Ensemble Learning through Diversity Management: Theory, Algorithms, and Applications

Huanhuan Chen and Xin Yao

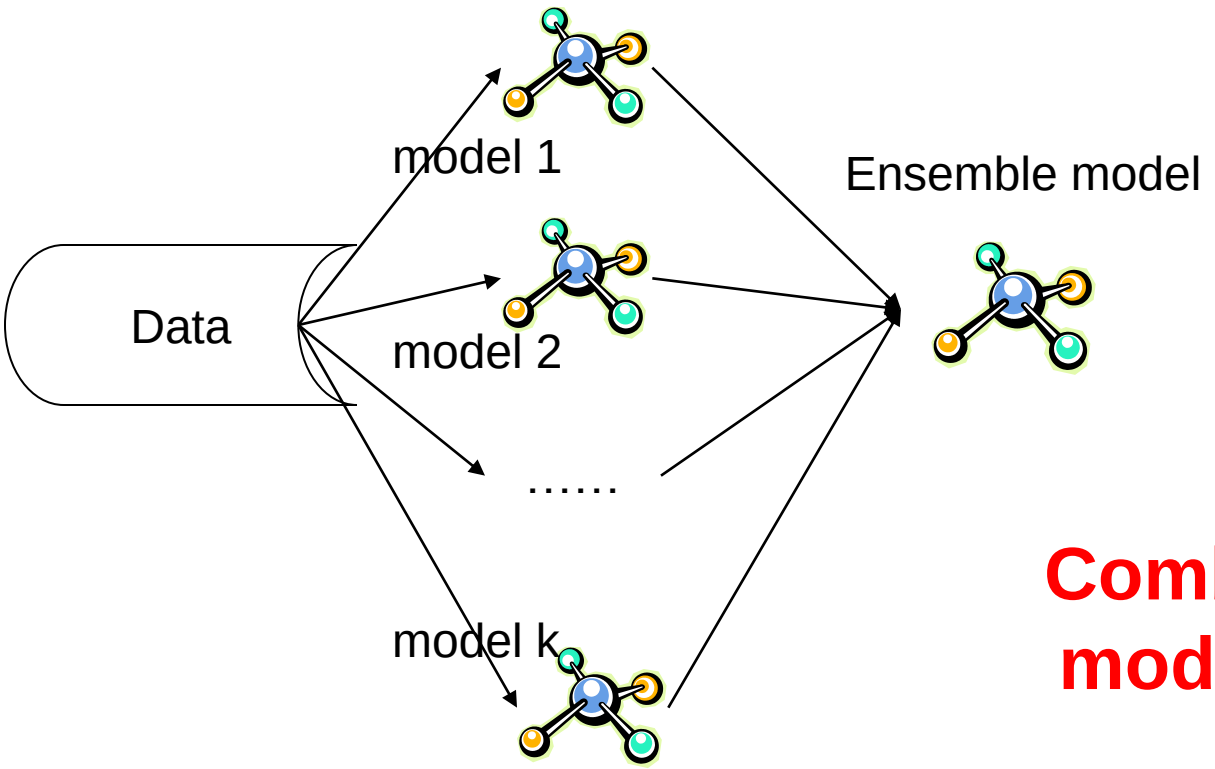
School of Computer Science
University of Birmingham

*Slides and references available at
http://www.cs.bham.ac.uk/~hxc/tutorial/

Outline

- An overview of ensemble methods
- Diversity generation methods
- Theoretical analysis of diversity
- Mange diversity in ensemble
  - Semi-supervised learning
  - Ensemble pruning
  - Multi-objective optimization
- Further topics and open discussions

Ensemble

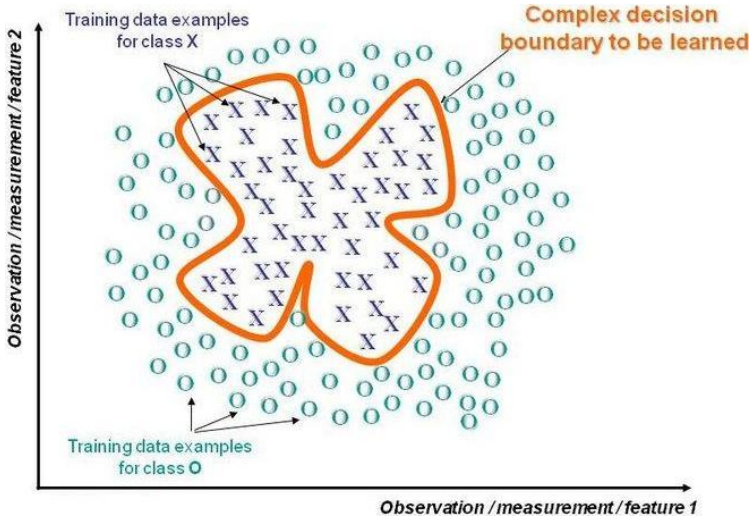


Combine multiple models into one!

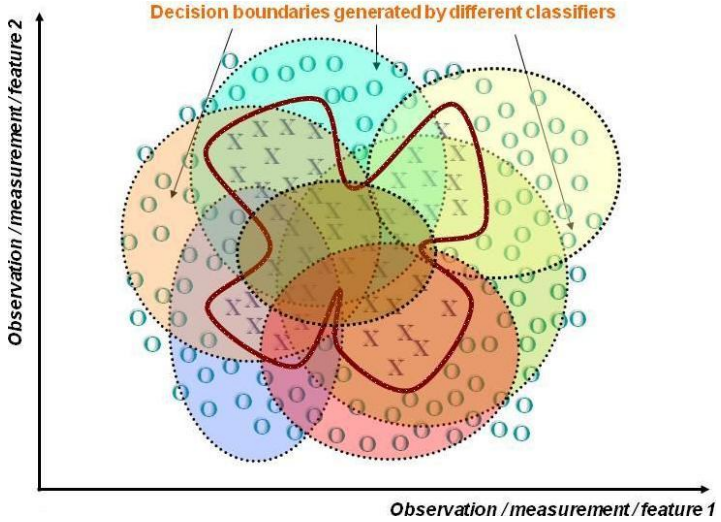
Ensemble is a group of learners that work together as a committee to solve a problem.

One Example

- Certain problems are just too difficult for a given classifier to solve
- Separate two classes “X” and “O”
- The weak classifier can only generate circular decisions



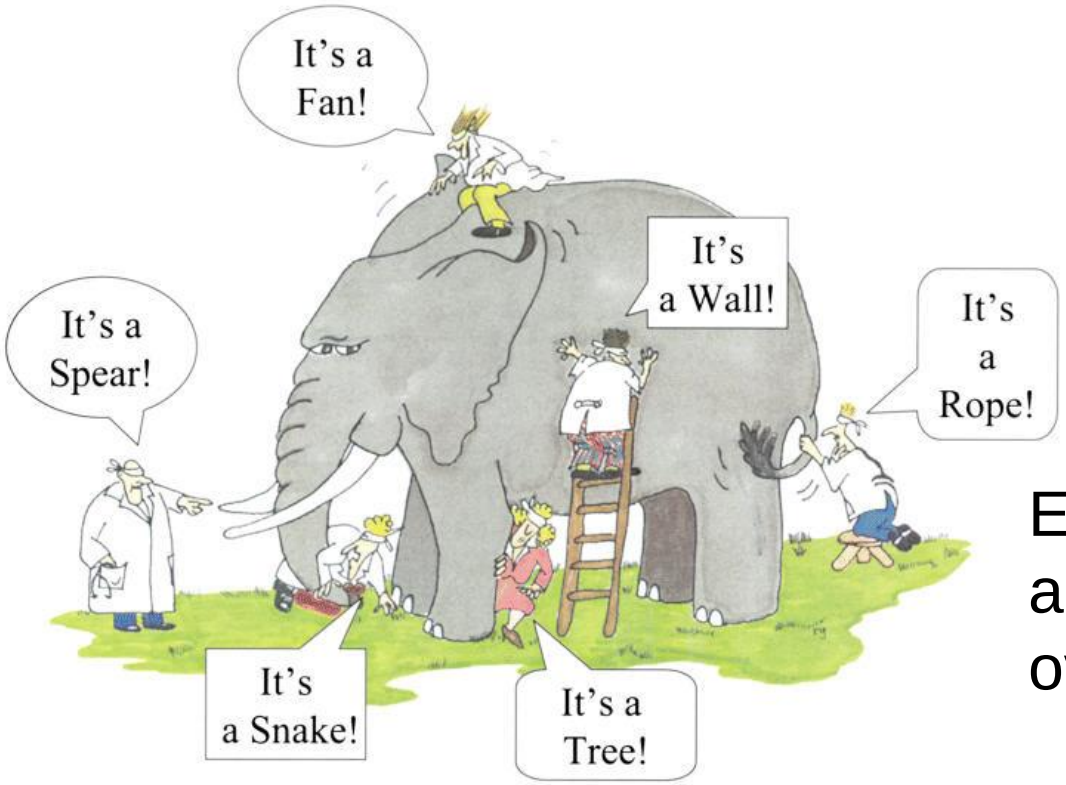
A complex decision that cannot be realized by circular boundaries



A combination of several circular boundaries can realize this complex boundary.

Images from wikipedia

Why Ensemble Works?

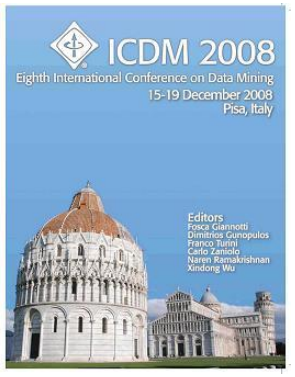


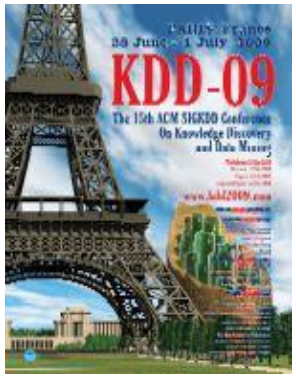
Ensemble model improves accuracy and robustness over single model methods

A complex problem can be decomposed into multiple sub-problems that are easier to understand and solve (divide-and-conquer approach)

Stories of Success

- Million-dollar prize
  - Improve the baseline movie recommendation approach of Netflix by 10.06% in accuracy
  - The top submissions (e.g. Pragmatic Chaos & The Ensemble) all combine several teams and algorithms as an ensemble





- Data mining competitions
  - Classification problems
  - Winning teams employ an ensemble of classifiers

Overview of Ensemble Methods

- Mixture of Experts [Jordan94]

  Divide and conquer

- Bagging [Breiman96]

  Bootstrapping on data

- Boosting [Schapire98]

  Recursively reweighting data.

- Random Forest [Breiman01]

  Randomly pick features and data to generate different classifiers (decision trees).

- Negative Correlation Learning [Liu99]

  Minimize the empirical training error and the correlation within the ensemble to generate ensembles.

