Lecture 9: Text Clustering & Document Expansion

Jiepu Jiang, Assistant Professor
Department of Computer Science, Virginia Tech
2018/9/18
HW2 issues

• Many apologies!
• Deadline extended to Thursday (11:59pm, EST)
• Download the new “nostem” and “porter2” indexes
• Rerun experiments & Update answers in Problem 1.2.3
• Only affects Problem 1.2.3
Final Project (30%)

Purpose
• Demonstrate your ability to conduct research related to information retrieval
• Most IR research are experimental (similar to what we did in HWs in style)

You can work either alone or in group of two
• The expected work load will be different
• Work alone: about 2-3 times as much as you did in a HW
• Work in group: about 4-5 times as much as you did in a HW
• Grading: presentation (5%) & final report (25%)
• Group members get the same grade (you should make your teammate work)

Milestones
• 10/4 – group formulation (or let me know if you decide to work alone)
• 10/23 – proposal due (one-page, ACM proceeding format)
• 11/29 & 12/4 – project presentation
• 12/9 – final report due (no more than four pages; ACM proceeding format)
Project: Proposing your own project

What you need to do

• Identify a task/problem or propose a new task/problem
• Should be related to information retrieval; but does not need to be IR ranking
• Design some reasonable solution
• Think about evaluation strategies (e.g., datasets, baselines, measures)
• Run experiments & analysis
• Draw conclusions & write your report
• Possible extra credit (up to 5% of your total course grade)
• (optional) Refine & submit to SIGIR 2019 (as a short paper, etc.)
  • The deadline is usually in Feb 2019
  • Short paper usually requires 4 pages (ACM proceeding format)
  • SIGIR is very generous in terms of supporting student travel
Project: Reproducing an existing study

Purpose

- Background: results in many published studies cannot be reproduced
- Re-examine a published paper; verify its claims & findings through experiments
- Is it IR research?
  - Yes, implementing baselines is a part of research effort (but boring)
  - Only a small fraction of authors will make their code public (and not necessarily usable)
- No extra credit

What you need to do

- Find a paper published later than 2008 in:
  - IR conferences: SIGIR, CIKM, WSDM, WWW (IR track), ECIR, ICTIR, etc.
  - NLP conferences: ACL, EMNLP, NAACL, COLING, etc.
- Implement the technique(s) described in the paper
- Run experiments in a few different datasets
- Compare the results & findings with those reported in the original paper
Helpful suggestions

Be practical

• Do something you can handle ...
• Datasets ...
  • Always check dataset availability ...
  • Probably should avoid creating datasets by yourself ...
  • We have many popular datasets ...
• Start early
  • Running experiments takes time ...
  • We will grade based on your report!
• Check with the instructor if you are not sure, e.g.,
  • Is this task/problem related to IR?
  • Is my idea reasonable?
  • Can I have access to xxx dataset? What datasets should I use?
  • How much work should I do? Is it enough?
Recap: Smoothing

• Motivated by the zero probability issue of MLE ...
  • “Interpolate” (mix) an MLE model with a background model
  • The background model is usually the corpus model

• Jelinek-Mercer Smoothing
  • Mixing MLE and the background model by a constant weight
    \[ P(t \mid \theta_D) = (1 - \lambda) \cdot P_{MLE}(t \mid \theta_D) + \lambda \cdot P(t \mid Corpus) \]

• Dirichlet Smoothing
  • The background model has a smaller weight in longer documents
    \[ P(t \mid \theta_D) = \frac{c_{t,D} + \mu \cdot P(t \mid Corpus)}{|D| + \mu} \]

MLE weight: \( 1 - \lambda = \frac{|D|}{|D| + \mu} \)  Corpus model weight: \( \lambda = \frac{\mu}{|D| + \mu} \)
Recap: Smoothing and IDF

• Similar to many retrieval models, we can write QL as:

\[
\text{score}_{QL}(q, D) = \sum_{t \in q} w_{QL}(t, D), \text{ where } w_{QL}(t, D) = \log P(t \mid D)
\]

• How fast \( w_{QL} \) increases while term frequency increases?

\[
\frac{\partial w_{QL}(t, D)}{\partial c_{t, D}} = \frac{1}{P(t \mid D)} \cdot \frac{\partial P(t \mid D)}{\partial c_{t, D}}
\]

