Case Study 2: Document Retrieval

Spectral Clustering

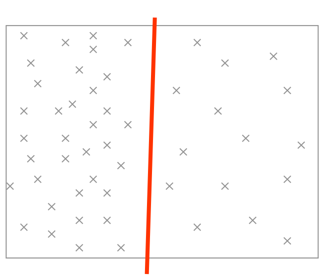
Machine Learning/Statistics for Big Data
CSE599C1/STAT592, University of Washington

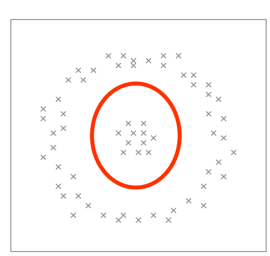
Emily Fox

February 14th, 2013

New Approach: Spectral Clustering

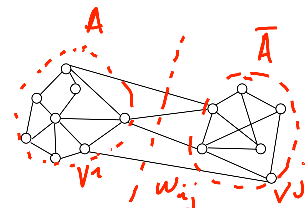
- **Goal:** Cluster observations
- **Method:**
  - Use similarity metric between observations
  - Form a similarity graph
  - Use standard linear algebra and optimization techniques to cut graph into connected components (clusters)





Setup

- Data: $x^1, \ldots, x^N$
- Similarity metric: s_{ij}



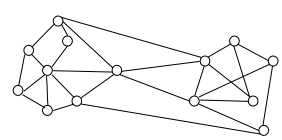
G = {V, E}

- Similarity graph
  - Nodes v^i
  - Edge weights $w_{ij} = f(s_{ij})$
- Problem: Want to partition graph such that edges between groups have low weights

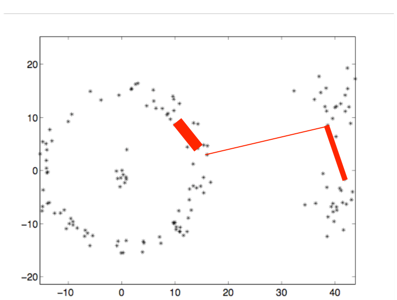
Graph Terminology I

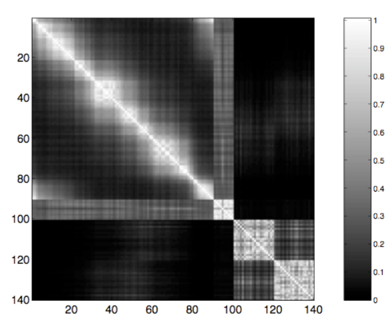
- Weighted adjacency matrix

$W = (w_{ij})\ i, j = 1, \ldots, N$



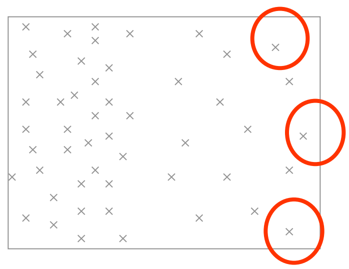
W





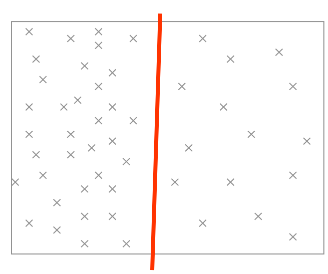
Issues with MinCut

- MinCut favors isolated clusters

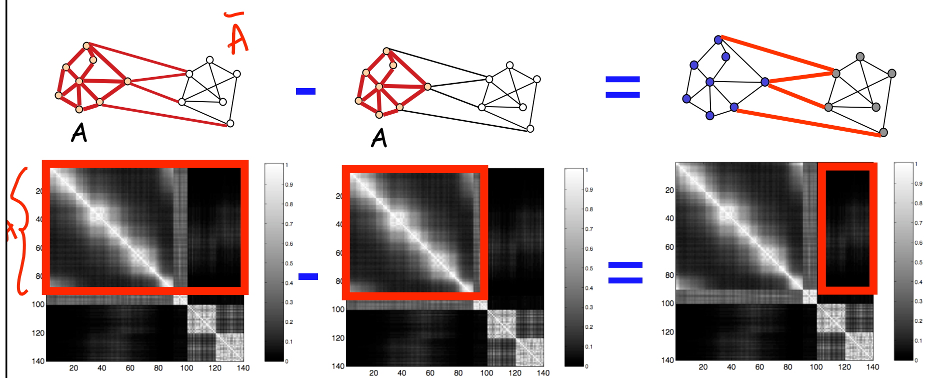


Cuts Accounting for Size

- Ratio cuts (RatioCut)
- Normalized cuts (Ncut)
- Lead to “balanced” clusters



Restating Cut Metric



$x^T D x \qquad x^T W x \qquad x^T \underbrace{(D - W)}_{\text{graph Laplacian } L} x$

