Lecture 6: Clustering

William Webber (william@williamwebber.com)

COMP90042, 2014, Semester 1, Lecture 6

What we'll learn today

- How to group documents into clusters by similarity
- How to evaluate clusters for quality
- The relationship between document and term clusters

Document clustering

Concept

- A “cluster” is a grouping of “similar” documents
- We can divide collection into (possibly overlapping) clusters
- Clusters can be hierarchical
- Hopefully, a cluster represents some common “meaning” or “topic” or “class”

Uses

- Form of unsupervised classification of the collection
- Corpus organization and browsing (particularly if hierarchical)
- Corpus summarization
- Result diversification

Similarity

- Need a concept of document "similarity"
- Ideally one that will also generalize to cluster "similarity"
- Cosine similarity for document similarity
- Clusters represented either by:
  - A representative (actual) document
  - An "average" of the documents (mean pseudo-document)

  both of which cosine similarity will handle

Clustering algorithm types

Three main types of clustering:

Agglomerative bottom-up (start with individual documents); naturally hierarchical

Partitioning top-down; partition into top-level groups; can be sub-partitioned

Hybrid combine or iterate both methods; a.k.a. "scatter-gather"

Agglomerative clustering

1. Place documents as singleton clusters in $\mathcal{C}$
2. Until $|\mathcal{C}| = 1$:
   2.1 Remove two most similar clusters c_1, c_2 from $\mathcal{C}$
   2.2 Join them in clusters $c_j = \{c_1, c_2\}$
   2.3 Place c_j in $\mathcal{C}$

- Creates hierarchy or (binary) "tree" of clusters
- Top of tree is whole collection
- Leafs of tree are documents

Computational considerations

Computational complexity

- Find most similar pair of documents: $O(n^2)$
- n steps to create full hierarchy
- Potentially $O(n^3)$
- ... or higher, if comparing (non-singleton) clusters is expensive

Compare clusters

Cluster similarity could be compared by:

- Most similar documents (aka single-link clustering)
- “Mean” document

Cluster comparison by most similar

1. Calculate upper triangular matrix of distances between doc pairs
2. For each doc save d_n, record its nearest neighbour in $\mathcal{P}$, going from rows to columns of triangular matrix; $|\mathcal{P}| = n - 1$.[1]
3. For $n - 1$ times:
   - 3.1 Remove closest pair (c_1, c_2) from $\mathcal{P}$
   - 3.2 Create $c_j = \{c_1, c_2\}$
   - 3.3 For $\langle c_a, c_b \rangle \in \mathcal{P}$[2]
     - 3.3.1 If $c_a = c_1$ or $c_a = c_2$:
     - 3.3.2 Replace c_a with c_j
     - 3.3.3 Else if $c_b = c_1$ or $c_b = c_2$:
     - 3.3.4 Replace c_b with c_j

- $O(n^2)$ time complexity (for creating $\mathcal{P}$)
- Can lead to poor clustering through "transitivity chains" (think long, thin clusters joined up end-wise)

---

[1] Modified 2014-03-20 to clarify directional nature of paired relationships

[2] Corrected 2014-03-20 from original version, which incorrectly swapped the order of the updated pairs in some conditions

Cluster comparison by mean

cluster mean Pseudo-document made by "averaging" all documents in the cluster

- Mean can be found by:
  - Averaging document vectors; or
  - Concatenating documents and creating vector
- When clusters combined, new mean from combining vectors
- Because docvec is sparse (most cells empty), update is quick
- Algorithm as "most similar", but update of $\mathcal{P}$ more expensive (all neighbours of c_1, c_2 must be re-neighboured)
- Still $O(n^2)$[3]

[3] Day and Edelsbrunner, "Efficient Algorithms for Agglomerative Hierarchical Clustering Methods", *J. Clsf*, 1984

Partition clustering

Concept

- Cluster at top level into (arbitrary) k clusters
- Can be sub-clustered (divide-and-conquer makes cheap)

Approach

- Select k documents "at random" as cluster seeds
- Assign documents to nearest center
- Iteratively improve centers, recluster
- Two implementations:

k medoid	Center is always (most central) document
k mean	After first iteration, center is mean pseudo-document

k means clustering

1. Randomly select k seeds as centroids $\mathcal{S} = \{s_1, \ldots, s_k\}$
2. Until "convergence":
   2.1 Assign each document to cluster c_i of nearest centroid s_i
   2.2 Calculate new centroid s_i as mean of c_i

- Relatively fast: $O(k \cdot n \cdot r)$, where r is number of repeats (may only require half-dozen or so) $\rightarrow O(n)$
- Sensitive to choice of seed documents (different clusters for different random seeds)
- Why is this not a complete disaster if seed documents are all next to each other?

Agglomerative, partitioning: why not both?

