Text Processing on the Web

Week 10
Partitional and Hierarchical Text Clustering

Edited from source slides from the Stanford textbook site
Recap and Outline

• TC as different from standard machine learning
  – High dimensionality
  – Feature selection / weighting
  – Dataset skew / # of examples

• Clustering
  – Partitional Text Clustering
  – Hierarchical Text Clustering
  – Evaluation Methods
“The Curse of Dimensionality”

• Dealing with high dimensionality is difficult
  – While clustering looks intuitive in 2 dimensions, many of our applications involve 10,000 or more dimensions…
  – High-dimensional spaces look different: the probability of random points being close drops quickly as the dimensionality grows.
  – One way to look at it: in large-dimension spaces, random sparse vectors are almost all almost perpendicular.

Why?
What is clustering?

**Clustering**: the process of grouping a set of objects into classes of similar objects

- Most common form of *unsupervised learning* (no class labels)

Why cluster?

- Whole corpus analysis/navigation – Enabling better UIs
- For improving recall in search applications
- For better navigation of search results - Effective “user recall” will be higher
- For speeding up vector space retrieval - Faster search
Issues for clustering

- Representation for clustering
  - Document representation
    - Vector space? Normalization?
  - Similarity/distance metric

- Cluster properties
  - Number of clusters?
    - Given or need to figure out?
    - Avoid “trivial” clusters - too large or small (In search UIs, if a cluster is too large, then for navigation purposes you've wasted a user click without narrowing the set of docs)
  - Hard or soft assignments?

- Clustering algorithm properties
  - Completely data driven? Interactive or takes user data?
What makes docs “related”? 

- Ideal: semantic similarity.
- Practical: statistical similarity
  - We will use cosine similarity.
  - Docs as vectors
  - For many algorithms, easier to think in terms of a distance (rather than similarity) between docs.
Partitional Clustering
Partitioning Algorithms

• Partitioning method: Construct a partition of \( n \) documents into a set of \( K \) clusters
• Given: a set of documents and the number \( K \)
• Find: a partition of \( K \) clusters that optimizes the chosen partitioning criterion
  – Globally optimal: exhaustively enumerate all partitions
  – Effective heuristic methods: \( K \)-means and \( K \)-medoids algorithms
**K-Means**

- Assumes documents are real-valued vectors.
- Clusters based on **centroids** (aka the **center of gravity** or mean) of points in a cluster, $c$:
  \[
  \bar{\mu}(c) = \frac{1}{|c|} \sum_{\tilde{x} \in c} \tilde{x}
  \]
- Reassignment of instances to clusters is based on distance to the current cluster centroids.
  - (Or one can equivalently phrase it in terms of similarities)
K Means Example
(K=2)

Pick seeds
Reassign clusters
Compute centroids
Reassign clusters
Compute centroids
Reassign clusters
Converged!
Time Complexity

- Computing distance between two docs is $O(m)$ where $m$ is the dimensionality of the vectors.
- Reassigning clusters: $O(Kn)$ distance computations, or $O(Knm)$.
- Computing centroids: Each doc gets added once to some centroid: $O(nm)$.
- Assume these two steps are each done once for $I$ iterations: $O(IKnm)$. 
Efficiency: Medoid As Cluster Representative

- The centroid does not have to be a document.
- Medoid: A cluster representative that is one of the documents, the document closest to the centroid
- One reason this is useful
  - Consider the representative of a large cluster (>1000 documents)
  - The centroid of this cluster will be a dense vector
  - The medoid of this cluster will be a sparse vector

- Compare: mean/centroid vs. median/medoid
- How does this relate to the curse of dimensionality?
Seed Choice

• Results can vary based on seed selection.
• Some seeds can result in poor convergence rate, or convergence to sub-optimal clusterings.
  – Select good seeds using a heuristic (e.g., doc least similar to any existing mean)
  – Try out multiple starting points
  – Initialize with the results of another method

Example showing sensitivity to seeds

Select B and E as centroids:
Converge to \{A,B,C\} and \{D,E,F\}

Select D and F, converge to \{A,B,D,E\} \{C,F\}
How Many Clusters?

• Number of clusters $K$ is given
  – Partition $n$ docs into predetermined number of clusters

• Finding the “right” number of clusters is part of the problem
  – Given docs, partition into an “appropriate” number of subsets.
  – E.g., for query results - ideal value of $K$ not known up front - though UI may impose limits.
K not specified in advance

• Grade clustering versus a metric.
• Metric must have at least two parts: Total Benefit - Total Cost
  • **Benefit** (of a doc) = cosine sim to its centroid
  • **Cost** (constant cost c) in creating a new cluster

What happens if one of these criterion is missing?
Hierarchical Clustering
Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of unlabeled examples.

- One option to produce a hierarchical clustering is to recursively apply partitional clustering.

- What are other ways?
Hierarchical Agglomerative Clustering (HAC)

• **Agglomerative (bottom-up):**
  – Start with each document being a single cluster.
  – Eventually all documents belong to the same cluster.

• **Divisive (top-down):**
  – Start with all documents belong to the same cluster.
  – Eventually each node forms a cluster on its own.

