Ensembles

Joydeep Ghosh
Machine Learning “Summer” School
UT Austin, Jan 2015
Ensembles have become Mainstream

- Part of All Leading Commercial Packages/Environments
  - “Ensemble” node in SAS Enterprise Miner

- Python:

- R:
Ensembles and Multi-Learner Systems

• **Goal**
  – use multiple “learners” to solve (parts of) the same problem
    • Function approximation/regression
    • Classification
    • Clustering

• **Competing learners: ENSEMBLES**
  – Multiple looks at same problem

• **Cooperative learners: Mixture of experts and modular networks**
  – Divide and conquer
Generic Multi-learner System

Final Decision

Feedback

Confidence, ROC,

Combiner

Classification/Regression Model 1

Model 2

Model n

Data, Knowledge, Sensors,…
RoadMap

I: Motivation and History

II: Regression and III: Classification Ensembles:
   - Design Issues
   - Lessons Learned

IV: Cooperative learners

V: Cluster Ensembles
Motivation: Ensembles

• Different learners have different “inductive bias”
  – generalize differently from same training set

• Different properties
  – local vs. global
  – computation time / memory
  – susceptibility to outliers

• In designing/choosing one learner, several get created anyway!

• Hopes: better accuracy and better reliability
Very Early History

• **Combining Votes/Ranks**: Roots in French revolution!
  – Borda Counts: Jean-Claude de Borda, 1781
  – Condorcet’s rule, 1785
    • Given rank orders, Select “person” who defeats all others in 1-1 contests
  
  ➔ No notion of “true class”, just the rankings

• **Combining Estimators**: Linear opinion pool (Laplace)
Early History

• Multi-class Winner-Take All: Selfridge’s PANDEMONUIM (1958)
  – Ensembles of specialized demons
  – Hierarchy: Data, computational and cognitive demons
  – Decision: pick demon that “shouted the loudest”
    • Hill climbing; re-constituting useless demons, ….

• Nilsson’s Committee Machine (1965)
  – Pick max of C linear discriminant functions:
    \[ g_i(X) = W_i^T X + w_{i0} \]
Hybrid PR in the 70’s and 80’s

**Theory:** “No single model exists for all pattern recognition problems and no single technique is applicable to all problems. Rather what we have is a bag of tools and a bag of problems” Kanal, 1974

**Practice:** multimodal inputs; multiple representations,…

- Syntactic + structural + statistical approaches (Bunke 86)
- **multistage models:**
  - progressively identify/reject subset of classes
  - Cascade solutions, e.g. invoke KNN if linear classifier is ambiguous
- Combining multiple output types: 0/1; [0,1],…

➔ Designs were typically specific to application

Joydeep Ghosh  UT-ECE
Combining Multiple Models in Other Areas

PR/Vision: Data /sensor/decision fusion
AI: evidence combination (Barnett 81)
Econometrics: combining estimators (Granger 89)
Engineering: Non-linear control
Statistics: model-mix methods, Bayesian Model Averaging,..
Software: diversity

....
Common Issues

- Designing meta-learner or combiner
- Selecting the experts / learners / classifiers
- Determining training sets (same? automatic decomposition?)
- Estimating the gains, reliability

Data, Knowledge, Sensors,…
II: Ensembles for Regression
Output Combining for **Function Approximation**

- Meta-learners (trained)
  - 1) Put another MLP on top?
  - 2) Stacked Generalization (Wolpert 92)
    - train second layer model on leave-one-out samples

Problems?
Output Combining for Function Approximation (cont.)