MLE: \( P(t, D) = \frac{c_{t, D}}{|D|} \), \[ \frac{\partial w_{QL}(t, D)}{\partial c_{t, D}} = \frac{1}{P(t \mid D)} \cdot \frac{1}{|D|} = \frac{1}{c_{t, D}} \]

• \( \frac{\partial w_{QL}(t, D)}{\partial c_{t, D}} \) only depends on term frequency \( c_{t, d} \)

• \( \frac{\partial w_{QL}(t, D)}{\partial c_{t, D}} \) does not relate to corpus-level term statistics

\( c_{t, d} \) is discrete, but let’s just assume the functions are all continuous here.
Recap: Smoothing and IDF

• How fast $w_{QL}$ increases while term frequency increases?

$$w_{QL}(t, D) = \log P(t \mid D)$$

JM smoothing: $P(t, D) = (1 - \lambda) \cdot \frac{c_{t,D}}{|D|} + \lambda \cdot P(t \mid Corpus)$

$$\frac{\partial w_{QL}(t, D)}{\partial c_{t,D}} = \frac{1}{P(t \mid D)} \cdot \frac{1 - \lambda}{|D|} = \frac{1 - \lambda}{(1 - \lambda) \cdot c_{t,D} + \lambda \cdot |D| \cdot P(t \mid Corpus)}$$

• $\frac{\partial w_{QL}(t, D)}{\partial c_{t,D}}$ depends on not only within-document term frequency ($c_{t,D}$), but also corpus-level term statistics
• $w_{QL}$ increases much slower for common terms, which have a higher $P(t \mid Corpus)$
• JM smoothing has an IDF effect
Recap: Smoothing and IDF

• How fast $w_{QL}$ increases while term frequency increases?

$$w_{QL}(t, D) = \log P(t \mid D)$$

Dirichlet smoothing: $$P(t, D) = \frac{c_{t,D} + \mu \cdot P(t \mid Corpus)}{|D| + \mu}$$

$$\frac{\partial w_{QL}(t, D)}{\partial c_{t,D}} = \frac{1}{P(t \mid D)} \cdot \frac{1}{|D| + \mu} = \frac{1}{c_{t,D} + \mu \cdot P(t \mid Corpus)}$$

• $\frac{\partial w_{QL}(t, D)}{\partial c_{t,D}}$ depends on not only within-document term frequency ($c_{t,D}$), but also corpus-level term statistics
• $w_{QL}$ increases much slower for common terms, which have a higher $P(t \mid Corpus)$
• Dirichlet smoothing also has an IDF effect
Recap: the KL-divergence Model

- Similarity implies relevance
  - Estimate a document language model
  - Estimate a query language model
  - Compare the similarity of the two models
- If we estimate the query model using MLE, KLD is equivalent to QL
  - Refined query model (next lecture)

\[-\text{KLD} \left( \theta_q \parallel \theta_D \right) = \sum_t P(t \mid \theta_q) \log P(t \mid \theta_D) - \sum_t P(t \mid \theta_q) \log \text{QL} \]

\[\sum_t P_{\text{MLE}}(t \mid \theta_q) \log P(t \mid \theta_D)\]

\[= \sum_t \frac{c(t,q)}{|q|} \log P(t \mid \theta_D) = \frac{\sum_t c(t,q) \log P(t \mid \theta_D)}{|q|} = \frac{1}{|q|} \cdot \text{QL}\]
Recap: Term Proximity

• Words are not independent of each other
  • e.g., phrases, two words co-occur in a window of size 8
• Usually improves 5%-15% in ad hoc search
  • No surprisingly better results than unigram
  • Performance vary by dataset
  • In fact, unigram models (bag-of-words) are quite strong in IR
  • But higher-order models are much stronger than unigram in NLP, probably because of task difference and the size of training data
    • We are estimating document models in IR.
    • Query generation assumption ...
  • The assumption does not always hold
    • Not all query terms have strong dependency ...
• A higher cost to use
  • Requires proximity search
Recap: The SDM Model

- The most popular/effective one is the sequential dependence model (SDM), e.g., consider the dependency between each two consecutive query terms.