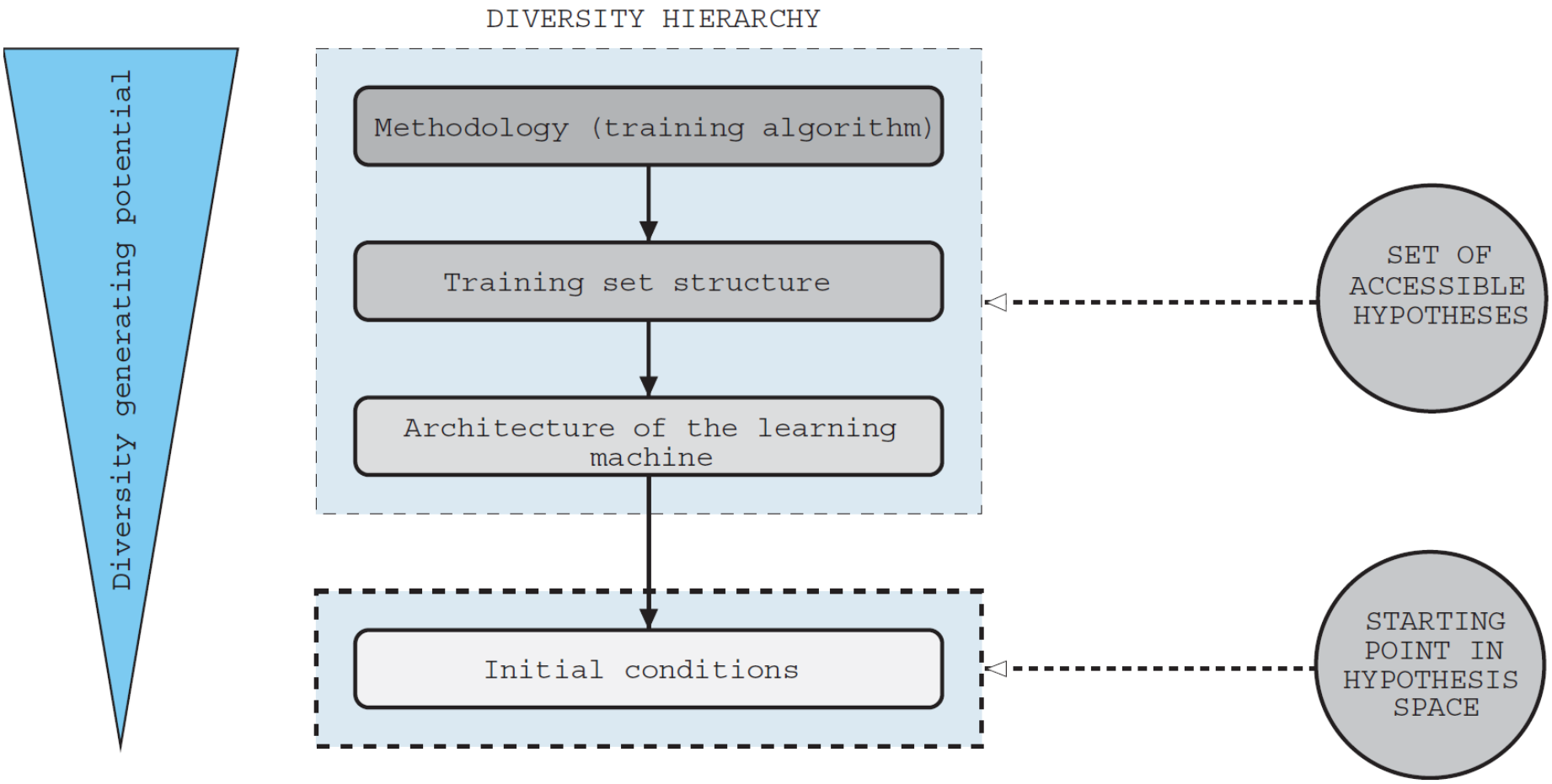
Outline

- An overview of ensemble methods
- Diversity generation methods
- Theoretical analysis of diversity
- Mange diversity in ensemble
  - Semi-supervised learning
  - Ensemble pruning
  - Multi-objective optimization
- Further topics and open discussions

Diversity Generation Methods

- Diversity Encouragement by Data Manipulation
- Diversity Encouragement by Architectures Manipulation
- Diversity Encouragement by Hypothesis Space Traversal

Diversity Hierarchy



Diversity Encouragement by Data Manipulation

- Data Distribution Manipulation
  - Bagging and Boosting
- Feature set Manipulation
  - Feature Subspace
- Feature & Distribution Manipulation
  - Random Forest

- Weak Learner is required!!!

Bootstrap & Bagging (1)

- Bootstrap
  - Sampling with replacement
  - Contains around 63.2% original records in each sample
- Bootstrap Aggregation
  - Train a classifier on each bootstrap sample
  - Use majority voting to determine the class label of ensemble classifier

[Breiman96]

Bootstrap & Bagging (1)

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

Bootstrap samples and classifiers:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x	0.1	0.2	0.3	0.4	0.5	0.5	0.9	1	1	1
y	1	1	1	-1	-1	-1	1	1	1	1

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

Combine predictions by majority voting

from P. Tan et al. Introduction to Data Mining.

Bootstrap & Bagging (3)

- Error Reduction
  - Under mean squared error, bagging reduces variance and leaves bias unchanged
  - Consider idealized bagging estimator:
  - The error is $\bar{f}(x) = E(\hat{f}_z(x))$

$$E[Y - \hat{f}_z(x)]^2 = E[Y - \bar{f}(x) + \bar{f}(x) - \hat{f}_z(x)]^2$$

$$= E[Y - \bar{f}(x)]^2 + E[\bar{f}(x) - \hat{f}_z(x)]^2 \geq E[Y - \bar{f}(x)]^2$$

  - Bagging usually decreases MSE

from Elder, John. From Trees to Forests and Rule Sets - A Unified Overview of Ensemble Methods. 2007.

Boosting (1)

- Principles
  - Boost a set of weak learners to a strong learner
  - Make records currently misclassified more important
- Example
  - Record 4 is hard to classify
  - Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

from P. Tan et al. Introduction to Data Mining.

Boosting (2)

- AdaBoost
  - Initially, set uniform weights on all the records
  - At each round
    - Create a bootstrap sample based on the weights
    - Train a classifier on the sample and apply it on the original training set
    - Records that are wrongly classified will have their weights increased
    - Records that are classified correctly will have their weights decreased
    - If the error rate is higher than 50%, start over
  - Final prediction is weighted average of all the classifiers with weight representing the training accuracy

Boosting (3)

- Determine the weight
  - For classifier *i*, its error is

$$\varepsilon_i = \frac{\sum_{j=1}^{N} w_j \delta(C_i(x_j) \neq y_j)}{\sum_{j=1}^{N} w_j}$$

  - The classifier's importance is represented as:

$$\alpha_i = \frac{1}{2} \ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

  - The weight of each record is updated as:

$$w_j^{(i+1)} = \frac{w_j^{(i)} \exp\left(-\alpha_i y_j C_i(x_j)\right)}{Z^{(i)}}$$

  - Final combination:

$$C^*(x) = \arg\max_y \sum_{i=1}^{K} \alpha_i \delta\left(C_i(x) = y\right)$$

Boosting (4)

- Explanation
  - Among the classifiers of the form:

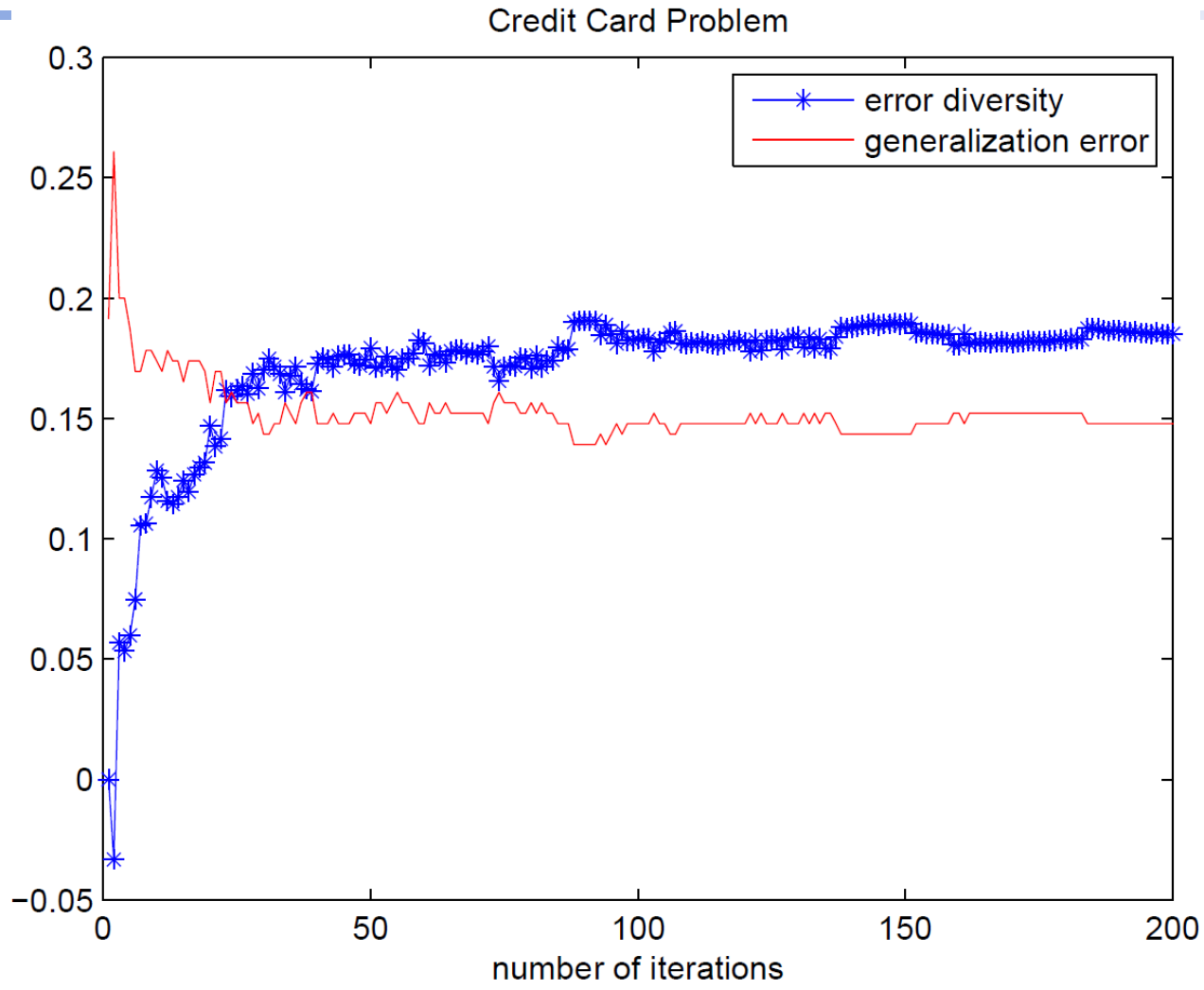
$$f(x) = \sum_{i=1}^{K} \alpha_i C_i(x)$$

  - We seek to minimize the exponential loss function:

$$\sum_{j=1}^{N} \exp\left(-y_j f(x_j)\right)$$

  - Not robust in noisy settings

Diversity in Boosting



It seems error diversity and generalization error are negatively correlated.

Random Forests (1)

