Lecture 14b: K-Means, EM, PLSA, & ANNs

CS540 5/02/19

Announcements

Tuesday: in class presentations
PowerPoint or similar quality slides
20 minutes (~10 slides)
Active Participation

1 Week from tomorrow
Final paper

Where were we?

Natural Language Processing
- Comparing document similarity
- Comparing words, ignoring syntax
- Every document is a point in term space
- Bag of Words approach

Latent Semantic Analysis (LSA)
- Models corpus as a Gaussian distribution of documents in term space
- Computes major axes of variance
- Compresses data
- Rule of thumb: keep 85% of variance
- Angle between vectors as similarity measure

Document matrices

\[ \begin{pmatrix} A_1 & \ldots & A_n \end{pmatrix} \]

Find a subset of the terms that accurately clusters the documents

SVD: Example

\[ \begin{pmatrix} 2nd \text{ (right) singular vector} \\ 1st \text{ (right) singular vector} \end{pmatrix} \]

Input: 2-d dimensional points
Output:
1st (right) singular vector: direction of maximal variance,
2nd (right) singular vector: direction of maximal variance, after removing the projection of the data along the first singular vector.

\[ \psi_1 \text{ measures how much of the data variance is explained by the first singular vector.} \]

\[ \psi_2 \text{ measures how much of the data variance is explained by the second singular vector.} \]

Step #1: Clustering

Assumptions
- K: the number of clusters
- Every descriptor is a point in feature space

Approaches
- Generative models: fit K statistical distributions that are most likely to explain the data (today)
- K-Means
- Expectation Maximization (EM)
K-Means
Select K samples as random, make them cluster centers
- There are useful variations on this step
Iterate until no change:
- Assign every sample to the nearest cluster center
- Move every cluster center to the mean of the samples assigned to it

Analysis of K-Means
K-Means minimizes \( \sum_{s \in S} \| s - C(s) \|_2 \)
- Where \( S \) is the set of samples
- \( C(s) \) is the cluster center that sample \( s \) is assigned to
The assignment step reduces the value by changing the assignments \( C(s) \)
The mean computation step reduces the value by centering the means
Together, they hill climb to a local optima

Probabilistic Interpretation of K-Means
Every cluster center can be viewed as the mean of a Gaussian random process
- St. Dev. is the same in every direction
- St. Dev. is the same for every process
- Samples are assigned to the process that was most likely to create them
This interpretation supports
- Estimating the likelihood of a sample
- If K-Means is run more than once, select the solution most likely to generate the observed data

Problem with Unequal Variances
Implicit assumption: Gaussian processes with equal variance
Two Gaussians, but different variances
Need to model each cluster, not just its center

Measuring Cluster Variance
Measure covariance of PDFs:
- Let \( X \) be the \( D \times N \) set of mean-subtracted samples:
  \[
  X = \begin{bmatrix}
  s_1 & \cdots & s_D \\
  \vdots & \ddots & \vdots \\
  s_1 & \cdots & s_D 
  \end{bmatrix}
  \]
- Then \( \Sigma \) is the covariance matrix:
  \[
  \Sigma = \frac{1}{N} XX^T
  \]
The Hard-Assignment Problem

Which Gaussian generated these samples?

Solution: soft assignments

Which process generated the points in the middle?
- Either could have
- For every sample/cluster pair, compute the likelihood that the sample was generated by the cluster
  - Note: the value is never zero
  - This is called “soft assignment”
  - Samples not uniquely assigned to clusters

Even Harder Overlapping Gaussians

Expectation Maximization (EM)

Initialize clusters using random samples, uniform variance
Iterate until minimal change
- For every sample
  - Compute the likelihood that it could be generated by each cluster
  - Normalize likelihoods to sum to 1
  - The sample exists!
- For every cluster
  - Estimate mean and covariance using probability-weighted samples

Probabilistic Interpretation of EM

Every cluster represents a Gaussian random process
- Assignment (Expectation) step computes likelihood of generation for each sample/process pair
- Fitting (Maximization) step estimates the Gaussian parameters most likely to have generated the data
This supports:
- Estimating the likelihood of the data set
- Estimating the likelihood of any sample being created by any process

K-Means vs EM

EM is a more general model
- Works for any known distribution
EM fits far more parameters
- Good if enough training data is available
- Good if data fits the model
K-Means is simpler, more robust
- Better when dimensionality is high (1000+)
- Better when data may not be Gaussian
Probabilistic LSA
Probabilistic Latent Semantic Analysis
- Essentially, a clustering technique
- Models data as a mixture of Gaussians
- Uses Expectation Maximization (EM) to...
- Fit cluster centers and \( \Sigma \) matrices (deviations)
- Assign cluster likelihoods to each sample
- Requires the number of clusters (\( K \)) as a parameter
- Project a sample into PLSA space:
  - Calculate probability of each cluster generating the sample
  - Using means and \( \Sigma \)
  - Sample exists
  - Converts probabilities to likelihoods
  - Resulting vector is point in PLSA space

PLSA Example
Gaussian mixture
- Each Gaussian has a mean \( \mu \) and matrix of std. dev.'s \( \Sigma \)
- Probability of generating any sample can be computed from \( \mu \) and \( \Sigma \) using
  \[
  p(x) = \frac{1}{\sqrt{(2\pi)|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}
  \]
- EM fits \( \mu \) and \( \Sigma \) for all \( k \) to maximize the probability of generating the training data

Using PLSA
Training Data
- Run EM on training samples
- For every sample
  - Compute likelihood
  - Point becomes point in PLSA space
  - Dimensions are cluster likelihoods
  - Likelihood vector defines location

Test Data
- Compute likelihoods for trained clusters
- Becomes new point in PLSA space
- Compare to training data using Euclidean distance

Nearest Neighbors
Goal: Find the nearest sample in a gallery to a novel probe sample

Obvious solution:
- Measure distance from probe to every gallery instance
- Select instance with smallest distance

Obvious problem:
- \( O(n) \)

Approximate Nearest Neighbors
Goal: find nearest sample in gallery
- As often as possible
- When wrong, pick sample that is still close
- \( O(\log(n)) \)

Approach: binary trees
- Recursively divide feature space
- Each split divides gallery \( \sim 50/50 \)

ANN Illustrated
ANN Trees

Previous example thresholded feature values to divide feature space

Boundaries can be
- Arbitrary hyperplanes (i.e. diagonal)
- Non-linear boundaries (i.e. spheres)
- Example: Hierarchical K-Means

Problem:
- Samples near boundaries cause errors

Randomized Forests

Build multiple ANN trees
- With different boundaries
- Requires randomized boundary selection

Look up nearest neighbor in each tree
- Select best

Two standard algorithms
- Randomized Hierarchical K-Means
- FLANN

Proximity Forest

Problem: What if the feature space is unknown
- Imagine you have a similarity measure, but not a feature vector
- Examples
  - Similarity measure too expensive to run $O(n^2)$ times
  - Similarity measure over raw documents
  - Similarity measures over raw videos

Solution: proximity tree
- Select pivot sample at random
- Sort gallery by distance to pivot
- Split 50/50 nearest/farthest samples
- Repeat

Proximity Forest Results

HKM = Hierarchical K-Means; KDT = Randomized KD-Trees; PF = Proximity Forest

SIFT data: 128 dimensions; MSER data: 12 dimensions; 3D data: 3 dimensions

Source: O’Hara & Draper, Are You Using the Right Approximate Nearest Neighbor Algorithm?, WACV 2013