• Does not require the number of clusters \( k \) in advance
• Merging/splitting history yields the binary hierarchy
• Assumes a binary symmetric distance function.
• Needs a termination/readout condition - why?
  – The final state in both agglomerative and divisive is no use.
Dendrogram: Document Example

- As clusters *agglomerate*, docs likely to fall into a hierarchy of “topics” or concepts.
Bisecting K-means

Almost identical to X-means as in Nomoto and Matsumoto’s summarization approach. How is it different?

• Divisive hierarchical clustering method using K-means

For \( l = 1 \) to \( k-1 \) do {
    Pick a leaf cluster \( C \) to split
    For \( j = 1 \) to \( \text{ITER} \) do {
        Use K-means to split \( C \) into two sub-clusters, \( C_1 \) and \( C_2 \)
        Choose the best of the above splits and make it permanent
    }
}

• Steinbach et al. suggest HAC is better than k-means but Bisecting K-means is better than HAC for their text experiments
Complexity

• In the first iteration, all HAC methods need to compute similarity of all pairs of \( n \) individual instances which is \( O(n^2) \).

• In each of the subsequent \( n-2 \) merging iterations, it must compute the distance between the most recently created cluster and all other existing clusters.
  – Since we can just store unchanged similarities

• In order to maintain an overall \( O(n^2) \) performance, computing similarity to each other cluster must be done in constant time.
  – Else \( O(n^2 \log n) \) or \( O(n^3) \) if done naively
Buckshot Algorithm

- Another way to an efficient implementation:
  - Cluster a sample, then assign the entire set
- First randomly take a sample of instances of size $\sqrt{n}$
- Run group-average HAC on this sample, which takes only $O(n)$ time.
- Use the results of HAC as initial seeds for K-means.
- Overall algorithm is $O(n)$ and avoids problems of bad seed selection. **Uses HAC to bootstrap K-means**
Cluster representative

• We want a notion of a representative point in a cluster

• Representative should be some sort of “typical” or central point in the cluster, e.g.,
  – point inducing smallest radii to docs in cluster
  – smallest squared distances, etc.
  – point that is the “average” of all docs in the cluster
    • Centroid or center of gravity
Example: $n=6$, $k=3$, closest pair of centroids

Centroid after first step.

Centroid after second step.
Outliers in centroid computation

• Can ignore outliers when computing centroid.

• What is an outlier?
  – Lots of statistical definitions, e.g.
  – *moment* of point to centroid > M \times \text{some cluster moment}.

  \uparrow

  Say 10.
Many variants to define closest pair of clusters

• “Center of gravity”
  – Clusters whose centroids (centers of gravity) are the most cosine-similar

• Average-link
  – Average cosine between pairs of elements

• Single-link
  – Similarity of the most similar (single-link)

• Complete-link
  – Similarity of the “furthest” points, the least similar
Single vs. Complete Link

- Use max sim pairs:
  \[ \text{sim}(c_i, c_j) = \max_{x \in c_i, y \in c_j} \text{sim}(x, y) \]
- Can result in long and thin clusters due to chaining effect.
  - When is it appropriate?

- Use min. sim of pairs:
  \[ \text{sim}(c_i, c_j) = \min_{x \in c_i, y \in c_j} \text{sim}(x, y) \]
- Makes “tighter,” spherical clusters that are typically preferable.

- After merging \( c_i \) and \( c_j \), the similarity of the resulting cluster to another cluster, \( c_k \), is:
  \[ \text{sim}((c_i \cup c_j), c_k) = \max(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k)) \]
  \[ \text{sim}((c_i \cup c_j), c_k) = \min(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k)) \]
Complete Link!

Single Link!
Group(wise) Average

- Use average similarity across all pairs within the merged cluster to measure the similarity of two clusters.

\[
sim(c_i, c_j) = \frac{1}{|c_i \cup c_j| (|c_i \cup c_j| - 1)} \sum_{\bar{x} \in (c_i \cup c_j)} \sum_{\bar{y} \in (c_i \cup c_j), \bar{y} \neq \bar{x}} \sim \bar{x}, \bar{y}
\]

- Compromise between single and complete link.

- Two options:
  - Averaged across all ordered pairs in the merged cluster
  - Averaged over all pairs \textit{between} the two original clusters

- Some previous work has used one of these options; some the other. No clear difference in efficacy
Computing Group Average Similarity

• Assume cosine similarity and normalized vectors with unit length.

• Always maintain sum of vectors in each cluster.

\[ \vec{s}(c_j) = \sum_{\vec{x} \in c_j} \vec{x} \]

• Compute similarity of clusters in constant time:

\[
\text{sim}(c_i, c_j) = \frac{(\vec{s}(c_i) + \vec{s}(c_j)) \cdot (\vec{s}(c_i) + \vec{s}(c_j)) - (|c_i| + |c_j|)}{(|c_i| + |c_j|)(|c_i| + |c_j| - 1)}
\]
Quick Question

- Consider agglomerative clustering on $n$ points on a line. Explain how you could avoid $n^3$ distance computations - how many will your scheme use?