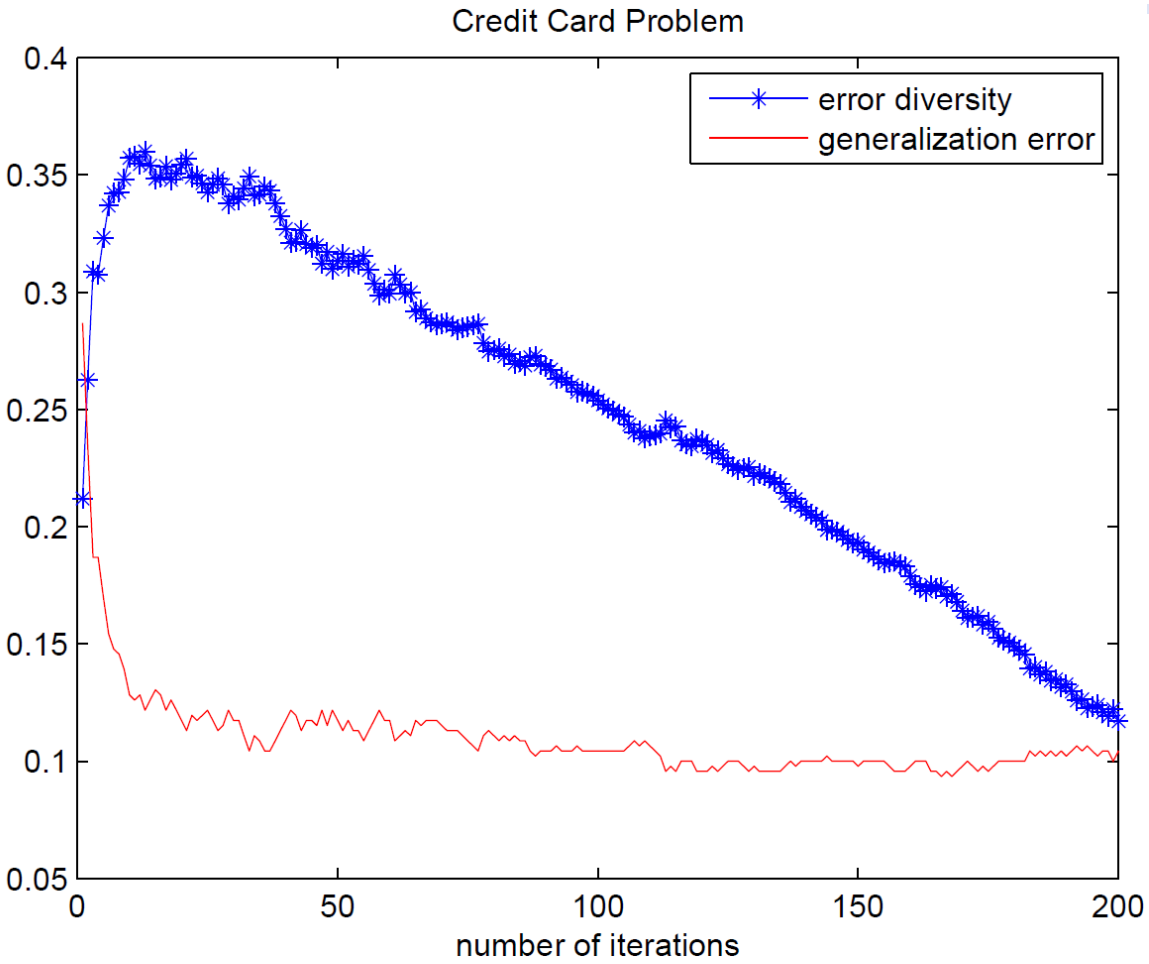
- Algorithm
  - Choose T — number of trees to grow
  - Choose $m<M$ (M is the number of total features) — number of features used to calculate the best split at each node (typically 20%)
  - For each tree
    - Choose a training set by choosing N times (N is the number of training examples) with replacement from the training set
    - For each node, randomly choose m features and calculate the best split
    - Fully grown and not pruned
  - Use majority voting among all the trees

[Breiman01]

Random Forests (2)

- Discussions
  - Bagging + random features
  - Improve accuracy
    - Incorporate more diversity and reduce variances
  - Improve efficiency
    - Searching among subsets of features is much faster than searching among the complete set

Diversity in Random Forests

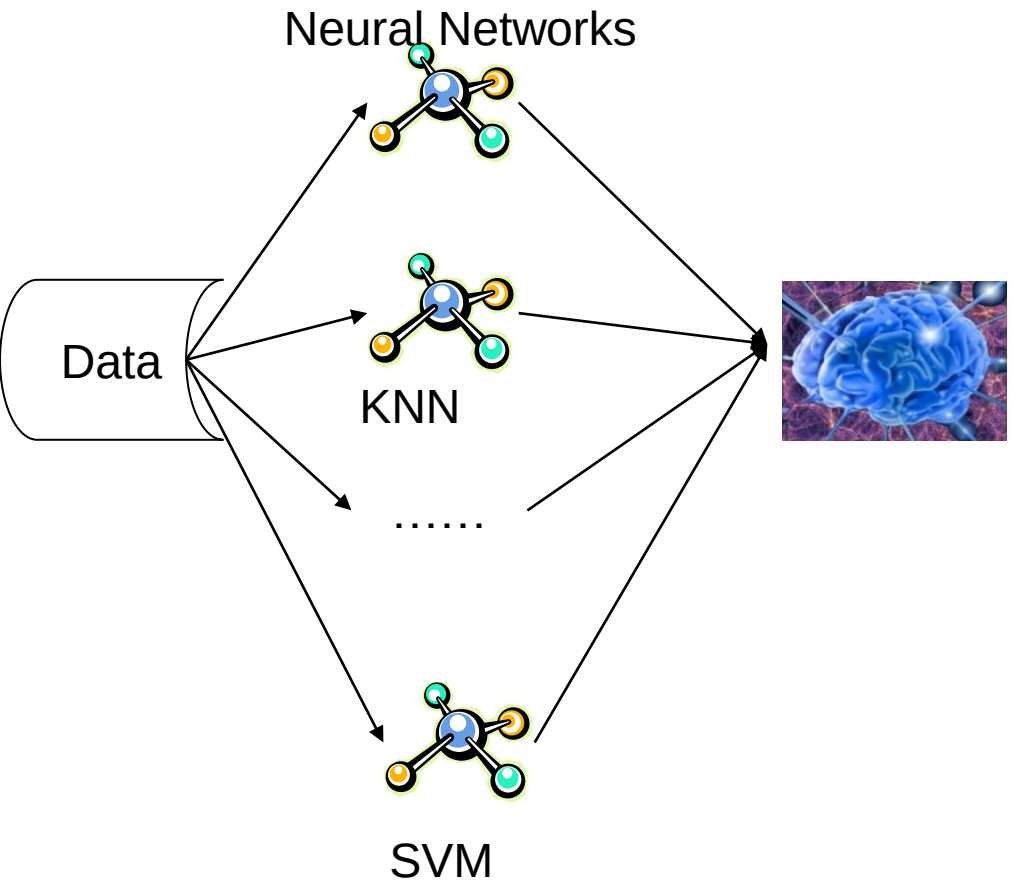


- Diversity is important in the beginning.

Diversity Generation Methods

- Diversity Encouragement by Data Manipulation
- Diversity Encouragement by Architectures Manipulation
- Diversity Encouragement by Hypothesis Space Traversal

Diversity Encouragement by Architectures Manipulation



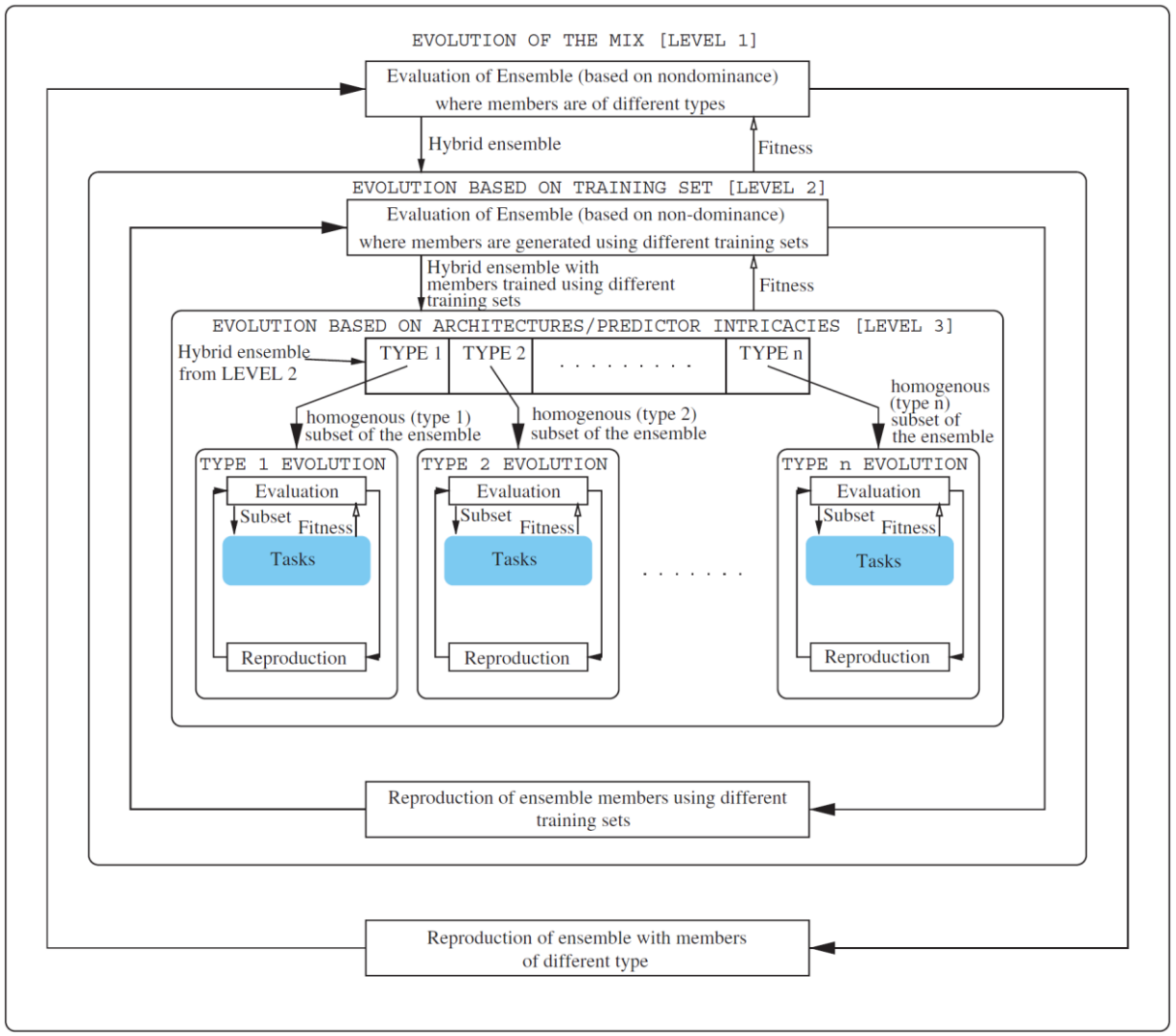
- Hybrid Ensemble
  - Different base learners: MLP, RBF, decision tress, KNN…
  - Kind of mixture of experts
  - Perform well but without solid foundations

[Islam03]

Architectures Manipulation

- Evolutionary neural networks approaches:
  - number of hidden nodes – least useful
  - Network topology
  - Cooperative training algorithm, negative correlation learning
  - Ensemble generation by combining final populations
  - Evolutionary neural networks methods: Cooperative neural-network ensembles (CNNE)

DIVerse and Accurate Ensemble Learning Algorithm (DIVACE)

