Clustering

LING 570

Fei Xia

Week 10: 12/02/2009

Outline

- The task
- Clustering algorithms
- Evaluation measures
- Hw10

The task

- Input: A collection of objects
- Output: clusters

- Potential benefits:
  - Document clustering
  - Unsupervised POS tagging
  - Smoothing for LM
  - Generalization (e.g., for MT)
  - …

An example

- Input: All the words in a language

- Output: word clusters
  - Ex: Mon, Tues, Wed, …
  - Ex: Mike, Bryan, Joshua, …
  - Ex: claim, announce, declare, …
  - Ex: politicians, lawyers, salesmen, …

Another example

- Input: A collection of documents

- Output: document clusters
  - sports, politics, business, travel, …
  - docs with the similar style: e.g., news, chat room, talk shows, …
  - docs written by similar kinds of authors
  - docs focusing on the same topic

Questions

- What should a cluster represent?
  - Two objects are similar.
- How can we find good clusters?
- How can one evaluate clustering results?
- How can one benefit from clustering?

Similarity

- Between two instances

- Between an instance and a cluster

- Between two clusters

Some similarity functions

$$p = (p_1, p_2, ..., p_n) \text{ and } q = (q_1, q_2, ..., q_n)$$

- Euclidean distance:

$$dist(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Cosine similarity:

$$cos(\theta) = \frac{\sum_i p_i q_i}{\sqrt{\sum_i p_i^2} \sqrt{\sum_i q_i^2}}$$

Outline

- The task
- Clustering algorithms
- Evaluation measures
- Hw10

Types of clustering algorithms

- Flat vs. hierarchical clustering
  - Flat: partition n objects into k clusters
  - Hierarchical: create a hierarchy

- Hard vs. soft clustering
  - Hard clustering: each instance belongs to one cluster
  - Soft clustering: an instance can belong to multiple clusters (e.g., with different costs)

K-means vs. k-medoids

- k-means (MacQueen, 1967): Each cluster is represented by the center of the cluster

- k-medoids (Kaufman & Rousseeuw, 1987): Each cluster is represented by the medoid of the cluster

K-means algorithm

- Select k initial centroids at random.

- Repeat until there is no more change
  - Assign each object to the cluster with the nearest centroid.
  - Compute each centroid as the mean of the objects assigned to it.

K-means (cont)

- Relatively efficient: O(t k n)
  - t: number of iteration
  - k: number of clusters
  - n: number of objects

- Need to specify k
- Often terminates at a local optimum
- Applicable only when mean is defined (what about categorical data?)
- Trouble with noisy data and outliers

K-medoids algorithm

- Select k objects as the initial medoids

- Repeat until the medoids do not change
  - Assign each object to the cluster with the nearest medoid.
  - Find the new medoid for each cluster

Find the medoid of a cluster

- A medoid is an object in a cluster whose average dissimilarity to all the objects in the cluster is minimal.

- To find the medoid in a cluster
  - for each p, calculate $f(p) = \sum_q sim(p,q)$
  - choose p with the highest f(p).

Hierarchical clustering: Greedy, bottom-up approach

- Initialization: Create a separate cluster for each object

- Repeat until all the objects are in the same cluster:
  - Find two most similar clusters and merge

Outline

- The task
- Clustering algorithms
- Evaluation measures
- Hw10

Evaluation

- When compared with a gold standard
  - Rand index
  - Precision/recall
  - Variation of information
- Other methods:
  - Task-based evaluation
  - Human inspection

The setting

- Given a set of objects $S = \{O_1, ..., O_n\}$

- Partition $X = \{x_1, x_2, ..., x_r\}$
- Partition $Y = \{y_1, y_2, ..., y_r\}$

	In the same set in X	**In different sets in X**
In the same set in Y	a	d
In different sets in Y	c	b

Rand index

- http://en.wikipedia.org/wiki/Rand_index

	In the same set in X	**In different sets in X**
In the same set in Y	a	d
In different sets in Y	c	b

- Rand index calculates how well the two partitions agree.

$$R = \frac{a+b}{a+b+c+d} \qquad R \in [0,1]$$

Precision and recall

- Treat one partition as gold standard, and the other as system output
- For each pair in a set in the system partition, check whether it appears in one set in the gold standard
- Calculate precision, recall, and f-score.