- Dependency of two query terms is not restricted to phrase

- Ordered phrase (bigram)
  - $t_1$ followed by $t_2$; two words adjacent to each other
  - Indri/Galago query: #1( $t_1$ $t_2$ )

- Unordered phrase within a window of size $k$ (skip-gram)
  - $t_1$ and $t_2$ co-occur in the same size $k$ window
  - Do not need to be consecutive; ignore sequence
  - e.g., $k = 8$; Indri/Galago query: #uw8( $t_1$ $t_2$ )
Recap: The SDM Model

- The original full dependence model is theoretically flexible to include almost any types of query term dependencies

- A few practical simplification:
  - Only sequential dependence of two consecutive query terms
  - Only consider unordered phrase within a particular window size $n$
    - $n = 8$ is the most common option (a sentence)
    - $n = 50$ (a paragraph)

- Combine different features using a linear model
  - Equivalent to QL if we set $\lambda_{ow} = 0$ and $\lambda_{uw} = 0$
  - Many use the weight 0.8, 0.1, 0.1

$$
\text{score}(q,d) = (1 - \lambda_{ow} - \lambda_{uw}) \log P(f_{\text{unigram}} | d) \\
+ \lambda_{ow} \log P(f_{ow} | d) + \lambda_{uw} \log P(f_{uw} | d)
$$
Recap: The SDM Model

- SDM score
  - Weighted linear combination of different features’ log probability

\[
score(q, d) = \left(1 - \lambda_{ow} - \lambda_{uw}\right) \log P(f_{unigram} | d) + \lambda_{ow} \log P(f_{ow} | d) + \lambda_{uw} \log P(f_{uw} | d)
\]

\[
P(f_{unigram} | d) = \sum_{q_i} \left( \frac{|D|}{|D| + \mu} \cdot c(q_i, D) + \frac{\mu}{|D| + \mu} \cdot c(q_i) \right)
\]

\[
P(f_{ow} | d) = \sum_{q_i, q_{i+1}} \left( \frac{|D|}{|D| + \mu} \cdot c(#1(q_i, q_{i+1}), D) + \frac{\mu}{|D| + \mu} \cdot c(#1(q_i, q_{i+1})) \right)
\]

\[
P(f_{uw} | d) = \sum_{q_i, q_{i+1}} \left( \frac{|D|}{|D| + \mu} \cdot c(#uw8(q_i, q_{i+1}), D) + \frac{\mu}{|D| + \mu} \cdot c(#uw8(q_i, q_{i+1})) \right)
\]
Outline Today

- Text Clustering & Retrieval
- K-means
- Topic model (PLSI and LDA)
- Refined Document Representation
Clustering: what’s the purpose

• Partition unlabeled instances (e.g., documents) in a dataset into disjoint subsets of clusters
  • Instances within a cluster are very similar
  • Instances in different clusters are very different
• Discover new categories in an unsupervised manner (no sample category labels provided).
• Text clustering
  • The instances are usually documents.
  • We usually perform clustering based on document contents (e.g., using words as features).
Clustering: an Example
Hierarchical Clustering

• Build a tree-based hierarchical taxonomy \textit{(dendrogram)} from a set of unlabeled examples.

• Recursive application of a standard clustering algorithm can produce a hierarchical clustering.
Properties of documents

• Document described by a set of features (properties)
• Content represented by word features (most common)
  • Individual words (unigrams)
  • Phrases (bigrams)
• Can leverage annotations
  • Names of people, locations, organizations
  • Events
  • Relationships
• Can use metadata
  • Date written, author
  • Genre
  • Popularity
  • Manually assigned keywords or subject headings
• Features (properties) are task-dependent
K-means

• A “hard” clustering algorithm
  • Each object can only belong to a single cluster

• Input
  • k: the number of clusters
  • (Optional): the initial $k$ centroids (can be just random centroids)

• Output
  • The final centroids of the $k$ clusters (as the representation of the $k$ clusters)
  • Object-cluster assignment
K-means: an Example

• k initial "means" (centroids) (in this case k=3) are randomly generated within the data domain (shown in color).

• Note that centroids are not instances (but a centroid can have the same representation as an instance)
K-means: an Example

• Partition objects (square) into k-clusters based on the current centroids (round)

• Compute the distance/similarity between objects and centroids, assign objects to the nearest (most similar) centroids.
K-means: an Example

• Re-estimate the centroids based on the current object assignment.
K-means: an Example

• Iterate until the centroids and partitions converge.