• 3) Weighted average (Hashem 93)

\[ y_{av}(x) = \sum w_i y_i(x) ; \]
\[ w_i \geq 0 ; \sum w_i = 1 \]

  – Weights independent of x

  – Expression for Optimal weights involves inverse of C, the covariance matrix of the “mismatches”
    • Problem: entries of C are estimated, so are noisy; rows are correlated as models see same/similar data
      – collinearity problems encountered during inversion of C

• 4) More Practical: weight each model inversely proportional to its (estimated) MSE
Gains from the Averaging Combiner

• Analysis of Ensemble based on Simple Averaging, i.e.
• Define ambiguity for \( i \) th model, whose output is \( y_i(x_j) \) for the \( j^{th} \) training point, as the average squared mismatch between its answer and the ensemble answer

\[
\text{Ambiguity of } i \text{ th model} = E_x [(y_i(x) - y^{\text{ave}}(x))^2]
\]

where \( y^{\text{ave}}(x) \) is the ensemble answer (simple average)

• Can show MSE of the “averaging” ensemble = average MSE values of models in that ensemble – average value of ambiguity of the individual models.

So, need good but diverse models
  • rather overtrain than undertrain!!
II: Ensembles for Classification
Multi-Classifier Systems (MCS)

Raw Data from Observed Phenomenon

Classifier 1

Feature Set 1

Classifier m

Feature Set 2

Classifier N

Feature Set M

$find$

$comb$
DARPA/ONR sonar classification

- Winning team: UT/Tracor.
- Detection + classification of spatio-temporal signals
- Handling data volume
• DAS during hydraulic fracture treatment
DARPA Sonar Transients Classification Program (1989-)


Pre-processed Data from Observed Phenomenon

Ave/median/..
MCS: Combining the Outputs

- Hard decisions? – majority vote (classic)
- Rank order available? – Borda counts, etc.
- Soft decisions? (e.g. Posterior probability estimates) – treat as fn. Approx.
- More complex methods: The Multisensor Fusion Framework (Dasarathy 94)
  - Communication costs; ROC effects considered
  - Typically solution is of form: coupled “weighted average of decisions compared to threshold”
Output Combining of Soft Decisions

- Pick max. of (product; average; median; entropy weighting…)

- **Best way?**
  - Theory: (Kittler, Hatef & Duin, 96)
    - exact posteriors + independent assumptions $\Rightarrow$ optimal solution is (weighted) product (Bayesian)
    - but product is much more susceptible to errors than average, median
  - Empirical: Average, median slightly better
  - Outliers? Order statistics combining

- **How much can be gained?**
  - Look at effect on decision boundaries (Tumer and Ghosh, 96)
    - gains “proportional” to lack of correlation among the errors
Selecting the Inputs

• Bagging (Breiman, 92)
  – Gains where sensitivity is high

• Boosting (Schapire, 90)
  – Spl. Case of Adaptive Reweighting and Combining (ARC) methods
Bagging (Bootstrap Aggregating)

- Variance reduction technique
- Method:
  - create bootstrap (sampling with replacement) replicates of dataset
  - fit a model to each replicate
  - combine predictions by averaging or voting
- Properties
  - stabilizes unstable models
    - Decision trees, neural nets
    - Prefer deeper trees (why??)
  - easily parallelizable; implementable
- Ref: [www.stat.berkeley.edu/users/breiman/](http://www.stat.berkeley.edu/users/breiman/)
FIGURE 8.10. Error curves for the bagging example of Figure 8.9. Shown is the test error of the original tree and bagged trees as a function of the number of bootstrap samples. The orange points correspond to the consensus vote, while the green points average the probabilities.
Boosting

• Goal: improving classification, esp. with weak models

• Method:
  – sequentially fit models
  – later models see more of the samples mispredicted by earlier ones (input reweighting)
  – combine using weighted average (later models get more weight)

• Properties
  – reduces both bias and variance
  – slow to overfit
  – works well with “stumps”, naïve Bayes,…

• Several Variations
  – Additive logistic regression: Logitboost (Boosting does gradient descent on function space)
  – Max. Margin Boosting (Boosting implicitly maxes the margin)