This idea is actually employed in topical (text) segmentation!
Efficiency by approximation

• In standard algorithm, must find closest pair of centroids at each step
• Approximation: instead, find nearly closest pair
  – use some data structure that makes this approximation easier to maintain
  – simplistic example: maintain closest pair based on distances in projection on a random line
Multi-lingual docs

• E.g., Canadian government docs.
• Every doc in English and equivalent French
  – Must cluster by concepts rather than language
• Simplest: pad docs in one language with
dictionary equivalents in the other
  – thus each doc has a representation in both languages
• Axes are terms in both languages
Feature selection

Which terms to use as axes for vector space? Discussed previously last week

• Better is to use highest weight *mid-frequency* words – the most discriminating terms
• Pseudo-linguistic heuristics, e.g.,
  – drop stop-words
  – stemming/lemmatization
  – use only nouns/noun phrases
• Good clustering should figure out some of these
Labeling

• After clustering algorithm finds clusters - how can they be useful to the end user?
• Need pithy label for each cluster
  – In search results, say “Animal” or “Car” in the *jaguar* example.
  – In topic trees (Yahoo), need navigational cues.
    • Often done by hand, a posteriori.
How to Label Clusters

Actually a summarization task!

• Show titles of typical documents
  – Titles are easy to scan
  – Authors create them for quick scanning!
  – But you can only show a few titles which may not fully represent cluster

• Show words/phrases prominent in cluster
  – More likely to fully represent cluster
  – Use distinguishing words/phrases
    • Differential labeling, like diversity in summarization
  – But harder to scan
Labeling

• Common heuristics - list 5-10 most frequent terms in the centroid vector.
  – Drop stop-words; stem.

• Differential labeling by frequent terms
  – Within a collection “Computers”, clusters all have the word \textit{computer} as frequent term.
  – Discriminant analysis of centroids.

• Perhaps better: distinctive noun phrase
  – Such work also goes by the name \textit{keyphrase extraction}
Clustering Evaluation

Partitional vs. Hierarchical
Internal vs. External
Evaluation of clustering

• Most measures focus on computational efficiency
  – Time and space
• For application of clustering to search:
  – Measure retrieval effectiveness
What Is A Good Clustering?

• Internal criterion: A good clustering will produce high quality clusters in which:
  – the *intra-class* (that is, intra-cluster) similarity is high
  – the *inter-class* similarity is low
  – The measured quality of a clustering depends on both the document representation and the similarity measure used

• Similar to benefit in computing number of clusters – what wasn’t considered?
Cluster Quality Evaluation

- Simple measure: **purity**, the ratio between the dominant class in the cluster $\pi_i$ and the size of cluster $\omega_i$

$$Purity(\omega_i) = \frac{1}{n_i} \max_j (n_{ij}) \quad j \in C$$

- Others are entropy of classes in clusters (or mutual information between classes and clusters)
Cluster I: Purity = 1/6 (max(5, 1, 0)) = 5/6
Cluster II: Purity = 1/6 (max(1, 4, 1)) = 4/6
Cluster III: Purity = 1/5 (max(2, 0, 3)) = 3/5
## Rand Index

<table>
<thead>
<tr>
<th>Number of points</th>
<th>Same Cluster in clustering</th>
<th>Different Clusters in clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same class in ground truth</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>Different classes in ground truth</td>
<td>B</td>
<td>D</td>
</tr>
</tbody>
</table>

Min-Yen Kan / National University of Singapore
Rand index: symmetric version

\[ RI = \frac{A + D}{A + B + C + D} \]

<table>
<thead>
<tr>
<th>Number of points in clustering</th>
<th>Same Cluster in clustering</th>
<th>Different Clusters in clustering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same class in ground truth</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>Different classes in ground truth</td>
<td>B</td>
<td>D</td>
</tr>
</tbody>
</table>

Compare with standard Precision and Recall. What’s different?

\[ P = \frac{A}{A + B} \]
\[ R = \frac{A}{A + C} \]
Hierarchical Evaluation: User inspection

- Induce a set of clusters or a navigation tree
- Have subject matter experts evaluate the results
  - Subjective, may have more than one good tree
- Often combined with search results clustering
- Not clear how reproducible across tests.
- Expensive / time-consuming
Extrinsic evaluation

• Anything - including clustering - is only as good as the economic utility it provides
• For clustering: net economic gain produced by an approach (vs. another approach)
• Strive for a concrete optimization problem
• Examples
  – recommendation systems
  – clock time for interactive search
Resources

  - Cutting, Karger, Pedersen, Tukey
  - http://citeseer.ist.psu.edu/cutting92scattergather.html
- Data Clustering: A Review (1999)
  - Jain/Murty/Flynn
  - http://citeseer.ist.psu.edu/jain99data.html
- A Comparison of Document Clustering Techniques
- Initialization of iterative refinement clustering algorithms. (1998)
  - Fayyad, Reina, and Bradley
  - http://citeseer.ist.psu.edu/fayyad98initialization.html
- Scaling Clustering Algorithms to Large Databases (1998)
  - Bradley, Fayyad, and Reina
  - http://citeseer.ist.psu.edu/bradley98scaling.html