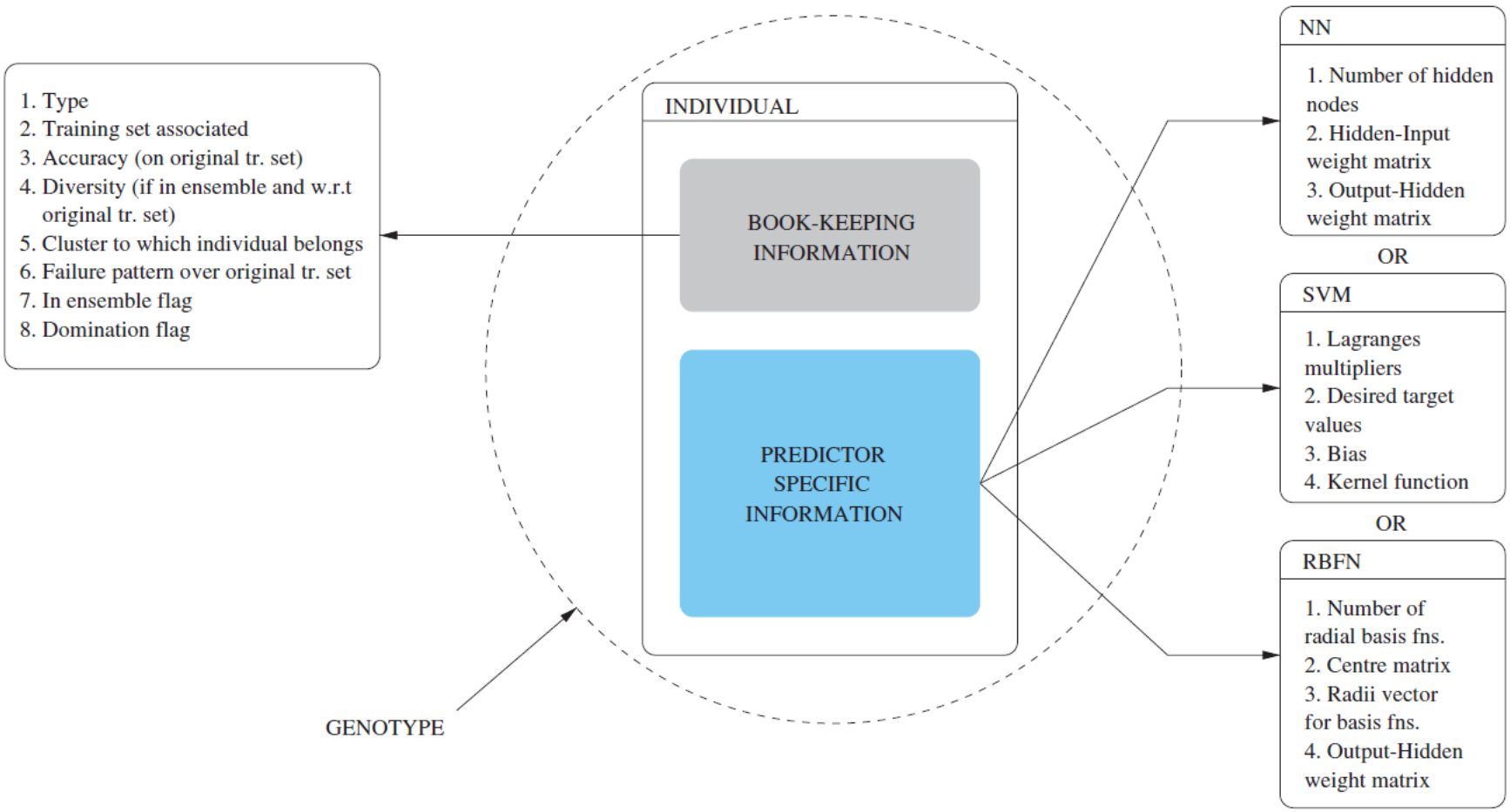


NOTE: Fitness consists of 2 values for each member in the ensemble/subset viz. Diversity and Accuracy

- Level 1: Methodological difference
- Level 2: Training set structure
- Level 3: Predictor intricacies

[Chandra06]

Genotypic Representation of an Individual



[Chandra06]

Diversity Generation Methods

- Diversity Encouragement by Data Manipulation
- Diversity Encouragement by Architectures Manipulation
- Diversity Encouragement by Hypothesis Space Traversal

Diversity Encouragement by Hypothesis Space Traversal

- Penalty methods by adding correlated terms

$$e_i = \frac{1}{2}(f_i - d)^2 + \lambda R$$

where $R = (f_i - f_{ens}) \sum_{j \neq i} (f_j - f_{ens}) = -(f_i - f_{ens})^2$ and $f_{ens} = \frac{1}{M} \sum_{i=1}^{i=M} f_i$

λ is a weigthing paramter on the penalty term R

- This is not a regularization term in the sense of Tikhonov regularization.
- Negative Correlation Learning = “Do not make error in the same place”

Outline

- An overview of ensemble methods
- Diversity generation methods
- Theoretical analysis of diversity
- Mange diversity in ensemble
  - Semi-supervised learning
  - Ensemble pruning
  - Multi-objective optimization
- Further topics and open discussions

Theoretical analysis of Ensemble and Diversity

- Error Decomposition for Regression and Classification Ensembles
- Bias, Variance Tradeoff
- Bias, Variance and Covariance Tradeoff

Regression Ensemble Error Decomposition

The Ambiguity Decomposition given (x, t) and

$$f_{ens} = \frac{1}{M} \sum_{i=1}^{i=M} f_i$$

$$(f_{ens} - t)^2 = \underbrace{\frac{1}{M} \sum_i (f_i - t)^2}_{\text{Average network error}} - \underbrace{\frac{1}{M} \sum_i (f_i - f_{ens})^2}_{\text{Average network ambiguity}}$$

"the error of the ensemble estimator is guaranteed to be less than or equal to the average network error"

[Krogh95]

Application

- Ambiguity has been used in many ways:
  - Pattern selection [Krogh95]
  - Feature selection [Opitz99]
  - Optimising topologies [Opitz96]
  - Optimising combination [Krogh95]
  - Optimising training time [Carney99]

  - **Optimising network weights (NC Learning)**

Diversity for Classification: Pair-wise diversity measures

- Q statistics [Yule1900]
  - The coefficient of association for two classifiers [-1,1]
- Kappa statistics [Dietterich00]
  - The chance-corrected proportional agreement
- Correlation coefficient [Sneath73]
  - The strength and direction of a linear relationship between two classifiers
- Disagreement measure [Ho98]
  - The ratio between the number of observations on which one classifier is correct and the other is incorrect to the total number of observations

Non-pairwise diversity measures

- Entropy measure [Cunningham00]
  - Measure the entropy measure
- Kohavi-Wolpert variance [Kohavi96]
  - Based on bias-variance decomposition of expected misclassification rate
- Measure of difficulty [Hansen90]
  - measure the distribution of difficulty and in order to capture the distribution shape
- Generalized diversity and Coincident failure diversity [Partridge97]
  - Based on the definitions of minimal diversity (randomly picked classifier failing) and maximal diversity (one incorrect, others correct)

Classification Ensemble Error Decomposition

- Given the definition of 0/1 classification error function

$$Err(x) = \begin{cases} 1 & if \quad x = -1 \\ 0.5 & if \quad x = 0 \\ 0 & if \quad x = 1 \end{cases}$$

and the definition of ensemble $f_{ens}(x_n) = sign\left(\sum_{i=1}^{i=M} c_i f_i(x_n)\right)$

The error decomposition:

$$Err\left(f_{ens}(x_n)\cdot y_n\right) = \sum_{i=1}^{M} c_i Err\left(f_i(x_n)\cdot y_n\right) - \frac{y_n}{2}\sum_{i=1}^{M}\left(\frac{1}{M} f_{ens}(x_n) - c_i f_i(x_n)\right)$$

weighted average error of the individuals.

difference between *ensemble and* individuals

Chen08:Thesis

Classification Diversity

- The right hand side can be written as:

$$\frac{y_n}{2}\sum_{i=1}^{M}\left(\frac{1}{M}f_{ens}(x_n)-c_i f_i(x_n)\right)=\frac{1-s}{2}y_n f_{ens}(x_n)$$

Margin of ensemble

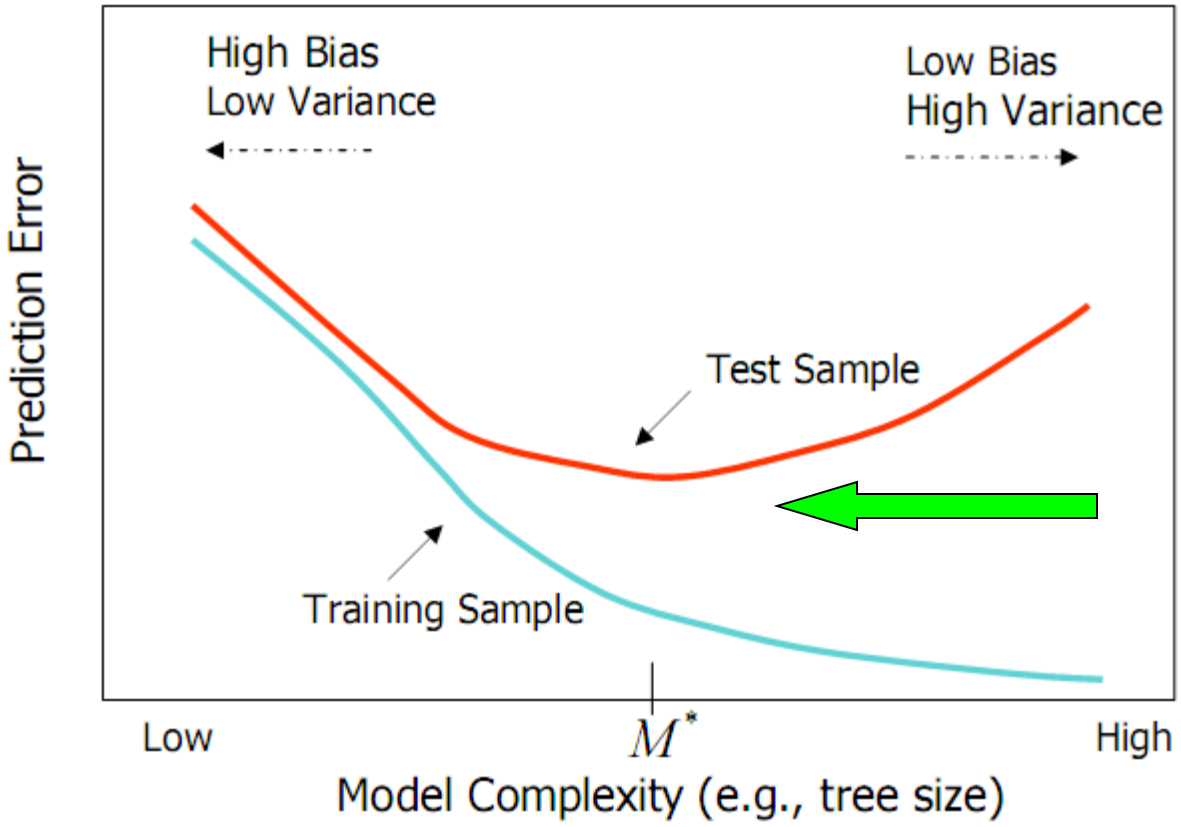
where $s_n=\left|\sum_{i=1}^{M}c_i f_i(x_n)\right|$ measures the difference between the number of positive and negative votes.

$$y_n f_{ens}\uparrow + s_n\downarrow \Rightarrow diversity\uparrow$$

Correct Decision Diverse members

Bias and Variance

- Ensemble methods
  - Combine learners to reduce variance



from Elder, John. From Trees to Forests and Rule Sets - A Unified Overview of Ensemble Methods. 2007.

Bias ,Variance and Covariance in Ensemble

$$E[(f-t)^2] = (E[f]-t)^2 + E[(f-E[f])^2]$$

$$MSE \quad = \quad bias^2 \quad + \quad var$$

and if *f* is a simple-average combination of other learning machines :

$$MSE \quad = \quad \overline{bias^2} \quad + \quad \left(\frac{1}{M}\right)\overline{var} \quad + \quad \left(1-\frac{1}{M}\right)\overline{covar}$$

Variance reduction

'Diversity' is a trade-off, just like the standard B-V decomposition

Bias ,Variance and Covariance in Ensemble

if f is a simple-average combination of other learning machines :

$$MSE = \overline{bias^2} + \left(\frac{1}{M}\right)\overline{var} + \left(1-\frac{1}{M}\right)\overline{covar}$$

Variance reduction

'Diversity' is a trade-off, just like the standard B-V decomposition

$$\overline{bias} = \left(\frac{1}{M}\sum_i (E_i\{f_i\} - y)\right)^2, \overline{\text{var}} = \frac{1}{M}\sum_i E_i\left\{(y - E_i\{f_i\})^2\right\}$$

$$\overline{\text{covar}} = \frac{1}{M(M-1)}\sum_i\sum_{j\neq i} E_{i,j}\left\{(f_i - E_i\{f_i\})(f_j - E_j\{f_j\})\right\}$$

- This *decomposition provides the theoretical grounding of negative cor*relation learning which takes amount of correlation together with the empirical error in training neural networks.