• ref: www.cs.princeton.edu/~schapire/boost.html
AdaBoost Algorithm

- **Note:** both outputs $y_i$ and hypothesis (classifier) $h_i$ are +/- 1

**Input:** Training set $(x_1, y_1), \ldots, (x_m, y_m)$

**Algorithm:** Initialize $D_1(i) = 1/m$

**For** $t = 1, \ldots, T$

- Train a weak learner using distribution $D_t$
- Get weak hypothesis $h_t$ with error $\epsilon_t = \Pr_{x \sim D_t}[h_t(x) \neq y]$
- Choose $\alpha_t = \frac{1}{2} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right)$
- Update

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

where $Z_t$ is the normalization factor

**Output:** $h(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$
Toy Example from Schapire’s NIPS 2007 tute

- Weak classifier: single horizontal or vertical half-plane


Joydeep Ghosh  UT-ECE
Rounds 1 and 2

Round 1

Round 2

$\varepsilon_1 = 0.30$
$\alpha_1 = 0.42$

$\varepsilon_2 = 0.21$
$\alpha_2 = 0.65$
Round 3

$\varepsilon_3 = 0.14$

$\alpha_3 = 0.92$
Final Classifier

\[ H_{\text{final}} = \text{sign} \left( 0.42 + 0.65 + 0.92 \right) \]

Joydeep Ghosh   UT-ECE
Boosting as Additive Logistic Regression

- (on a novel “exponential loss”)
  - training error proof shows AdaBoost actually minimizes
  \[
  \prod_t Z_t = \frac{1}{m} \sum_i \exp(-y_i f(x_i))
  \]
  where \( f(x) = \sum_t \alpha_t h_t(x) \)

- on each round, AdaBoost greedily chooses \( \alpha_t \) and \( h_t \) to minimize loss

- exponential loss is an upper bound on 0-1 (classification) loss

- AdaBoost provably minimizes exponential loss
Loss Function Comparison (see HTF Ch 10)

- Additive model based on binomial deviance is more robust

**FIGURE 10.4.** Loss functions for two-class classification. The response is $y = \pm 1$; the prediction is $f$, with class prediction sign$(f)$. The losses are misclassification: $I(\text{sign}(f) \neq y)$; exponential: $\exp(-yf)$; binomial deviance: $\log(1 + \exp(-2yf))$; squared error: $(y - f)^2$; and support vector: $(1 - yf)_+$ (see Section 12.3). Each function has been scaled so that it passes through the point $(0, 1)$. 
Gradient Boosting

- See HTF Ch 10 for details and case studies
- R Code: `gbm` (Commercial: TreeNet)
- **Gradient descent in function space**
  - Less Greedy, more accurate, robust and interpretable than Adaboost
  - Can apply a variety of loss functions
  - Importance of variables (for trees): net information gain across all splits

**TABLE 10.2.** Gradients for commonly used loss functions.

<table>
<thead>
<tr>
<th>Setting</th>
<th>Loss Function</th>
<th>$-\partial L(y_i, f(x_i))/\partial f(x_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>$\frac{1}{2}[y_i - f(x_i)]^2$</td>
<td>$y_i - f(x_i)$</td>
</tr>
<tr>
<td>Regression</td>
<td>$</td>
<td>y_i - f(x_i)</td>
</tr>
</tbody>
</table>
| Regression  | Huber                   | $y_i - f(x_i)$ for $|y_i - f(x_i)| \leq \delta_m$
|             |                         | $\delta_m \text{sign}[y_i - f(x_i)]$ for $|y_i - f(x_i)| > \delta_m$
|             |                         | where $\delta_m = \alpha$-th-quantile{$|y_i - f(x_i)|$} |
| Classification | Deviance               | $k\text{th component: } I(y_i = G_k) - p_k(x_i)$ |
Gradient Boosted (Decision) Trees

Algorithm 10.3 Gradient Tree Boosting Algorithm.