$\mathbb{1}_A$

Ratio Cuts for General k

- Define cluster indicator variables:

$$F_{ij} = \begin{cases} 1/\sqrt{|A_j|} & v^i \in A_j \\ 0 & otherwise \end{cases} \qquad \begin{array}{l} F'_{\mathcal{A}} F_{\mathcal{A}} = I \\ F_{\mathcal{A}} \in \mathbb{R}^{N \times k} \end{array}$$

- RatioCut

$$\text{RatioCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} f'_{\mathcal{A}i} L f_{\mathcal{A}i} = \boxed{\text{Tr}(F'_{\mathcal{A}} L F_{\mathcal{A}})}$$

graph Laplacian

- Reformulating RatioCut problem

$$\min_{A_1, \ldots, A_k} \text{Tr}(F'_{\mathcal{A}} L F_{\mathcal{A}}) \ \text{ s.t. } \ F'_{\mathcal{A}} F_{\mathcal{A}} = I$$

Soln: F = 1st k eigenvectors of L

- Relaxation

$$\min_{F \in R^{N \times k}} \text{Tr}(F' L F) \ \text{ s.t. } \ F'F = I$$

sparse

Normalized Cuts for General k

- Define cluster indicator variables:

$$F_{ij} = \begin{cases} 1/\sqrt{\mathrm{vol}(A_j)} & v_i \in A_j \\ 0 & ow \end{cases} \qquad \begin{array}{c} F'_{\mathcal{A}}F_{\mathcal{A}} = I \\ F'_{\mathcal{A}}DF_{\mathcal{A}} = I \end{array}$$

- Reformulating RatioCut problem

$$\min_{A_1,\dots,A_k} \mathrm{Tr}(F'_{\mathcal{A}}LF_{\mathcal{A}}) \ \text{ s.t. } \ F'_{\mathcal{A}}DF_{\mathcal{A}} = I$$

$\Updownarrow \ F = D^{-1/2}H$

1 .. k

N

$v_1 \cdots v_k$

cluster rows using k-means

- Relaxation

$$\min_{H\in R^{N\times k}} \mathrm{Tr}(H' \underbrace{D^{-1/2}LD^{-1/2}}_{\triangleq L_{sym}} H) \quad \text{s.t. } H'H = I$$

- Solution:
  - H is matrix of first *k* eigenvectors of L_{sym}, which is equivalent to the approximate F being the first *k* eigenvectors of $L_{rw} = I - D^{-1}W$

Random Walks on Graphs

- Stochastic process with random jumps from v_i to v_j wp:

$P_{ij} = \frac{w_{ij}}{d_i}$ ← prob. of $v_i \to v_j$ transition

- Transition matrix:

$P = D^{-1}W$

- Connection to graph Laplacian:

$L_{rw} = I - D^{-1}W = I - P$

- Intuitively, want to partition graph s.t. random walk stays in cluster for a while and rarely jumps between clusters

Random Walks on Graphs

- Assume that stationary distribution exists and is unique. Then,

$\pi = (\pi_1, \ldots, \pi_N)$ $\quad \pi_i = \frac{d_i}{vol(V)}$

- Proposition: $\mathrm{Ncut}(A, \bar{A}) = P(A \mid \bar{A}) + P(\bar{A} \mid A)$

assume starting at stat

$P(X_1 \in A \mid X_0 \in \bar{A})$

- Proof:

$P(B|A) = \frac{P(X_0 \in A, X_1 \in B)}{P(X_0 \in A)}$

$\sum_{i \in A, j \in B} P(X_0 = i, X_1 = j) = \sum \pi_i P_{ij}$

$\frac{vol(A)}{vol(V)}$

$= \sum \frac{d_i}{vol(V)} \frac{w_{ij}}{d_i} = \frac{1}{vol(V)} \sum w_{ij}$

$= \frac{\sum_{i \in A, j \in B} w_{ij}}{vol(A)}$

- Minimizing normalized cuts is equivalent to minimizing the probability of transitioning between clusters

Case Study 3: fMRI Prediction

fMRI Prediction Task, LASSO Regression

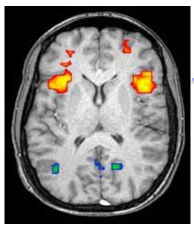
Machine Learning/Statistics for Big Data
CSE599C1/STAT592, University of Washington
Emily Fox
February 14th, 2013

big-P domain

fMRI Prediction Task

- **Goal:** Predict word stimulus from fMRI image

Can we read your brain?



