documents

• Terms can be words, N-grams or other word sequences
  – Choice for Termolator: technical noun/adjective sequences
Summary: Text Mining Applications

• Discussed Explicitly
  – Information Retrieval (and Question Answering)
  – Document Classification (and Sentiment Analysis)
  – Terminology Extraction

• Not Discussed, but Related
  – Other types of document classification: authorship, time period, style, modality, ...
    • Other features: sentence length, punctuation, ...
    • Different stop words (closed class words may be important)
  – Document Summarization
    • Find “important” sentences (not just important words)
    • Choose sentences to include in summary
      – Exclude a sentence if too similar to sentence already included in summary
  – Word Embedding approaches to word meaning
    • Define word meaning by vectors of words they co-occur with
      – Uses: identify senses of words; identify words that share similar meanings
    • Can also be used in applications discussed here
About Final Project Topics

• There are many final project topics that would use the techniques discussed in this set of slides.
• Using existing data (including evaluation set) allows you to focus on techniques that you use.
• Creating your own data set is potentially a task distinct from (and as time consuming as) making a system.
Extra Slides
Evaluation of Distributional Term System

- Foreground Corpus: 2500 patents about optical systems
- Background Corpus: 2500 randomly selected patents
- Years: 1997-2007
- Ran the Distributional System and Ordered the Terms
  - Confidence$_1$ = Percentile X Well-Formedness
    - Uses the Percentile Ranking based on the distributional score, but filters out ill-formed terms
  - Took the top 30K out of 219K terms and reranked using:
    - Relevance only; and
    - Confidence$_2$ = Percentile X Well-Formedness X Relevance
      - Uses Relevance on 30K terms due to time constraints
Evaluation Distribution System Slide 2

• Took Top 5000 terms ranked each of 3 ways and Scored for Precision
  • Confidence_1 Precision = 71%
  • Relevance Precision = 82%
  • Confidence_2 Precision = 86%

• For each ranked set, we took samples of 100 terms:
  • 20 from first 20%, 20 from second 20%, ... 20 from 5th 20%.

• We manually evaluated the samples:
  • Terms were deemed correct if the term was deemed a valid keyword, was not missing any crucial modifier, and did not contain any spurious word.