1. Initialize $f_0(x) = \arg \min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.

2. For $m = 1$ to $M$:
   
   (a) For $i = 1, 2, \ldots, N$ compute
   
   $$r_{im} = -\left[ \frac{\partial L(y_t, f(x_t))}{\partial f(x_t)} \right]_{f=f_{m-1}}.$$

   (b) Fit a regression tree to the targets $r_{im}$ giving terminal regions $R_{jm}$, $j = 1, 2, \ldots, J_m$.

   (c) For $j = 1, 2, \ldots, J_m$ compute
   
   $$\gamma_{jm} = \arg \min_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma).$$

   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.

3. Output $\hat{f}(x) = f_M(x)$. 
Many Stumps win in the long run

**FIGURE 10.9.** Boosting with different sized trees, applied to the example (10.2) used in Figure 10.2. Since the generative model is additive, stumps perform the best. The boosting algorithm used the binomial deviance loss in Algorithm 10.3; shown for comparison is the AdaBoost Algorithm 10.1.
GBDT can rank order features

- See HTF 10.14 for case studies

**Figure 10.24.** Predictor variable importances separately for each of the four classes with lowest error rate for the demographics data.
Random Forests

- Bagging decision trees with additional randomization
  - generate bootstrap samples.
  - build one tree per bootstrap sample
  - increase diversity via additional randomization: randomly pick a subset of features of size $m << p$ to split at each node
    - Goal: Decrease correlation among trees without affecting bias
    - Should determine $m$ using out of bag (OOB) samples.
  - take equally weighted average of the trees
- **Theory**: Variance of average of $B$ (identically distributed, but not independent) variables $= \rho \sigma^2 + \frac{1 - \rho}{B} \sigma^2$.

Where $\rho = \text{pair-wise correlation}$ and $\sigma^2$ is variance of each variable.

Joydeep Ghosh  UT-ECE
Random Forests

- R: randomForest
- Reduce correlation among bagged trees
  - Consider only subset of variables at each split

**Algorithm 15.1 Random Forest for Regression or Classification.**

1. For $b = 1$ to $B$:
   
   (a) Draw a bootstrap sample $Z^*$ of size $N$ from the training data.
   (b) Grow a random-forest tree $T_b$ to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size $n_{min}$ is reached.
      
      i. Select $m$ variables at random from the $p$ variables.
      ii. Pick the best variable/split-point among the $m$.
      iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point $x$:

Regression: $\hat{f}_H^B(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the $b$th random-forest tree. Then $\hat{C}_H^B(x) = \text{majority vote} \{\hat{C}_b(x)\}_1^B$. 

Joydeep Ghosh   UT-ECE
Performance

- Easy to train/tune
  - And can get variable importance like GBDT
- Typically better than bagging, often comparable to boosting, not as good as GBDT
Summary of Committee Classifiers

• Variance reduction method
• Diversity is good (agree to disagree!)
• Good gains with little extra effort
• Provides estimate of decision confidence
  – Increased flexibility in accuracy-rejection rates
• “Standard”
IV: Co-operative Learners

a. Mixtures of Experts
b. Output Space Decomposition
Motivation for Modular Networks

(Sharkey 97)
- More interpretable localized models (Divide and conquer)
- Incorporate prior knowledge
- Better modeling of inverse problems, discontinuous maps, switched time series,..
- Future (localized) modifications
- Neurobiological plausibility

Varieties:
Cooperative, successive, supervisory,..
Automatic or explicit decomposition
Cop-operative Approach: Mixtures of Experts (MoE)

- Both $g_i(x)$’s and expert parameters adapted during training.

- Hierarchical versions possible

![Diagram of MoE model]

$$Y(x) = \sum g_i(x)y_i(x)$$

Joydeep Ghosh  UT-ECE
Generalizing MoE models

- Mixtures of X
  - $X = \text{HMMs, factor models, trees, principal components...}$

- State dependent gating networks
  - Sequence classification

- Mixture of Kalman Filters
  - Outperformed NASA’s McGill filter bank!