Classifier
(logistic regression, kNN, …)

~~HAMMER~~
or
HOUSE

fMRI

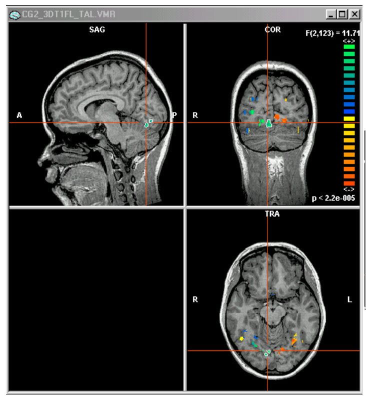
fMRI

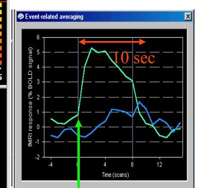
high res.

- ~1 mm resolution

pretty slow

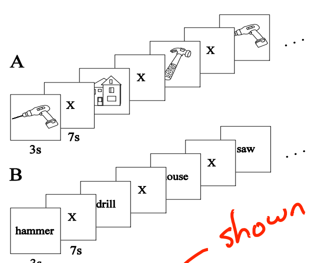
- ~1 image per sec.
- 20,000 voxels/image
- safe, non-invasive
- measures Blood Oxygen Level Dependent (BOLD) response





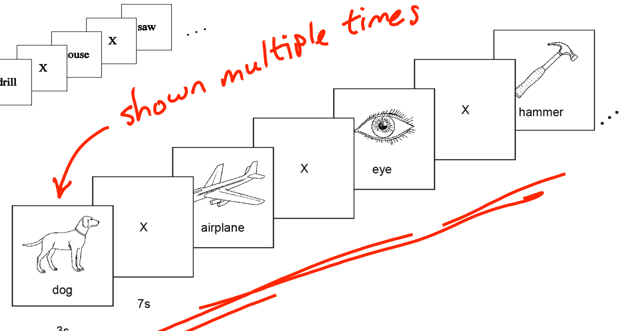
Typical fMRI response to impulse of neural activity

Typical Stimuli

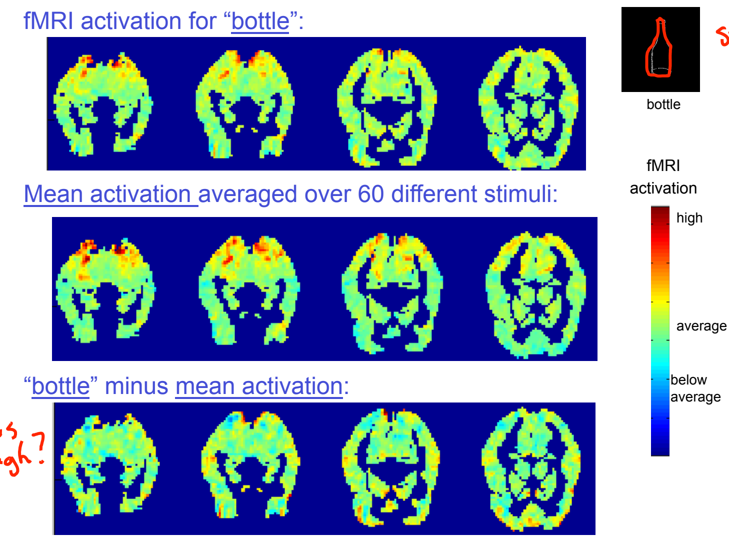


Each stimulus repeated several times

shown multiple times



fMRI Activation



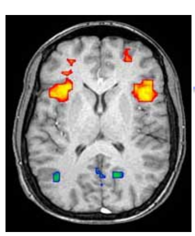
stimulus

is this enough?

fMRI Prediction Task

- **Goal:** Predict word stimulus from fMRI image
- **Challenges:** # of voxels = # of params
  - p >> N (feature dimension >> sample size)
  - Cost of fMRI recordings is high
  - Only have a few training examples for each word

many more param than obs.
what can we do?