Outline

- An overview of ensemble methods
- Diversity generation methods
- Theoretical analysis of diversity
- Mange diversity in ensemble
  - Supervised learning
  - Semi-supervised learning
  - Ensemble pruning
  - Multi-objective optimization
- Further topics and open discussions

Manage Diversity in Ensemble

- Negatively Correlated Ensembles for Diversity Management
- Various Training Algorithms
- Regularized Negatively Correlated Ensembles with Bayesian Inference

Negative Correlation Learning

NC learning

$$e_i = \frac{1}{2}(f_i - d)^2 - \lambda(f_i - \bar{f})^2$$

We know this relationship…

Ambiguity decomposition

$$(\bar{f} - d)^2 = \frac{1}{M}\sum_i (f_i - d)^2 - \frac{1}{M}\sum_i (f_i - \bar{f})^2$$

…and this one can be shown with a few calculations (Brown, 2004)

Diversity trade-off

$$MSE = \overline{bias^2} + \left(\frac{1}{M}\right)\overline{var} + \left(1 - \frac{1}{M}\right)\overline{covar}$$

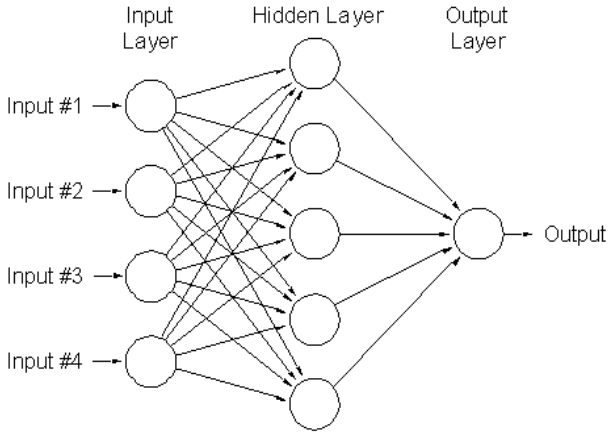
Therefore, unlike other algorithms before it…

NC learning explicitly manages the regression diversity trade-off.

Negatively Correlated Ensembles

- Bagging, Random forests, often train ensemble members independently.
- Negative correlation learning (NCL) aims to negatively correlate errors of ensemble member. Make it cooperative.
- NCL uses a number of neural networks as ensemble members.
- Given the data set $\{x_n, y_n\}_{n=1}^{N}$, the training function of a neural network is listed as follows:

$$\min \sum_{n=1}^{N} E_n = \sum_{n=1}^{N} (f(x_n) - y_n)^2$$



Formulation of NCL

- The training of NCL is implemented by minimized the following error function for every network

$$e_i = \frac{1}{2}(f_i - y_n)^2 + \frac{1}{2}\lambda p_i \quad \text{where} \quad p_i = (f_i - f_{ens})\sum_{j \neq i}(f_j - f_{ens}) = -(f_i - f_{ens})^2$$

$$e_i = \frac{1}{2}(f_i - y_n)^2 - \frac{1}{2}\lambda(f_i - f_{ens})^2$$

Where i=1,2,…,M and $f_{ens} = \frac{1}{M}\sum_{i=1}^{i=M} f_i$

The training minimizes the training error, while simultaneously makes it as different from the rest of ensemble members as possible.

Sum these terms, we could get the cost function for ensemble

$$E_n = \sum_{i=1}^{M} e_i = \frac{1}{2}\sum(f_i - y_n)^2 - \frac{1}{2}\lambda\sum(f_i - f_{ens})^2 \quad \text{When} \quad \lambda = 1$$

$$E_n = \sum_{i=1}^{M} e_i = \frac{1}{2}\sum(f_i - y_n)^2 - \frac{1}{2}\sum(f_i - f_{ens})^2 = \frac{1}{2}M(f_{ens} - y_n)^2$$

This is to minimize the training error of the whole ensemble.

Over fitting of Neural Network

- This formulation that only minimizes the training error might lead to overfitting

$$\min \sum_{n=1}^{N} E_n = \sum_{n=1}^{N} (f(x_n) - y_n)^2$$

- Weight Decay or some other model selection techniques are used to control the complexity of neural networks, such as

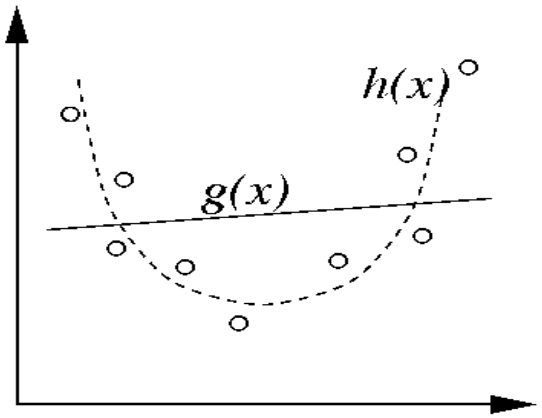
$$\min \sum_{n=1}^{N} E_n = \sum_{n=1}^{N} (f(x_n) - y_n)^2 + \alpha \sum_{i} w_i^2$$

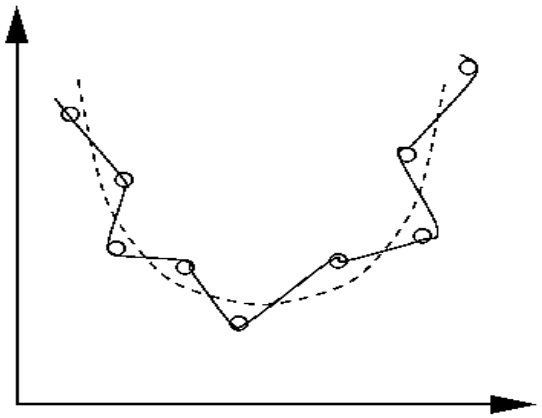
- Recall the formulation of NCL, when $\lambda = 1$, is

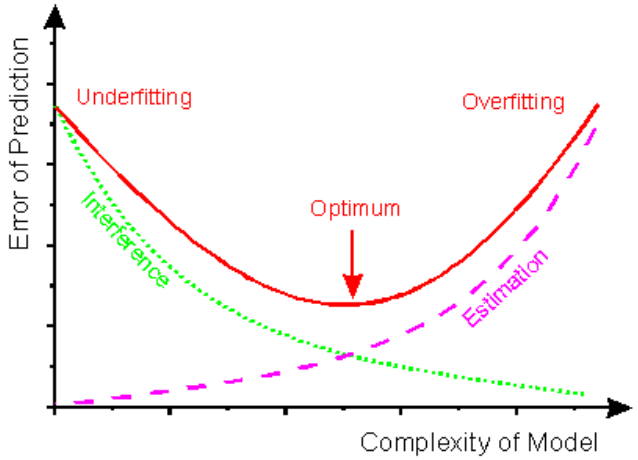
$$\min \sum_{n=1}^{N} E_n = \sum_{n=1}^{N} (f_{ens} - y_n)^2$$

Without regularization, this formulation might lead to overfitting.

Under fitting and over fitting







Large α value
Under fitting

Small α value
Over fitting

Relationship between under fitting and overfitting.

The parameter of α is very important to the system.

Improper one will lead to either underfitting or overfitting.

Bayesian methods can be used to find the “optimal” α value under Bayesian framework.

Regularized Negative Correlation Learning

Control the complexity to avoid overfitting of NCL

$$E_{ens} = \sum_{n=1}^{N}(f_{ens}(x_n) - y_n)^2 + \sum_{i=1}^{M}\alpha_i w_i^T w_i$$

The error function for network i can be obtained as follows:

$$e_i = \sum_{n=1}^{N}(f_i(x_n) - y_n)^2 - \sum_{n=1}^{N}(f_{ens}(x_n) - f_i(x_n))^2 + \alpha_i w_i^T w_i$$

Bayesian Inference for Parameters(1)

- Assume targets are sampled from the model with additive/independent noise $e_n \sim N(0, \beta^{-1})$

$$y_n = f_{ens}(\mathbf{x}_n) + e_n = \frac{1}{M}\sum_{i=1}^{M} f_i(\mathbf{x}_n) + e_n,$$

- According to the Bayesian theorem,

$$P(w \mid D) = \frac{P(D \mid w, \beta)P(w \mid \mu)}{P(D \mid \mu, \beta)}$$

where w is the weight vector, μ and β are hyperparameters.

Bayesian Inference for Parameters(2)

- Assume weight vector of each network w_i is a Gaussian distribution with mean zero and variance μ_i^{-1}.

- The prior of the weight vector w is obtained as follows

$$P(w \mid \mu) = \prod_{n=1}^{N} \left(\frac{\beta}{2\pi} \right)^{1/2} \exp\left(-\frac{\beta}{2} e_n^2 \right)$$

Bayesian Inference for Parameters(3)

- Omit all the constants and the normalization factor, and apply Bayesian theorem:

$$P(w \mid D) = \exp\left(-\frac{\beta}{2}\sum_{n=1}^{N} e_n^2\right)\exp\left(-\sum_{n=1}^{N}\frac{\mu_i}{2} w_i^T w_i\right)$$

- Taking the negative logarithm, the maximum of the posterior w is obtained as

$$\min J_1(w) = \frac{\beta}{2}\sum_{n=1}^{N} e_n^2 + \sum_{n=1}^{N}\frac{\mu_i}{2} w_i^T w_i$$

Empirical training error + Regularization term

Bayesian Inference for Parameters (4)

- According to Bayesian rule, the posteriors of α and β are obtained by

$$P(\mu,\beta \mid D) = \frac{P(D \mid \mu,\beta)P(\mu,\beta)}{P(D)} \propto P(D \mid \mu,\beta)$$

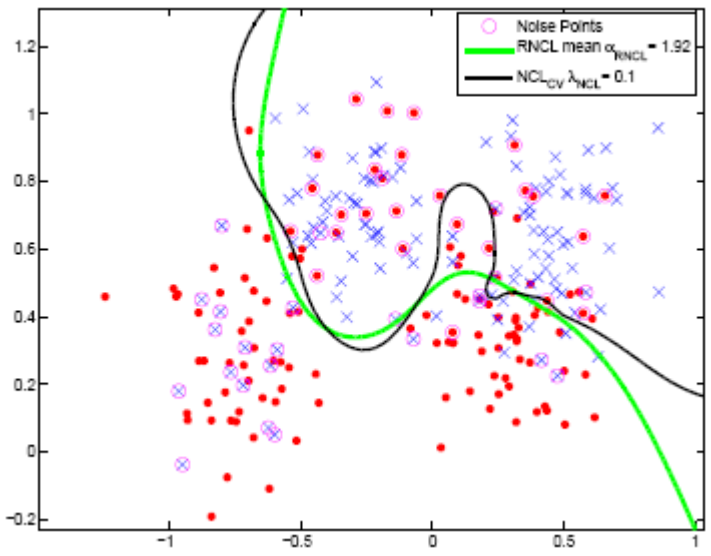
where a flat prior is assumed on the hyperparameters.

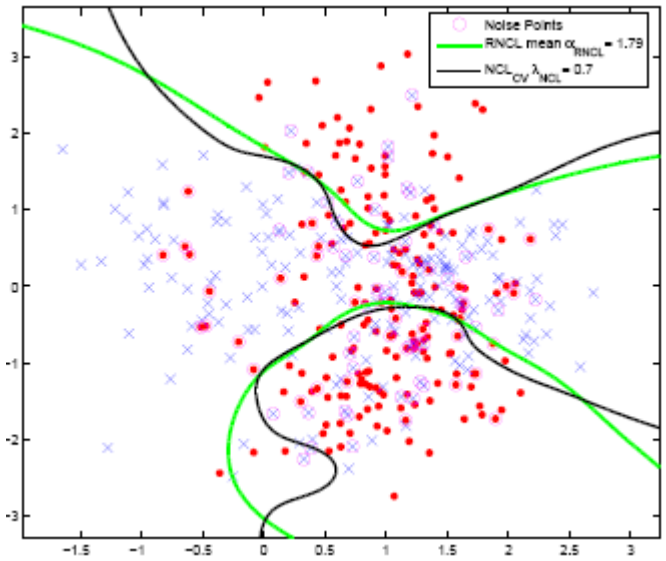
$$P(D \mid \mu,\beta) = \frac{P(D \mid w,\beta)P(w \mid \mu)}{P(w \mid D)}$$

$$\alpha_i^{new} = \frac{\sum_{n=1}^{N} e_{n,MP}(n_i - \sum_{j \in n_i} \frac{\alpha_i}{\lambda_j + \alpha_i})}{w_{i,MP}^T w_{i,MP}(N - \sum_{j=1}^{W} \frac{\alpha_i}{\lambda_j + \alpha_i})}$$

NCL: with and without Regularization (1)

- Two synthetic data sets are used with randomly generated noisy points, denoted with circle points.