Beyond Mixtures of Experts

- Problems with soft-max based gating network

- Alternative: use normalized Gaussians
  - Structurally adaptive: add/delete experts
    - on-line learning versions
    - hard vs. soft switching; error bars, etc
  - Piaget’s assimilation & accommodation

Output Space Decomposition
(for multiclass problems)

Applying Multi-(binary) classifier systems for multi-class problems
- ECOC
- Hierarchical classifiers

History:
- Pandemonium, committee machine
  - “1 class vs. all others”
- Pairwise classification (how to combine?)
  - Limited
- Application specific solutions (80’s)

- **ECOC**: Error correcting output coding (Dietterich & Bakhiri, 95)
  - ve : groupings may be forced

- Desired: a general framework for natural grouping of classes
  - Hierarchical with variable resolution
  - Custom features
Error correcting output coding

- Given \( m \) distinct classes

  - Create an \( m \times n \) binary matrix \( M \)
    - Based on error correcting codes, or random matrices
  - Each class (“symbol”) is assigned **ONE** row (“code”) of \( M \).
  - Each column of the matrix divides the classes into **TWO** groups
    - Hence defines a two (meta)-class problem
  - Train the Base classifiers to learn the \( n \) binary problems.
    (example: 4 classes, 5 binary problems)
  - For test instance, get binary vector of classifier outputs and find closest match (Hamming distance)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0 0</td>
<td>1 1</td>
<td>1 1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1 0</td>
<td>1 0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0 1</td>
<td>1 1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0 1</td>
<td>0 0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

- **Drawback:** grouping of classes is not based on “proximity”
  - may result in complicated decision surfaces
Are Classes Truly “Orthogonal”? 

![Diagram showing a tree structure with categories like Division, Product Family, and SKU.](image-url)
Binary Hierarchical Classifier

- Building the tree:
  - Bottom-Up
  - Top-Down
- Hard & soft variants
- Provides valuable domain knowledge
- Simplified feature extraction at each stage

Joydeep Ghosh  UT-ECE
Hierarchical Grouping of Classes

- **Top down: Solve 3 coupled problems**
  - group classes into two meta-classes
  - design feature extractor tailored for the 2 meta-classes (e.g. Fisher)
  - design the 2-meta-class classifier

- **Solution using Deterministic Annealing:**
  - Softly associate each class with both partitions
  - Compute/update the most discriminating features
  - Update associations;
    - For hard associations: also lower temperature
  - Recurse

  - Fast convergence, computation at macro-level
Evolution of Class Partitioning

Iteration # -

Joydeep Ghosh  UT-ECE
BHC-top down for AVIRIS (remote sensing)

- KSC; 183 band subset of 224 bands
- Class 1-8 Uplands; 9-12 Wetland
V: Cluster Ensembles/Consensus Clustering
Consensus Clustering

- **given** multiple clusterings
  - possibly distributed in time and space
  - possibly non-identical sets of objects

- **obtain** a single integrated clustering
  - w/o sharing algos or features (attributes)

- **Example**: Customer segmentation for marketing
  - (needs-based, psychographics, brand choice,...)
Application Scenarios

• **Knowledge reuse**
  – Consolidate legacy, blackbox clusterings

• **Distributed Data Mining (Multiview clustering)**
  – Only some features available per clusterer
  – Only some objects available per clusterer

• **Improve quality and robustness**
  – Reduce variance
  – Good results on a wide range of data using a diverse portfolio of algorithms (robustness)
  – Estimate reasonable K
Cluster Ensemble Algorithms

- Probabilistic approaches
- Co-association based approaches
- Direct and other approaches
Cluster Ensemble Algorithms

Three main categories:

• **Probabilistic approaches** – mixture model for cluster ensembles, Bayesian cluster ensemble etc.