• Example in IR:
  • Documents as objects
  • Bag-of-words features with TF-IDF weighting
  • Similarity/distance based on cosine similarity
K-means for document clustering

• Represent each document as vectors.
  • (most simple method): each word is a dimension (VSM)
  • Can include other features as dimensions

• Define clusters based on centroids $c$ (e.g., what is an “average” instance in a cluster looks like):

$$
\mu(c) = \frac{1}{|c|} \sum_{\tilde{x} \in c} \tilde{x}
$$

• Reassignment of instances to clusters is based on distance to the current cluster centroids.

• Iterate until converge …
Distance Metrics

• Euclidian distance ($L_2$ norm):

$$L_2(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

• $L_1$ norm:

$$L_1(\vec{x}, \vec{y}) = \sum_{i=1}^{m} |x_i - y_i|$$

• Cosine Similarity (transform to a distance by subtracting from 1) – most common for document clustering

$$1 - \frac{\vec{x} \cdot \vec{y}}{|\vec{x}| \cdot |\vec{y}|}$$
“Soft” Clustering

• K-means is an example of “hard” clustering algorithms
• Each document belongs to only one cluster?
• Documents in a cluster may be close or far away from the centroid
• For documents far away from the centroid, it’s association with that cluster is somewhat weak …
• Quantify the association between instances & clusters

<table>
<thead>
<tr>
<th></th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document 1</td>
<td>0.74</td>
<td>0.45</td>
<td>0.01</td>
</tr>
<tr>
<td>Document 2</td>
<td>0.12</td>
<td>0.38</td>
<td>0.01</td>
</tr>
<tr>
<td>Document 3</td>
<td>0.09</td>
<td>0.19</td>
<td>0.94</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LSI has a similar idea ...

- We learn some hidden concept dimensions
- Represent each document as vectors in the hidden dims
- We can somehow consider hidden dims as clusters
- Then, the document vectors define document-cluster assignments.
Latent Semantic Indexing (LSI)

- We want even more compact representation.
- We just use the most important $n$ dimensions ($n \ll \text{rank}(C)$).
- The transformed representation is an approximation of the original one.

$$C_{k \times m} \approx U_{k \times n} S_{n \times n} V_{m \times n}^T$$

\( n = 3 \)

\[
\begin{array}{c|c|c|c|c}
\text{index} & H1 & H2 & H3 & H4 \\
-0.18 & -0.46 & -0.33 & -0.07 \\
\text{retrieval} & -0.16 & -0.45 & -0.10 & 0.58 \\
\text{search} & -0.21 & -0.46 & -0.16 & -0.53 \\
\text{information} & -0.65 & -0.21 & 0.48 & 0.04 \\
\text{data} & -0.40 & 0.33 & -0.19 & 0.50 \\
\text{computer} & -0.37 & 0.31 & 0.39 & -0.29 \\
\text{science} & -0.42 & 0.35 & -0.66 & -0.21 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c}
\text{D1} & D2 & D3 & D4 \\
H1 & -0.37 & -0.33 & -0.60 & -0.63 \\
H2 & -0.61 & -0.61 & 0.27 & 0.42 \\
H3 & 0.20 & -0.31 & 0.69 & -0.62 \\
H4 & 0.67 & -0.65 & -0.29 & 0.22 \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c|c}
\text{H1} & H2 & H3 & H4 \\
44.54 & 0 & 0 & 0 \\
0 & 26.94 & 0 & 0 \\
0 & 0 & 6.29 & 0 \\
0 & 0 & 0 & 2.69 \\
\end{array}
\]
PLSI: Probabilistic Latent Semantic Indexing

• The purpose is to be a probabilistic version of LSI
  • And the name sounds like so

• Well, technically very different idea
  • LSI based on SVD
  • PLSI based on generative language model

• PLSI: the corpus is generated as follows
  • select a document with \( P(d) \)
  • pick a “latent class” (a topic) \( z \) with \( P(z|d) \)
  • generate a word \( w \) with \( P(w|z) \)

PLSI: Probabilistic Latent Semantic Indexing

• Training
  • Learn models that can maximize the likelihood of the corpus (observed data)
  • Well, maximize the likelihood of the observed data is usually not our purpose, but it can be helpful to learn useful models
  • P(z): how popular/important a topic is in the corpus
  • P(d|z): the association between a document and a topic
  • P(w|z): the representation of a topic (word distributions)

• EM for estimation
  • Using an iterative algorithm to estimate the parameters
  • Until converge; or a reasonable large number of times ...