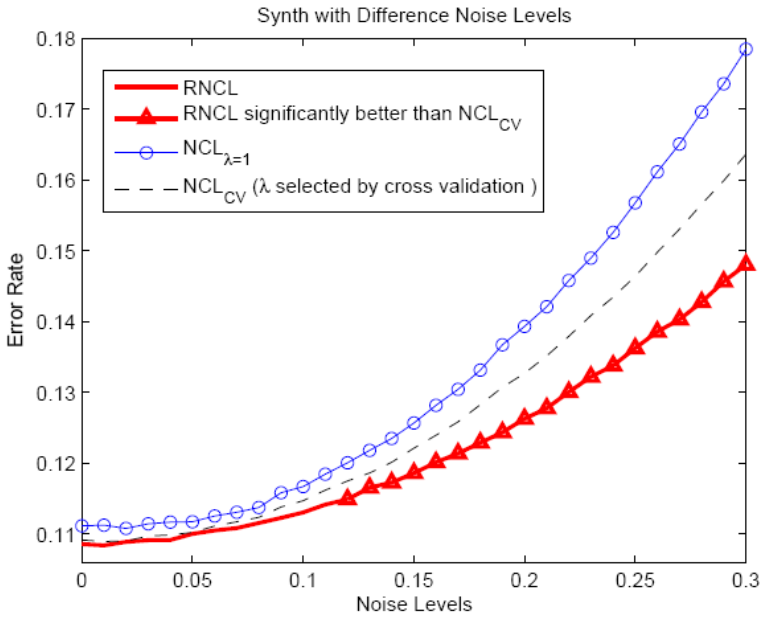


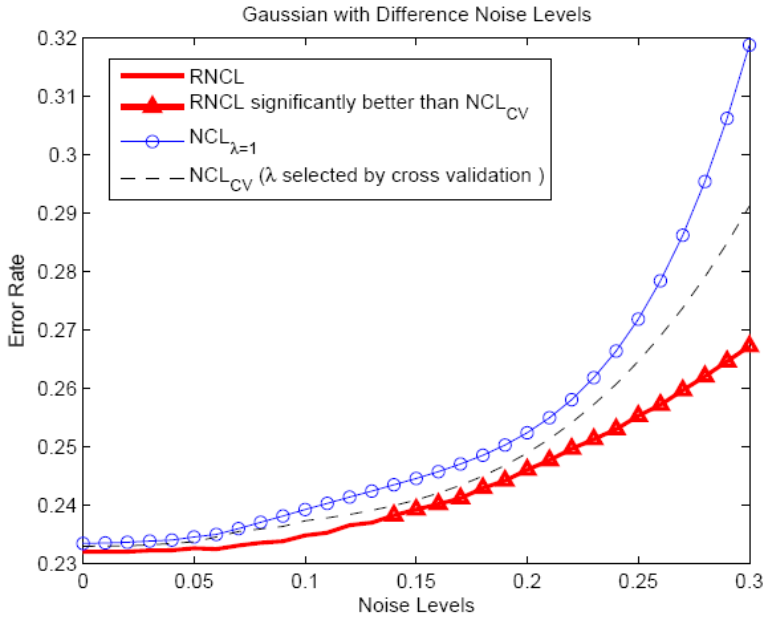


LEFT: *Synth* Problem. Green (RNCL), Black (NCL)

RIGHT: *Gaussian* Problem. Green (RNCL), Black (NCL)

NCL: with and without Regularization (2)





- Classification error of RNCL (solid), NCL (circled), and NCL (dashed) versus noise levels.
- A statistical t-test (95% significance level) is conducted to compare RNCL with NCL
- The triangles represent those points where RNCL significantly outperforms NCL

RNCL are more robust against noise.

Diversity Management with Semi-supervised Learning

- Ensemble Methods for Semi-supervised Problems

- Diversity Encouragement in both Labelled and Unlabeled Space

Semi-supervised NC Ensemble

- Given labelled set $D_l = \{(x_1, y_1), \cdots, (x_N, y_N)\}$ and unlabelled set $D_u = \{x_{N+1}, \cdots, x_{N+V}\}$
- The training error is:

$$e_i^{semi} = \frac{1}{N}\sum_{n=1}^{N}(f_i(x_n) - y_n)^2 - \frac{\lambda}{N+V}\sum_{n=1}^{N+V}(f_i(x_n) - f_{ens}(x_n))^2 + \frac{\alpha}{N}\sum_{j=1}^{h_i} w_{i,j}^2$$

Labelled training error

Labelled and unlabelled correlation term

Regularization

Bounds on the Penalty Coefficients

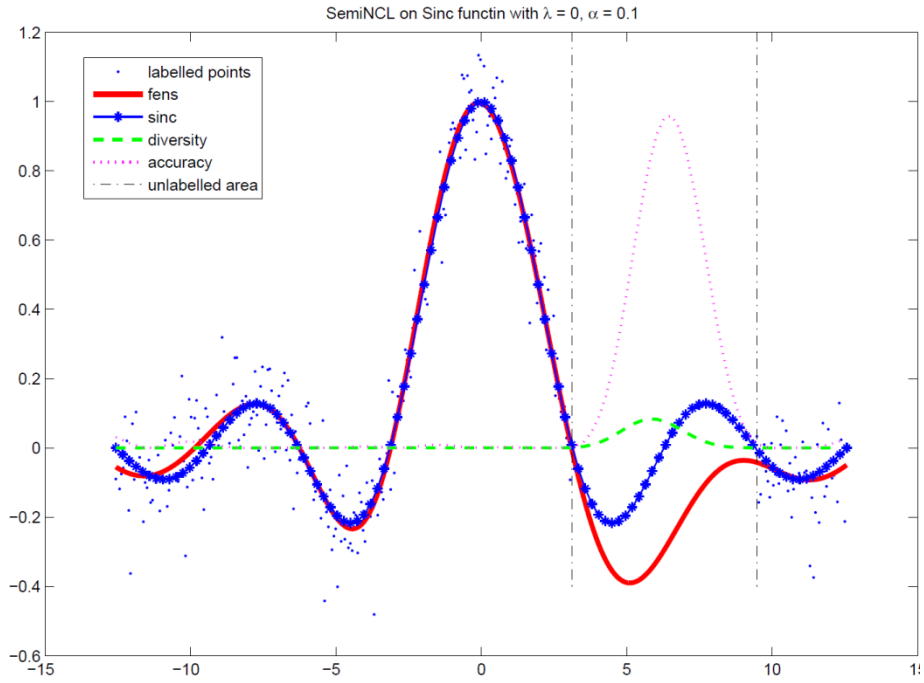
- Hessian matrix non-positive definite

$$\frac{\partial^2 e_i}{\partial^2 w_{i,j}^2} > 0 \Rightarrow \lambda < \left(\frac{c_1}{c_2} + \frac{\alpha}{N c_2} \right) \left(\frac{M}{M-1} \right)^2$$

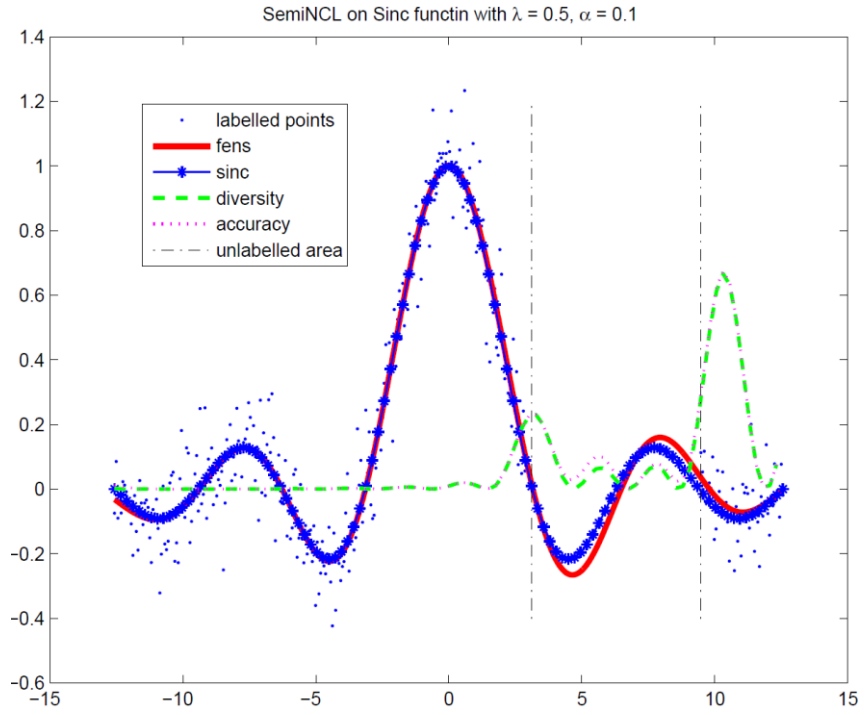
$$c_1 = \frac{1}{N} \sum_{n=1}^{N} \phi_{i,j}^2(x_n), \quad c_2 = \frac{1}{N+V} \sum_{n=1}^{N+V} \phi_{i,j}^2(x_n)$$

- RBF networks $f_i(x_n) = \sum_{k=1}^{K} w_{ki} \phi_{ki}(x_n)$
- Negative λ ⇨ positively correlated ensemble
- large positive λ ⇨ non-positive definite Hessian.

SemiNCL for SinC (1)

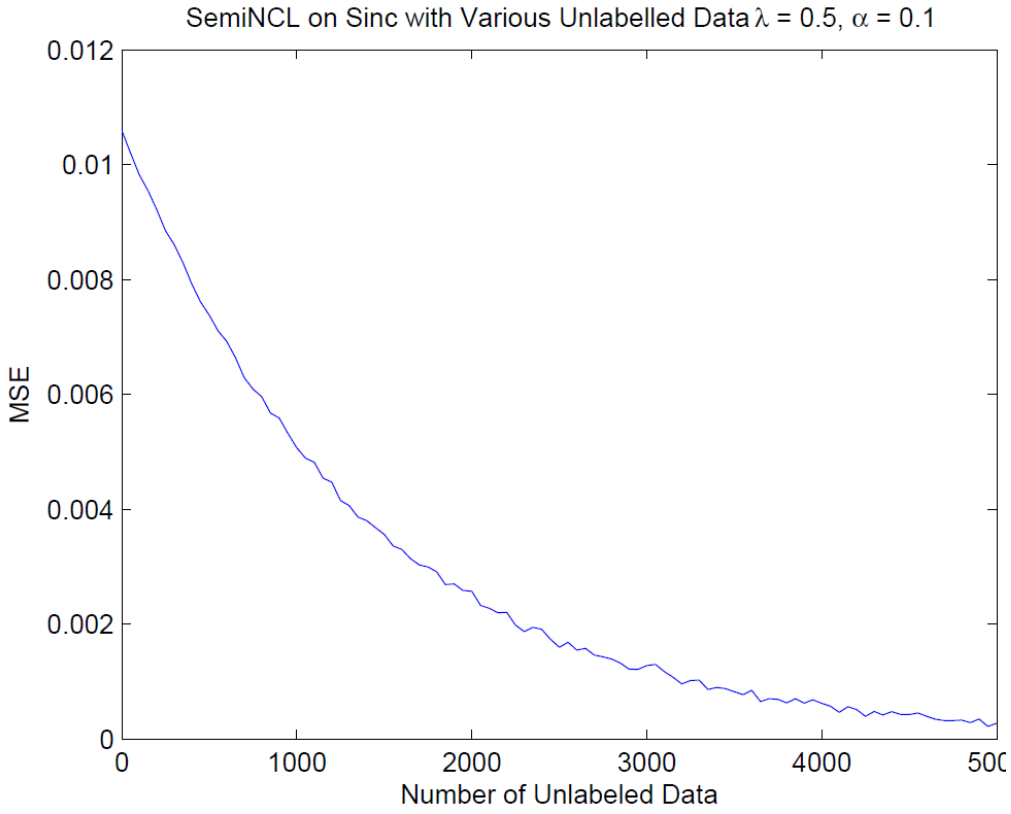


SemiNCL without unlabelled data



SemiNCL with 2000 unlabelled points

SemiNCL for SinC (2)



SemiNCL with Different Unlabelled Data

- SemiNCL achieves better performance by encouraging diversity,
- compensates the increment of accuracy, in both labelled and unlabelled areas.