• **Ensemble Co-association based approaches** – CSPA, evidence accumulation
  – also linked with non-negative matrix factorization (NNMF) and correlation clustering.

• **Direct and heuristic based approaches** – hypergraph partitioning, approaches based on genetic algorithms and ant colony optimization algorithms.
Cluster Ensembles

- Given a set of provisional partitionings, we want to aggregate them into a single consensus partitioning, even without access to original features.

Clusterer #1

\( \Phi \) (individual cluster labels)

\( \lambda^{(1)} \)

\( \lambda^{(2)} \)

\( \Gamma \) (consensus labels)

\( \lambda \)
Cluster Ensemble Problem

- Let there be \( r \) clusterings \( \lambda^{(r)} \) with \( k^{(r)} \) clusters each
- What is the integrated clustering \( \lambda \) that optimally summarizes the \( r \) given clusterings using \( k \) clusters?

<table>
<thead>
<tr>
<th></th>
<th>( \lambda^{(1)} )</th>
<th>( \lambda^{(2)} )</th>
<th>( \lambda^{(3)} )</th>
<th>( \lambda^{(4)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>?</td>
</tr>
<tr>
<td>( x_4 )</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>( x_5 )</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>( x_6 )</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>?</td>
</tr>
</tbody>
</table>
| \( x_7 \) | 3                | 1                | 3                | ?                

*Much more difficult than Classification ensembles*
What is “best” consensus?

Consensus function: \( \Gamma: \lambda^{(q)}|q \in \{1,2,\ldots,M\} \rightarrow \hat{\lambda} \)

No ground truth -- maximize average of pairwise correlation with each individual solution (with \( k \) given)

Three most popular measures of correlation:

- Normalized Mutual Information (NMI):
  \[
  \text{NMI}(\lambda^{(a)}, \lambda^{(b)}) = \frac{I(\lambda^{(a)}, \lambda^{(b)})}{\sqrt{H(\lambda^{(a)})H(\lambda^{(b)})}}
  \]

- Adjusted Rand Index (ARI): number of overlaps between pairwise solutions appropriately normalized.

- Normalized Variation of Information (VI):
  \[
  \text{VI}(\lambda^{(a)}, \lambda^{(b)}) = 1 - \frac{2I(\lambda^{(a)}, \lambda^{(b)})}{H(\lambda^{(a)}) + H(\lambda^{(b)})}
  \]
ANMI (guess) vs. NMI (true) Correlation

(match with ground truth $\rightarrow$)

(match with consensus $\rightarrow$)

correlation coeff 0.923, except Yahoo (0.61)

Joydeep Ghosh  UT-ECE
Probabilistic Approaches

Mixture Model of Cluster Ensemble (Topchy et al. 2004):

Each element below is indexed by $y_{ij}$

<table>
<thead>
<tr>
<th></th>
<th>$\lambda^{(1)}$</th>
<th>$\lambda^{(2)}$</th>
<th>...</th>
<th>$\lambda^{(r)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>2</td>
<td>1</td>
<td>...</td>
<td>2</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>3</td>
<td>...</td>
<td>3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$x_N$</td>
<td>3</td>
<td>2</td>
<td>...</td>
<td>6</td>
</tr>
</tbody>
</table>
Probabilistic Approaches

Bayesian Cluster Ensemble (Wang et al. 2009):
Pairwise Similarity Based Approaches

- CSPA (next slide)

- Some other algorithms (generate the base solutions as part of algo)
  a. Evidence accumulation (Fred & Jain, 2005) obtains co-association matrix by random initialization of \( k \)-means.
  b. Monti *et al.* 2003 introduced perturbations in the base clusterings by resampling.
Cluster-based Similarity Partitioning (CSPA)

- Pairwise object similarity = # of shared hyperedges
\[ S = \frac{1}{r} \mathbf{H} \mathbf{H}^\dagger \]
- Cluster objects based on “consensus” similarity matrix
  - using e.g., graph-partitioning
Heuristics for Increasing Consensus