PLSI: model estimation (using EM)

• E-stem

\[ P(z|d,w) = \frac{P(z)P(d|z)P(w|z)}{\sum_{z'} P(z')P(d|z')P(w|z')} , \]

• M-stem

\[ P(w|z) = \frac{\sum_d n(d,w)P(z|d,w)}{\sum_{d,w'} n(d,w')P(z|d,w')} , \]

\[ P(d|z) = \frac{\sum_w n(d,w)P(z|d,w)}{\sum_{d',w} n(d',w)P(z|d',w)} , \]

\[ P(z) = \frac{1}{R} \sum_{d,w} n(d,w)P(z|d,w), \quad R \equiv \sum_{d,w} n(d,w) \]
LDA: Latent Dirichlet Allocation

- Conceptually very similar to PLSI
  - A set of topics $z$, $P(w|z)$, $P(d|z)$
- A different (more complex) generation process
  - Particularly the generation of $z$ is more complex
- Usually believed as better than PLSI


Figure 7: Graphical model representation of the smoothed LDA model.
“Soft” Clustering

• PLSI and LDA can be considered as examples of “soft” clustering (also called topic models)

• Two main components
  • Topic representation (each topic is a unigram LM)

<table>
<thead>
<tr>
<th></th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word1</td>
<td>0.74</td>
<td>0.45</td>
<td>0.01</td>
</tr>
<tr>
<td>Word2</td>
<td>0.12</td>
<td>0.38</td>
<td>0.01</td>
</tr>
<tr>
<td>word3</td>
<td>0.09</td>
<td>0.19</td>
<td>0.94</td>
</tr>
</tbody>
</table>

• Document-topic association, e.g. \( P(z|d) \) or \( P(d|z) \)

<table>
<thead>
<tr>
<th></th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document 1</td>
<td>0.74</td>
<td>0.45</td>
<td>0.01</td>
</tr>
<tr>
<td>Document 2</td>
<td>0.12</td>
<td>0.38</td>
<td>0.01</td>
</tr>
<tr>
<td>Document 3</td>
<td>0.09</td>
<td>0.19</td>
<td>0.94</td>
</tr>
</tbody>
</table>
How to improve retrieval?

• Clustering search results
  • Group top-ranked results into different topics
  • Showing only a few results for each topic
  • To avoid that the top-ranked results are biased towards only one particular topic ...

• Using clusters to improve document representation
  • Because a document is (relatively) short ...
  • Document representation is boosted by taken into account the clusters/topics it belongs to
Cluster-based Document Model

• Liu & Croft, SIGIR ’04
• Using k-means for clustering; unigram as features
• Represent a cluster as a language model

\[ P(t \mid \text{cluster}) = \lambda \cdot \frac{c(t, \text{cluster})}{\sum_{t_i} c(t_i, \text{cluster})} + (1 - \lambda) \cdot P(t \mid \text{corpus}) \]

• Smooth a document \( D \)’s MLE model using
  • The corpus model (the same as QL)
  • The model of the cluster \( D \) belongs to

\[ P(t \mid D) = \lambda_1 P_{\text{MLE}}(t \mid D) + \lambda_2 P(t \mid \text{cluster}) + (1 - \lambda_1 - \lambda_2) \cdot P(t \mid \text{corpus}) \]
Cluster-based Document Model

• Liu & Croft, SIGIR ’04

• Comparing to QL:
  • Improves mean AP by 5%-15% on different datasets

<table>
<thead>
<tr>
<th>Collection</th>
<th>Simple Okapi</th>
<th>QL+DM</th>
<th>QL+CBDM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP (K=2000)</td>
<td>0.2198</td>
<td>0.2179</td>
<td>0.2326 (+)</td>
</tr>
<tr>
<td>WSJ (K=2000)</td>
<td>0.2762</td>
<td>0.2958 (+)</td>
<td>0.3006 (+)</td>
</tr>
<tr>
<td>FT (K=2000)</td>
<td>0.2556</td>
<td>0.2610</td>
<td>0.2713 (+)</td>
</tr>
<tr>
<td>SJMN (K=2000)</td>
<td>0.2098</td>
<td>0.2032</td>
<td>0.2171 (+)</td>
</tr>
<tr>
<td>LA (K=2000)</td>
<td>0.2279</td>
<td>0.2468 (+)</td>
<td>0.2590 (+)</td>
</tr>
<tr>
<td>FR (K=1000)</td>
<td>0.2644</td>
<td>0.2875</td>
<td>0.3316</td>
</tr>
</tbody>
</table>
Cluster-based Document Model