Implementations

- Gradient Descent
  - Time consuming
  - Optimize RBF centers with widths
  - Better performance with more time
- Matrix Inversion
  - No local optimization
  - Fast for some problems
  - Not scalable
- Distributed SemiNCL
  - Fast implementation
  - Approximate the cost function

Distributed Semi NCL

Given: the final number of predictors M, a labeled training set $D_l = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)\}$ with N labeled examples and an unlabelled sample set $D_u = \{\mathbf{x}_{N+1}, , \mathbf{x}_{N+V}\}$ with V unlabeled points, the parameters λ and α.

Begin: Construct matrix $\mathbf{L}_i$, $\mathbf{G}_i$ and $\mathbf{U}_i$ $(i = 1, \cdots, M)$ as defined in Section III.D. The predictions of $\mathbf{f}_i^l$ and $\mathbf{f}_i^u$ are initialled to zero vectors. Initialize parameters λ, α, and $t = N/(N+V)$.

Repeat:

- For each ensemble member from $i = 1, 2, \cdots, M$ do:

$$\begin{aligned} \mathbf{w}_i &= \mathbf{G}_i^{-1}\left(\mathbf{L}_i\mathbf{y} - \frac{\lambda t}{M}\sum_{j=1, j\neq i}^{M}\left(\mathbf{L}_i\mathbf{f}_j^l + \mathbf{U}_i\mathbf{f}_j^u\right)\right). \\ \mathbf{f}_i^l &= \mathbf{L}_i^T\mathbf{w}_i \text{ and } \mathbf{f}_i^u = \mathbf{U}_i^T\mathbf{w}_i \\ & \quad \text{share } \mathbf{f}_i^l \text{ and } \mathbf{f}_i^u \end{aligned}$$

- Repeat for a desired number of iterations or until convergence.

Diversity with Ensemble Pruning

- Selection based Ensemble Pruning
- Weight based Ensemble Pruning
- Empirical, Greedy and Probabilistic Ensemble Pruning Methods

Selection based Ensemble Pruning

- Classifier ranking and pick the best ones
- KL-divergence pruning
- Kappa pruning
- Kappa-error convex hull pruning
- Back-fitting pruning
- maximizing the pair-wise difference between the selected ensemble members
- several pruning algorithms for their distributed data mining systemHeuristics

[Chawla04]

Weight based Ensemble Pruning (1)

- The optimal combination weights minimizing the mean square error (MSE) can be calculated analytically

$$w_i = \frac{\sum_{j=1}^{M} (C^{-1})_{ij}}{\sum_{k=1}^{M} \sum_{j=1}^{M} (C^{-1})_{kj}}$$

where C is the correlation matrix with elements indexed as

$$C_{ij} = \int p(x)(f_i(x) - y)(f_j(x) - y)dx$$

$$C_{ij} \approx \frac{1}{N} \sum_{n=1}^{N} \left[(y_n - f_i(x_n))(y_n - f_j(x_n)) \right]$$

- This approach rarely works well in real-world applications.
- ill-conditioned matrix C due to the fact that there are often some estimators that are quite similar in performance

Weight based Ensemble Pruning (2)

- Least square (LS) pruning
  - Minimize the ensemble MSE
  - Lead to negative combination weights
  - negative combination weights are unreliable [Ueda00]
- Genetic algorithm pruning [Yao98] [Zhou02]
  - weigh the ensemble members by constraining the weighs to be positive
  - sensitive to noise.
- Mathematical Programming [Zhang06]
  - mathematical programming to look for good weighting schemes.
- Bayesian Automatic Relevance Determination (ARD) Pruning [Tipping01]
  - Sparse prior for sparse ensembles

ARD Pruning

- weighted-based ensemble pruning can be viewed as a sparse Bayesian learning problem by applying Tipping's relevance vector machine

Automatic Relevance Determination (ARD)

- Give the classifier weight independent Gaussian priors whose variance, $\boldsymbol{\alpha}^{-1}$, controls how far away from zero each weight is allowed to go:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_i \mathcal{N}(w_i|0, \alpha_i^{-1}),$$

- Maximize $p(\mathbf{t}|\boldsymbol{\alpha})$ the marginal likelihood of the model, with respect to $\boldsymbol{\alpha}$.
- Outcome: many elements of $\boldsymbol{\alpha}$ go to infinity, which naturally prunes irrelevant features in the data.

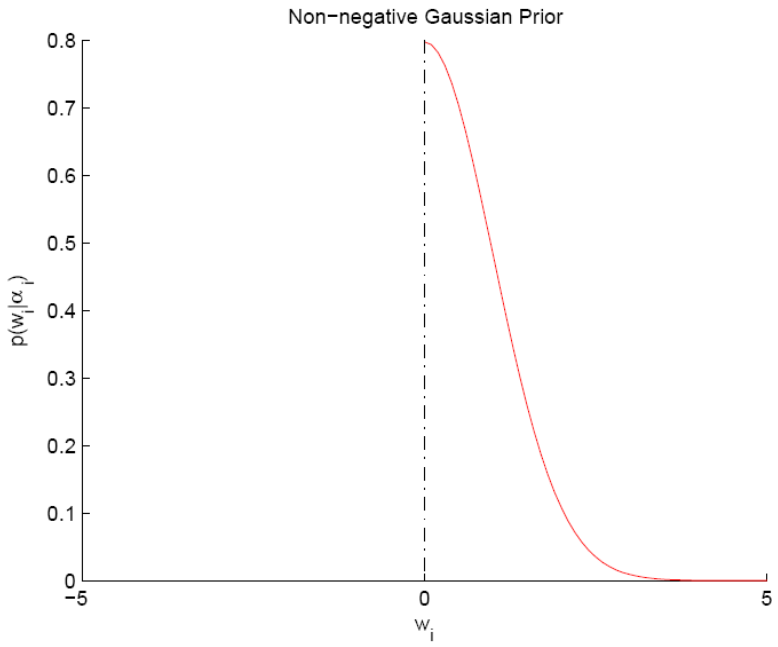
Ensemble Selection by Expectation Propagation

- Small ensembles can be better than large ensembles in terms of accuracy and computational complexity [Zhou02]

$$f_{ens}(\mathbf{x};\mathbf{w}) = \sum_{i=1}^{M} w_i f_i(\mathbf{x}) = \mathbf{w}^T \mathbf{F}(\mathbf{x}),$$

- Adopt truncated Gaussian prior for weight vectors to prune some ensemble members

Truncated Gaussian Prior



- Negative weight vectors are neither intuitive nor reliable.
- Bayesian inference is intractable with truncated prior

Ensemble Selection by Expectation Propagation

- Expectation propagation (EP) a integral approximating algorithm.

- EP adopts a family of Gaussian functions to approximate each term by minimizing the KL-divergence between the exact term and the approximation term, and then combines these approximations analytically to obtain a Gaussian posterior.

$$p(D, \mathbf{w}) = p(\mathbf{w}) \prod_{n=1}^{N} p(y_n|\mathbf{w}, x_n) = \prod_{i=1}^{M} p(w_i) \prod_{n=1}^{N} p(y_n|\mathbf{w}, x_n) = \prod_{i=1}^{M} t_i \prod_{n=1}^{N} g_n,$$

EP for Ensemble Selection

- Prior $$p(w_i|\alpha_i) = \begin{cases} 2N(w_i|0,\alpha_i^{-1}) & \text{if } w_i \geq 0 \\ 0 & \text{if } w_i < 0 \end{cases}$$

- Likelihood $$p(y|\mathbf{x},\mathbf{w}) = \prod_{n=1}^{N} p(y_n|\mathbf{x}_n,\mathbf{w}) = \prod_{n=1}^{N} \Phi\left(y_n \sum_{i=1}^{M} w_i f_i(\mathbf{x}_n)\right)$$

Where $\Phi(x) = \int_{-\infty}^{x} N(t|0,1)dt$ is the Gaussian cdf.

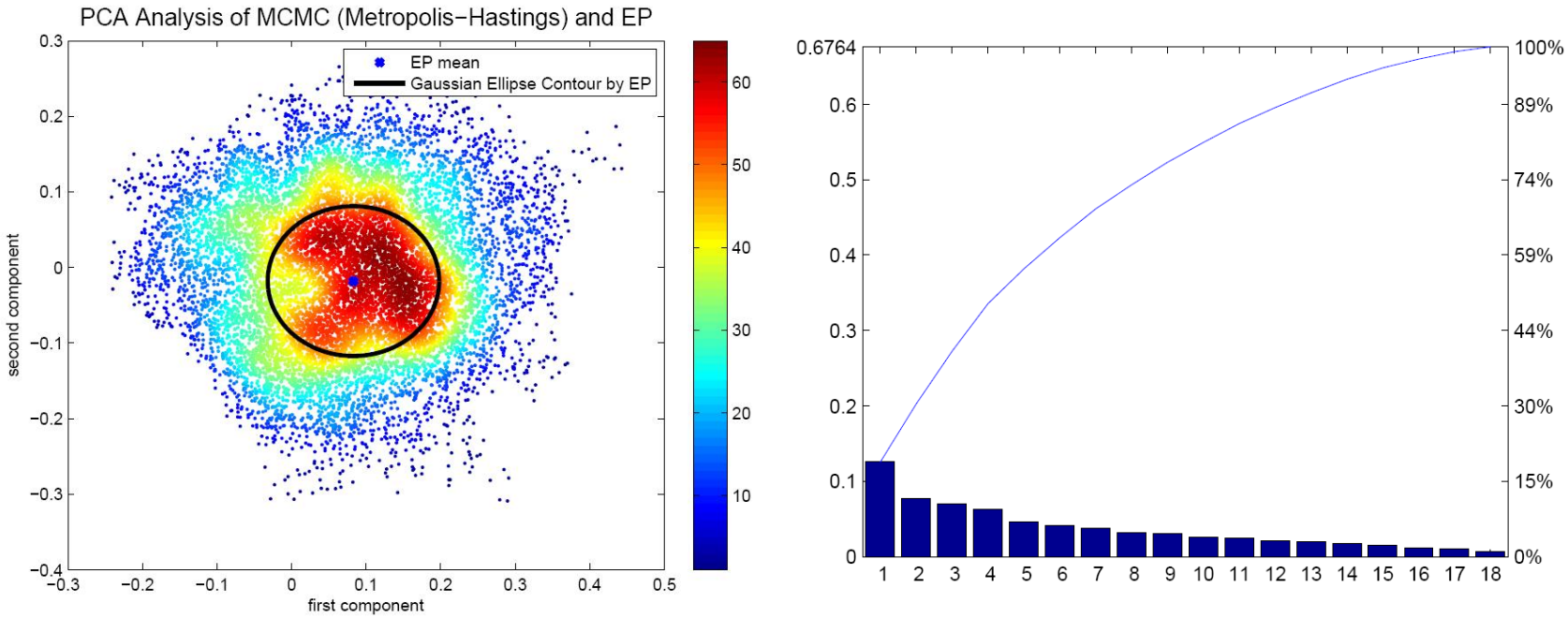
- Posterior $$p(\mathbf{w}|\mathbf{x},y,\alpha) \propto \prod_{i=1}^{M} p(w_i|\alpha_i) \prod_{n=1}^{N} p(y_n|\mathbf{x}_n,\mathbf{w})$$

Hyperparameters Optimization

- Update the hyperparameter alpha based on the type-II marginal likelihood, also known as the evidence.