- Direct optimization – impractical

- Three efficient heuristics based on graph partitioning
  - HyperGraph Partitioning Alg. (HGPA)
    - $O(nkr)$
  - Cluster-based Similarity Partitioning Alg. (CSPA): Cluster the Rows
    - $O(n^2kr)$
  - Meta-Clustering Alg. (MCLA): Cluster the Columns
    - $O(nkr^2)$

- Supra-consensus function: performs all three and picks the one with highest ANMI (fully unsupervised)
Hypergraph Representation

- One hyperedge/cluster

- Example:

<table>
<thead>
<tr>
<th></th>
<th>$\lambda^{(1)}$</th>
<th>$\lambda^{(2)}$</th>
<th>$\lambda^{(3)}$</th>
<th>$\lambda^{(4)}$</th>
<th>$H^{(1)}$</th>
<th>$H^{(2)}$</th>
<th>$H^{(3)}$</th>
<th>$H^{(4)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>$v_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x_2$</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>$v_2$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x_3$</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>?</td>
<td>$v_3$</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$x_4$</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>$v_4$</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$x_5$</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>$v_5$</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$x_6$</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>?</td>
<td>$v_6$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$x_7$</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>?</td>
<td>$v_7$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Applications and Experiments

• application setups
  – Feature Distributed Clustering (FDC)
  – Object distributed clustering
  – Robust Consensus Clustering (RCC)
    • Different algorithms
    • Different resolutions (k)
FDC Example

- Data: 5 Gaussians in 8 dimensions
- Experiment: 5 clusterings in 2-dimensional subspaces
- Result: Avg. ind. 0.70, best ind. 0.77, ensemble 0.99
Experimental Results FDC

- Reference clustering and consensus clustering

- Ensemble always equal or better than individual

More than double the avg. individual quality in YAHOO (news web-document data, $k=40$, $d=2903$, $n=2340$)

<table>
<thead>
<tr>
<th>input and parameters</th>
<th>quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>subspace</td>
</tr>
<tr>
<td>2D2K</td>
<td>1</td>
</tr>
<tr>
<td>8D5K</td>
<td>2</td>
</tr>
<tr>
<td>PENDIG</td>
<td>4</td>
</tr>
<tr>
<td>YAHOO</td>
<td>128</td>
</tr>
</tbody>
</table>

Joydeep Ghosh    UT-ECE
Applications and Experiments

- Feature Distributed Clustering (FDC)
- Object distributed clustering
- Robust Consensus Clustering (RCC)
  - Different algorithms
  - Different resolutions (k)
Robust Consensus Clustering (RCC)

- Goal: Create an `auto-focus` clusterer that works for a wide variety of data-sets
- Diverse portfolio of 10 approaches
  - SOM, HGP
  - GP (Eucl, Corr, Cosi, XJac)
  - KM (Eucl, Corr, Cosi, XJac)
- Each approach is run on the same subsample of the data and the 10 clusterings combined using our supra-consensus function
- Evaluation using increase in NMI of supra-consensus results increase over Random
Robustness Summary

- Avg. quality versus ensemble quality
- For several sample sizes $n$ (50, 100, 200, 400, 800)
- 10-fold exp.
- $\pm 1$ standard deviation bars
Robustness Detail

- Individual performance versus RCC
- (a)
- (d)
Combining Clusterings of Different Resolutions

Motivation

• Robust combining of cluster solutions of different resolutions, produced in real life distributed data scenarios.