• Liu & Croft, SIGIR ’04

• The size of the cluster matters a little bit
  • More clusters: finer-grained topics
  • Fewer clusters: ...
  • What is the appropriate k value? No theory
  • Mostly k is heuristically parameter you need to tune

Table 3. Retrieval results (in average precision) using different number (K) of clusters generated by K-means clustering. Retrieval model is query likelihood model with CBDM as the document model.

<table>
<thead>
<tr>
<th>Collection</th>
<th>K=500</th>
<th>K=1000</th>
<th>K=2000</th>
<th>K=3000</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>0.2296</td>
<td>0.2298</td>
<td>0.2326</td>
<td>0.2318</td>
</tr>
<tr>
<td>FR</td>
<td>0.2643</td>
<td>0.3316</td>
<td>0.2993</td>
<td>0.2861</td>
</tr>
</tbody>
</table>
LDA-based Document Model

• Wei & Croft, SIGIR ’05

• Issues of hard clustering
  • Each document belongs to only one topic; a binary partition
  • Cluster-based smoothing may not be accurate for outliers
    (documents that are far away from the centroids and those right
    at the border lines of several clusters)

• Topic model
  • A technique that learns topics from a corpus
  • Each topic can be considered as a language model
  • Each document as generated by a mixture of different topics (and
    thus can be considered as soft clustering)

• Examples:
  • PLSA (a probabilistic version of LSI)
  • LDA (latent Dirichlet allocation)
LDA-based Document Model

• Wei & Croft, SIGIR ’05
• Similar to the cluster-based document model
• Smooth a document MLE model with
  • The corpus model
  • A mixture model of the document’s topics

\[
P(w \mid D) = \lambda \left( \frac{N_d}{N_d + \mu} P_{ML}(w \mid D) + (1 - \frac{N_d}{N_d + \mu}) P_{ML}(w \mid \text{coll}) \right) \\
+ (1 - \lambda) P_{lda}(w \mid D)
\] (7)

\[
P_{lda}(w \mid d, \hat{\theta}, \hat{\phi}) = \sum_{z=1}^{K} P(w \mid z, \hat{\phi}) P(z \mid \hat{\theta}, d)
\]
LDA-based Document Model

- Wei & Croft, SIGIR ’05
- In some collections, can improve CBDM by 10%

<table>
<thead>
<tr>
<th>Collection</th>
<th>QL</th>
<th>CBDM</th>
<th>LBDM</th>
<th>%chg over QL</th>
<th>%chg over CBDM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AP</td>
<td>0.2179</td>
<td>0.2326</td>
<td>0.2651</td>
<td>+21.64*</td>
<td>+13.97*</td>
</tr>
<tr>
<td>FT</td>
<td>0.2589</td>
<td>0.2713</td>
<td>0.2807</td>
<td>+7.54*</td>
<td>+3.46*</td>
</tr>
<tr>
<td>SJMN</td>
<td>0.2032</td>
<td>0.2171</td>
<td>0.2307</td>
<td>+13.57*</td>
<td>+6.26*</td>
</tr>
<tr>
<td>LA</td>
<td>0.2468</td>
<td>0.2590</td>
<td>0.2666</td>
<td>+8.02²</td>
<td>+2.93</td>
</tr>
<tr>
<td>WSJ</td>
<td>0.2958</td>
<td>0.2984</td>
<td>0.3253</td>
<td>+9.97*</td>
<td>+9.01*</td>
</tr>
</tbody>
</table>
To sum up

• Text clustering: an unsupervised technique for grouping similar text units in a corpus

• Applications of text clustering in IR

• Learn better representations of documents based on the document itself and the clusters/topics it belongs to (and essentially similar documents in the corpus)
  • Still, a document is just a limited sample
Thursday

- HW2 due
- Query expansion: boosting query representations
- Pseudo-relevance feedback