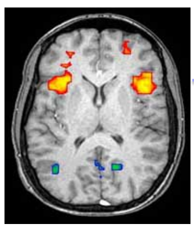


Classifier (logistic regression, kNN, …) → ~~HAMMER~~ or HOUSE

Zero-Shot Classification

- **Goal:** Classify words not in the training set
- **Challenges:**
  - Cost of fMRI recordings is high
  - Can't get recordings for every word in the vocabulary

Never showed "giraffe" in scanner



Classifier
(logistic regression, kNN, …)

~~HAMMER~~
or
HOUSE

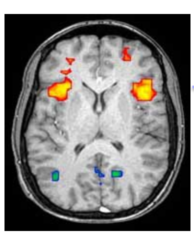
©Emily Fox 2013

19

Zero-Shot Classification

- **Goal:** Classify words not in the training set
- **Challenges:**
  - Cost of fMRI recordings is high
  - Can't get recordings for every word in the vocabulary
- We don't have many brain images, but we have a lot of info about the words and how they relate (co-occurrence, etc.)
- How do we utilize this "cheap" information?

many docs that contain "giraffe" also contain
"neck"
"animal"
"zoo"
⋮



Classifier
(logistic regression, kNN, …)

~~HAMMER~~
or
HOUSE

©Emily Fox 2013

20

Semantic Features

Google Trillion word corpus

Semantic feature values: "**celery**"	Semantic feature values: "**airplane**"
0.8368, eat	0.8673, ride
0.3461, taste	0.2891, see
0.3153, fill	0.2851, say
0.2430, see	0.1689, near
0.1145, clean	0.1228, open
0.0600, open	0.0883, hear
0.0586, smell	0.0771, run
0.0286, touch	0.0749, lift
…	…
…	…
0.0000, drive	0.0049, smell
0.0000, wear	0.0010, wear
0.0000, lift	0.0000, taste
0.0000, break	0.0000, rub
0.0000, ride	0.0000, manipulate

Zero-Shot Classification

- From training data, learn two mappings:
  - S: input image → semantic features
  - L: semantic features → word

A = { [ ] → "dog" } few

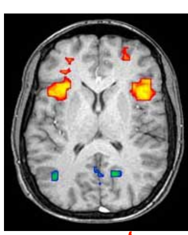
B = { [ ] → "dog" } many

- Can use "cheap" co-occurrence data to help learn L

from B

Training: { [ ] → [ ] → "dog" } N examples … N small

use both A + B



Features of word → Classifier (logistic regression, kNN, …) → ~~HAMMER~~ or HOUSE

new image

using B

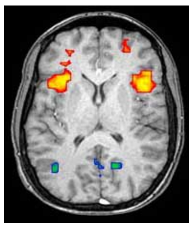
Predict [ ] $\xrightarrow{S}$ [ ] → "giraffe"

S ← learned from training data

fMRI Prediction Subtask

- **Goal:** Predict semantic features from fMRI image

Learning S: images → semantic features



Features of word

y^i

x^i

$$\begin{bmatrix} x_1^i \\ x_2^i \\ \vdots \\ x_{20,000}^i \end{bmatrix} \xrightarrow{S} \begin{bmatrix} y_1^i \\ y_2^i \\ \vdots \\ y_d^i \end{bmatrix}$$

let's consider each semantic feature separately

$\in \mathbb{R}^{20,000}$

$d = \#$ of semantic features

$\in \mathbb{R}^d$

Linear Regression – *review*

- Model: $y^i = \beta_0 + \beta_1 x_1^i + \dots + \beta_p x_p^i + \epsilon^i$

  $= \beta^T x^i + \epsilon^i$

  $\epsilon^i \sim N(0, \sigma^2) \Rightarrow y^i \sim N(\beta^T x^i, \sigma^2)$

- MLE: $\hat{\theta} = \arg\max_\theta \underbrace{\log p(D \mid \theta)}_{\sum_{i=1}^N \log p(y^i \mid x^i, \theta)}$ $\quad \theta = \{\beta, \sigma^2\}$

  $= \frac{-1}{2\sigma^2} \underbrace{\sum_i (y^i - \beta^T x^i)^2}_{\triangleq RSS(\beta) = \sum_i \epsilon^{i\,2}} - \frac{N}{\sqrt{2\pi\sigma^2}}$

  $\hat{\beta} = \arg\min_\beta NLL(\beta) = \arg\min_\beta RSS(\beta)$

  neg. log. like.