- Agglomerative robust, but expensive ($O(n_2)$)
- Partitioning fast ($\approx O(n)$), but seed-sensitive
- Combine two methods[4]:
  - Agglomerate sample of documents to pick good seeds
  - Then use k-means to improve these seeds

Buckshot scatter-gather

- Randomly select $\sqrt{k \cdot n}$ documents
- Agglomeratively cluster them to k seeds ($O(k \cdot n)$)
- Run k means clustering algorithm on these k seeds

[4] Cutting, Karger, Pedersen, and Tukey, "Scatter/Gather: a cluster-based approach to browsing large document collections", SIGIR 1998

Term clustering

Idea

- Just as we can cluster documents by similarity in term space
- . . . we can also cluster terms by similarity in document space

Uses

Term clusters potentially useful as:

- Identification, representation of concepts
- Fast query expansion

Cluster representation

Representing clusters in a human-understandable way:

- Term clusters naturally represented using terms in the cluster (somehow weighted)
- Document clusters not usefully represented by list of documents
- Common document cluster representation is by high-weighted terms
- For instance:
  - Take (calculate) mean document
  - Present highest-weighted (TF*IDF) terms in mean document

Co-clustering

Idea of representing document clusters by frequent term groups alerts us to connection between term and document clusters

- Cluster of documents are those with frequently co-occurring terms
- Cluster of terms are those that frequently co-occur in documents
- Two-stage document clustering:
  - First, create word clusters
  - Then, represent documents by the word cluster occurrence
  - Finally, cluster documents by word cluster
- Co-clustering (or bi-clustering)
  - Algorithm clusters both documents and terms at same time
  - Generally allow overlapping clusters

These ideas especially exploited in decomposition techniques (next lecture) and topic modelling (later in semester)

Evaluating cluster quality

Cluster literature distinguishes between *internal* and *external* evaluation:

- **Internal**: quality of separation of based on data itself
- **External**: compare to some external (human) standard cluster

These usually called "indices" rather than "metrics"

Internal evaluation

An internally "good" clustering will have two features:

Homogeneity Members of same cluster should be close

Separation Cluster should be far apart

Davies-Bouldin Index

σ_x average distance from member to centroid for cluster x (measures *homogeneity*)

$d(c_x, c_y)$ distance between centroids of clusters x and y (measures *separation*)

n number of clusters

$$\mathrm{DB} = \frac{i}{n} \sum_{i=1} n \max_{i \neq j} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \tag{1}$$

(Smaller values better)

Internal evaluation: is it circular?

- Internal evaluation violates IR evaluation rule:
  - purely algorithmic evaluation metric is not useful
  - must evaluate to human judgment
- However, computing a solution can be intractable
- ...when verifying (evaluating) that solution may be tractable
  - Think **P**v**NP**
- We can think of (say) DB as the aimed-at model
- Then valid to measure how close algorithms approach model
- Nevertheless, there is no "universal" objective function
- And different cluster algorithms will approximate different objective functions

External evaluation

- Have human- (or other reliable-) labelled classes
  - For example, labelled data set used for classifier evaluation (such as RCV1v2 for text)
- Compare agreement between gold standard and clustering

Rand index

a Number of pairs of documents in same set in gold standard G and in machine cluster M

b Num pairs in different sets for both G and M

c Num pairs in same set for G but different for M

d Num pairs in different sets for G but same for M

$$R = \frac{a+b}{a+b+c+d} \tag{2}$$

Looking back and forward

Back

- Document clustering an extension of document similarity to group documents
- May be flat partitioning or hierarchical clustering
- Intractability of creating "perfect" clustering (even according to formal model) leads to various heuristic or approximate solutions
- Evaluation can then be both to the theoretical model of cluster quality, or to human perception
- Terms can also be clustered into (we hope) "concepts"
- Natural interrelation between term

Looking back and forward

Forward

- Matrix decomposition methods (next week) do a form of bi-clustering
- More general field of topic modelling extends biclustering to identify overlapping "topics" in a text
- Multi-class text classification is a kind of clustering, but where the human specifies the clusters

Further reading

- Cutting, Karger, Pedersen, and Tukey, "Scatter/Gather: a cluster-based approach to browsing large document collections"[5], SIGIR 1998. Note only describes hybrid cluster methods, but also the use of clustering as an information exploration tool.
- Aggarwal and Zhai, "A Survey of Text Clustering Algorithms"[6], in Aggarwal and Zhai (ed.), *Mining Text Data*, Springer, 2012.
- Manning, Raghavan, and Schutze, Chapters 16 ("Flat clustering")[7] and 17 ("Hierarchical clustering")[8], *Introduction to Information Retrieval*, CUP, 2008.

---

[5] http://courses.washington.edu/info320/au11/readings/Week4.Cutting.et.al.1992.S
Gather.pdf

[6] http://www.charuaggarwal.net/text-cluster.pdf

[7] http://nlp.stanford.edu/IR-book/pdf/16flat.pdf

[8] http://nlp.stanford.edu/IR-book/pdf/17hier.pdf