Variations of information

- http://en.wikipedia.org/wiki/Variation_of_information

VI(X,Y) = H(X) + H(Y) - 2*MI(X,Y)

$$H(X) = -\sum_x P(x) log P(x)$$

$$MI(X;Y) = \sum_x \sum_y P(x,y) log \frac{P(x,y)}{P(x)P(y)}$$

Outline

- The task
- Clustering algorithms
- Evaluation measures
- Hw10

Hw10

- Unsupervised POS tagging
  - One method: clustering words
- Features: the previous words and the next words
  - Ex: “book L=the 15 R=of 3 …”
- Clustering algorithm: k-medoids algorithm (with cosine as the similarity function)
- Evaluation: tagging accuracy after mapping sys clusters to clusters in gold standard

Q1: create_vector.sh

- create_vector.sh train_file output_file word_list feat_list

- train_file: w1 w2 …
- word_list: word freq
- feat_list: word freq
- output_file: word fn1 fv1 fn2 fv2 …

“fn1 fv1”

- fn:
  - Format: featidx_(L|R)=x
  - Ex: “new york” appears 919 times, and “new” is the 37th feature (i.e., appearing on the 38th line) in the feature file.

    ➔ “york …. 37_L=new 919”

- fv:
  - The occurrence of the bigrams
  - Ex: “york … 37_L=new 919 …137_R=new 0 …”

The order of vector file

- Lines are sorted by the order in word_list

- (fn, fv) pairs are sorted by feature index.

  has 0_L=, 260 7_L=and 69 8_L='s 12 … 100_R=, 5 101_R=the 55 102_R=. 4 103_R=of 3 ...

- Why do we need to sort features?
  - It can make the calculation of cosine faster.

Q2: k-medoids.sh

- k-medoids.sh vector_file cluster_size sys_cluster

- vector_file is created in Q1
- cluster_size is an integer
- sys_cluster:
  - “medoid word1 word2 …”
  - medoid serves as the name of the cluster

Q2: k-medoids.hs

- similarity function: cosine

- initial medoids:

The i-th medoid is at line $x = (i-1) * \lfloor N/C \rfloor$

N is the number of vectors, C is the number of clusters.

Ex: N=100, C=34, $\lfloor N/C \rfloor$=2

Initial medoids are at line 0, 2, 4, .., 66.

Mapping sys cluster to gold cluster: greedy one-to-one

	g1	g2	g3
s1	2	$\boxed{10}$	9
s2	$\boxed{7}$	4	2
s3	0	9	$\boxed{6}$
s4	5	0	3

(1) find the largest number in the matrix
(2) remove both the row and the column
(3) repeat (1)-(2) until no more row left

s1 => g2 10
s2 => g1 7
s3 => g3 6

Acc=(10+7+6)/sum

Mapping sys cluster to gold cluster: greedy many-to-one

	g1	g2	g3
s1	2	$\boxed{10}$	9
s2	$\boxed{7}$	4	2
s3	0	$\boxed{9}$	6
s4	$\boxed{5}$	0	3

(1) find the largest number
(2) remove the row, but not the column
(3) repeat (1)-(2) until no more row left.

s1 => g2   10
s3 => g2   9
s2 => g1   7
s4 => g1   5

Q3: calc_acc.sh

- calc_acc.sh gold_cluster sys_cluster flag > map_file 2>acc_file

- gold_cluster: “cluster_name w1 w2 …”
- sys_cluster: “medoid w1 w2 …”
- flag:
  - 0: one-to-one mapping
  - 1: many-to-one mapping
- map_file: “sys_cluster => gold_cluster cnt”
- acc_file: Acc=xx

Q4: wrapper.sh

- wrapper.sh train_file word_list feawt_list cluster_size gold_cluster output_dir

- output_dir:
  - vectors: created by Q1
  - sys_cluster: created by Q2
  - res.*.map and res.*.acc: created by Q3

Q5: Fill out a table

word list	feat list	cluster size	gold cluster	output dir	1-to-1 Acc	many-to-1 Acc	running time
word.100	word.100	34	gold.100	100-100-34			
word.100	word.500	34	gold.100	100-100-34			
word.500	word.100	36	gold.500	500-100-36			
word.500	word.500	36	gold.500	500-500-36			
word.1000	word.100	39	gold.1000	1K-100-39			
word.1000	word.500	39	gold.1000	1K-500-39			
word.5000	word.100	41	gold.5000	5K-100-41			
word.5000	word.500	41	gold.5000	5K-500-41			

Calculating cosine function

$$cos(\theta) = \frac{\sum_i p_i q_i}{\sqrt{\sum_i p_i^2}\sqrt{\sum_i q_i^2}}$$

$$= \sum_i \frac{p_i}{\sqrt{\sum_i p_i^2}} \frac{q_i}{\sqrt{\sum_i q_i^2}}$$