  MLE =

- Minimizing RSS= least squares regression

Linear Regression – *review*

- Taking the gradient
  - Reformulate objective

$$\underbrace{\begin{bmatrix} \epsilon^1 \\ \epsilon^2 \\ \vdots \\ \epsilon^N \end{bmatrix}}_{} = \underbrace{\begin{bmatrix} y^1 \\ \vdots \\ y^N \end{bmatrix}}_{y} - \underbrace{\begin{bmatrix} x_1^1 & \cdots & x_p^1 \\ \vdots & & \vdots \\ x_1^N & \cdots & x_p^N \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}}_{\beta}$$

small ↓ huge →

$$\frac{1}{2}RSS(\beta) = \frac{1}{2}(y - X\beta)^T(y - X\beta) = \frac{1}{2}\beta^T(X^TX)\beta - \beta^T(X^Ty) + \text{const.}$$

  - Set gradient = 0

$$\nabla_\beta NLL(\beta) = \nabla_\beta \frac{1}{2\sigma^2} RSS(\beta) = \frac{1}{\sigma^2}(X^TX\beta - X^Ty) = 0$$

$$\Rightarrow \hat{\beta}_{ML} = (X^TX)^{-1}X^Ty$$

low rank pxp matrix !!!

Ridge Regression

- Ameliorating issues with overfitting: penalization of weights = "regularization"
- New objective:

$$\min_\beta \underbrace{\sum_{i=1}^N (y^i - (\beta_0 + \beta^T x^i))^2}_{RSS} + \lambda \|\beta\|_2^2$$

don't penalize intercept term

$\beta^T\beta$

$$\min_\beta RSS(\beta) \quad \text{s.t.} \quad \|\beta\|_2^2 \le S$$

  - Reformulate:

$$F(\beta) = \underbrace{\frac{1}{2}\beta^T(X^TX)\beta - \beta^T(X^Ty) + \text{const.}}_{RSS(\beta)} + \frac{1}{2}\lambda\beta^T\beta$$

$$= \frac{1}{2}\beta^T(X^TX + \lambda I)\beta - \beta^T(X^Ty) + \text{const.}$$

  - Set gradient = 0

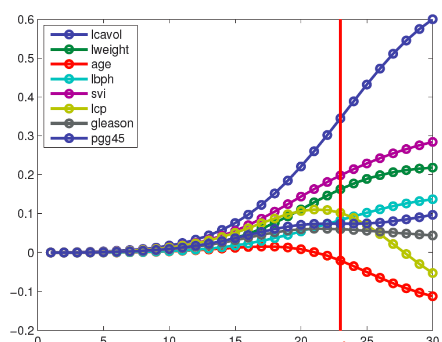
$$\hat{\beta}_{ridge} = (X^TX + \lambda I)^{-1}(X^Ty)$$

Ridge Regression

- Solution is indexed by the regularization parameter λ
- Larger λ high reg.
- Smaller λ low reg.
- As λ → 0 $\hat{\beta}_{ridge} \to \hat{\beta}_{ML}$
- As λ → ∞ $\hat{\beta}_{ridge} \to 0$

Ridge Coefficient Path

coeff.



S

CV soln

inc. λ

From
Kevin Murphy
textbook

$\|\beta\|_2^2 \le S$

- Typical approach: select λ using cross validation (CV)

Variable Selection

- Ridge regression: Penalizes large weights

min. this obj.) coeff. are very sensitive to what's inc. in model

- What if we want to perform "feature selection"?
  - E.g., Which regions of the brain are important for word prediction?
  - Can't simply choose predictors with largest coefficients in ridge solution
  - Computationally impossible to perform "all subsets" regression

discrete

2^p subsets of predictors .... can't do this

  - Stepwise procedures are sensitive to data perturbations and often include features with negligible improvement in fit ← greedy, but ∃ backtracking. –.

- Try new penalty: Penalize non-zero weights
  - Penalty:

$$\|\beta\|_1 = \sum_j |\beta_j|$$

  - Leads to sparse solutions
  - Just like ridge regression, solution is indexed by a continuous param λ

Acknowledgements

- Some material in this lecture was based on slides provided by:
  - Jianbo Shi – spectral clustering
  - Tom Mitchell – fMRI
  - Rob Tibshirani – LASSO