• Ensemble helps estimate the “natural” number of clusters
  – Use ANMI to guide $k$
Behavior of ANMI w.r.t. $k$ (#clusters)
Soft cluster ensembles

<table>
<thead>
<tr>
<th></th>
<th>$\Pi^{(1)}$</th>
<th>$\Pi^{(2)}$</th>
<th>$\Pi^{(3)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.7 0.2 0.1</td>
<td>0.1 0.7 0.2</td>
<td>0.6 0.3 0.1</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.9 0.1 0.0</td>
<td>0.0 0.8 0.2</td>
<td>0.8 0.2 0.0</td>
</tr>
<tr>
<td>$x_3$</td>
<td>0.9 0.0 0.1</td>
<td>0.1 0.4 0.5</td>
<td>0.5 0.5 0.0</td>
</tr>
<tr>
<td>$x_4$</td>
<td>0.2 0.6 0.2</td>
<td>0.1 0.2 0.7</td>
<td>0.2 0.07 0.1</td>
</tr>
<tr>
<td>$x_5$</td>
<td>0.1 0.9 0.0</td>
<td>0.0 0.1 0.9</td>
<td>0.0 0.5 0.5</td>
</tr>
<tr>
<td>$x_6$</td>
<td>0.0 0.2 0.8</td>
<td>0.8 0.1 0.1</td>
<td>0.1 0.2 0.7</td>
</tr>
<tr>
<td>$x_7$</td>
<td>0.1 0.2 0.7</td>
<td>0.7 0.1 0.2</td>
<td>0.1 0.3 0.6</td>
</tr>
</tbody>
</table>

- Can extend MCLA, CSPA; also apply info-theoretic clustering
- Often substantial gains (Punera & Ghosh 06)

Also use of consensus for detecting overlapping clusters. (Deodhar & Ghosh 06).
References: Consensus Clustering


• Survey containing extensive reference list: J. Ghosh and A. Acharya, Cluster Ensembles WIREs Data Mining and Knowledge Discovery Vol. 1 No. 4 pp. 305-315 2011

References: Ensembles

• Combining pattern classifiers: methods and algorithms
  L Kuncheva, John Wiley & Sons, 2004

• Ensemble Methods: Foundations and Algorithms

• The Elements of Statistical Learning (Several Chapters)
  Trevor Hastie, Robert Tibshirani, and Jerome Friedman,
  ElemStatLearn/

Joydeep Ghosh   UT-ECE
Summary

Ensembles deliver!

- substantial gains for tough problems
  - Not much more extra effort
  - Bagging and boosting getting commercialized
  - Adds robustness to results

- Modular approaches suitable for niche problems

- Even greater gains for clustering
Backups on Consensus
Direct Approaches

- Graph Partitioning Based Approaches
  a) HGPA
  b) MCLA
  c) HBGF
Relating Pairwise Similarity Based Approaches to Other Formulations

• Consensus clustering as non-negative matrix factorization (NNMF) Li et al. 2007:
  \[
  \arg\min_U \|S - U\|^2
  \]
  is the weighted average of the base co-association matrices and the matrix norm is the Frobenius norm.

• Consensus clustering as correlation clustering (Gionis et al. 2007):
  \[
  \hat{\lambda} = \arg\min_{\lambda} \sum_{(i,j):\lambda(x_i)=\lambda(x_j)} d_{ij} + \sum_{(i,j):\lambda(x_i)\neq\lambda(x_j)} (1 - d_{ij})
  \]
  \[d_{ij} \in [0,1]\] is the normalized count of co-occurrences of \(x_i\) and \(x_j\) in the same cluster.
HyperGraph Partitioning Alg. (HGPA)

- Partition the hypergraph so that a minimum number of hyperedges are cut
- Hypergraph partitioning is a well-known problem from e.g., VLSI
- We use HMETIS
Meta-CLustering Algorithm (MCLA)

Build a meta-graph such that
- vertex is cluster (vertex weight is cluster size)
- edge weight is similarity between clusters
  - Similarity = intersection/union (Jaccard distance between $h_a$ and $h_b$)
- Balanced partitioning of this r-partite graph (METIS)
- Assign each object to best matching meta-cluster