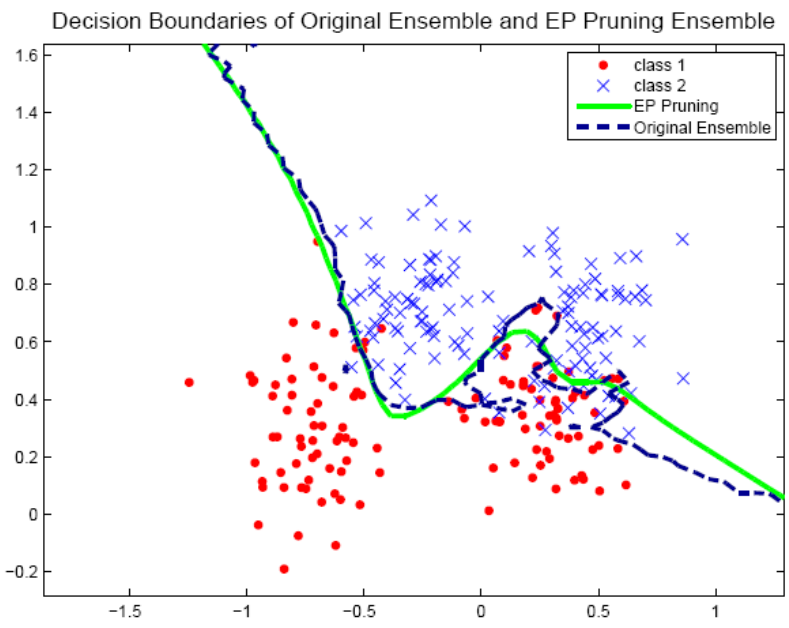
- According to the updated value, we choose to add one learner to the ensemble, delete one ensemble member or re-estimate the hyperparameter alpha.

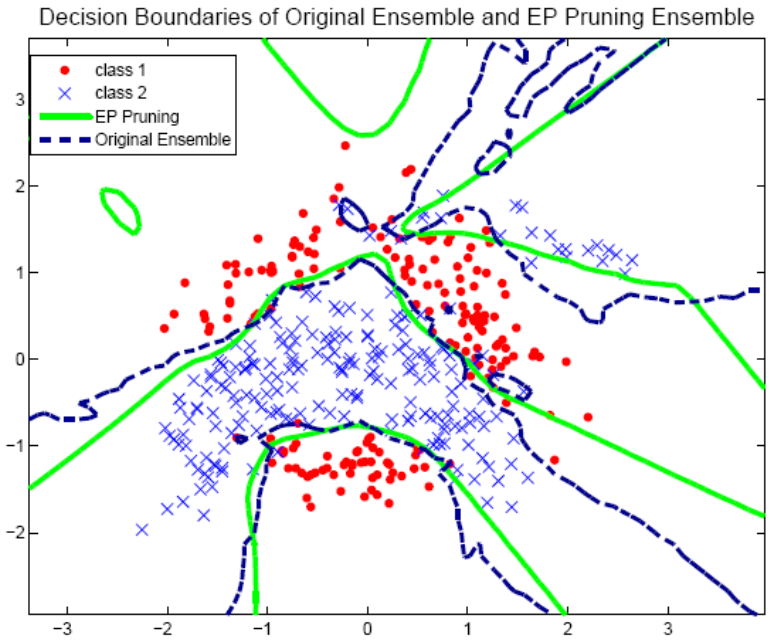
Expectation Propagation and MCMC



- The posteriors of combination weights calculated by MCMC (30000 sampling points) and EP. The color bar indicates the density (the number of overlapping points) in each place.

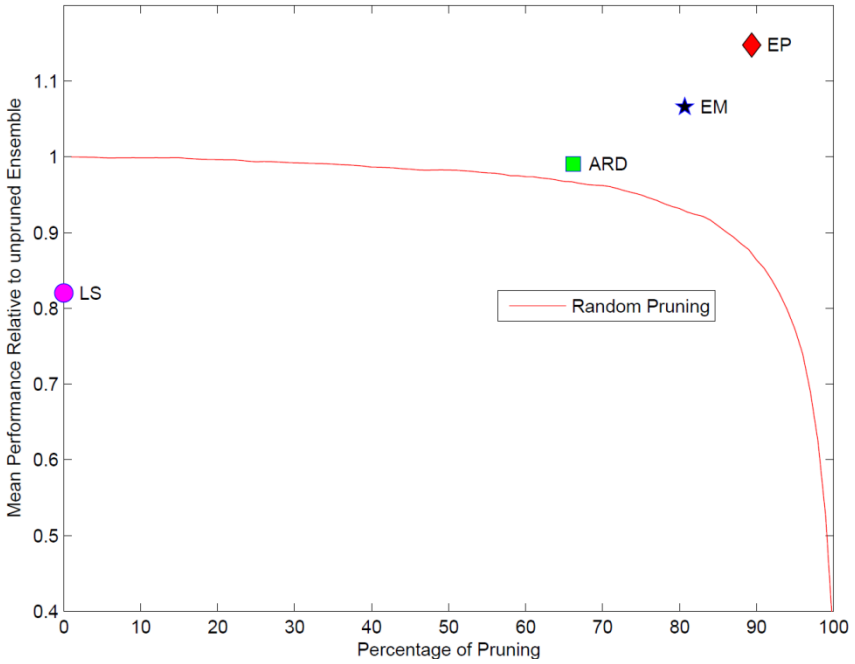
Decision Boundary by EP



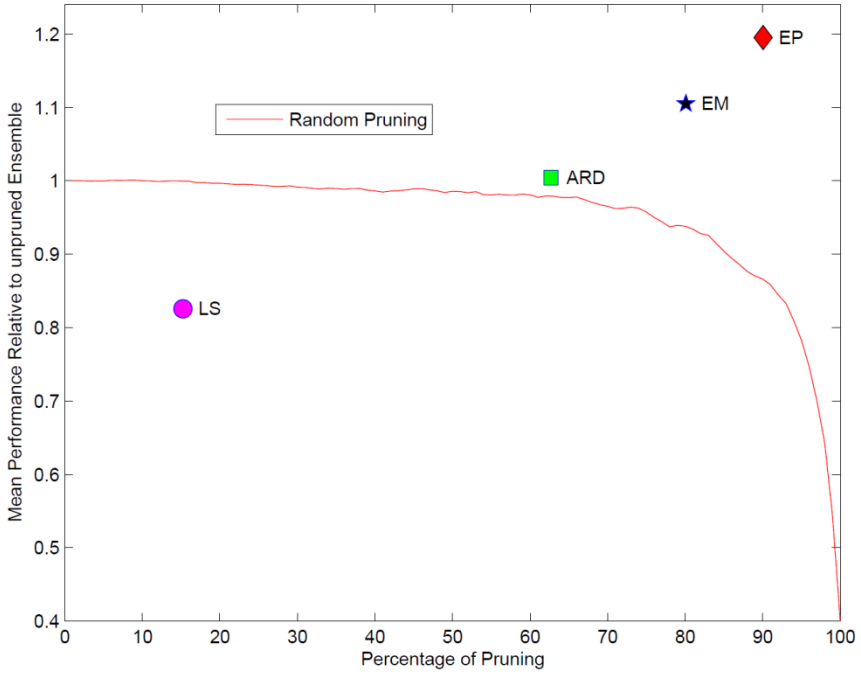


- Comparison of EP-pruned ensembles and un-pruned Adaboost ensembles on Synth and banana data sets. The Adaboost ensemble consists of 100 neural networks with random selected hidden nodes (3-6 nodes).

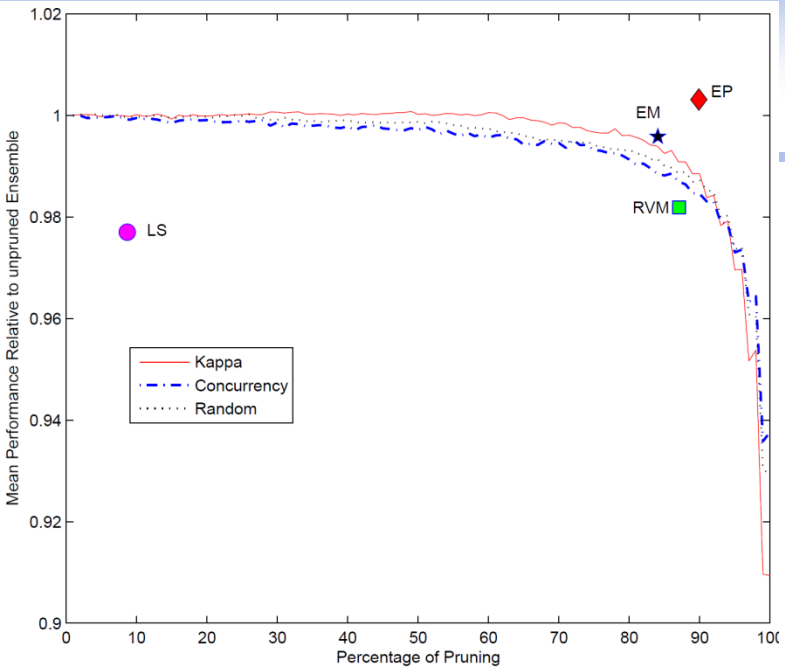
Regression



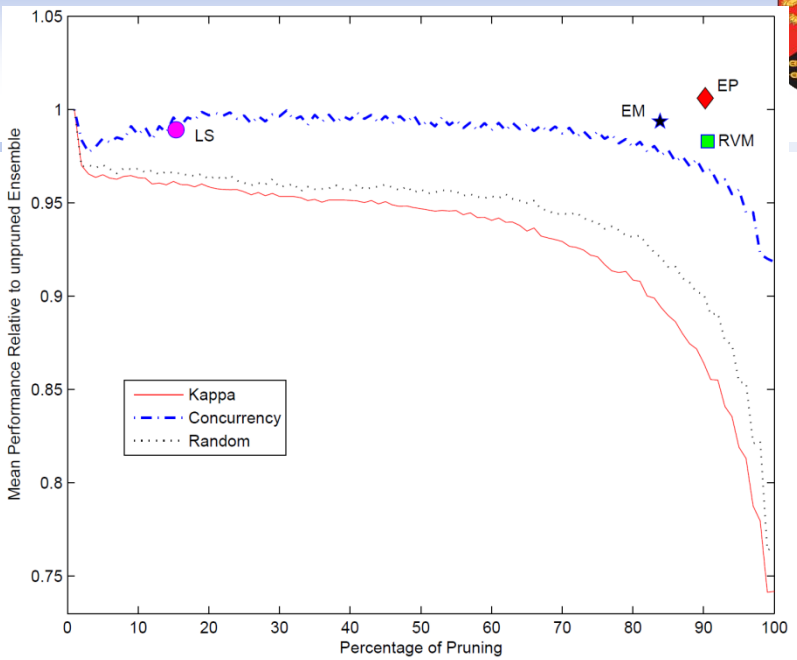
Bagging



Random Forest

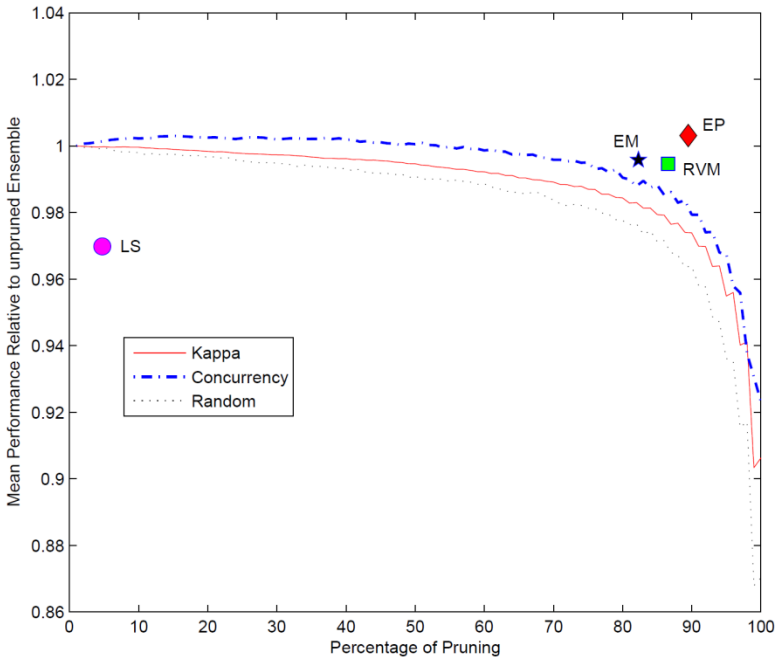


Bagging



Adaboost

Classification Problems



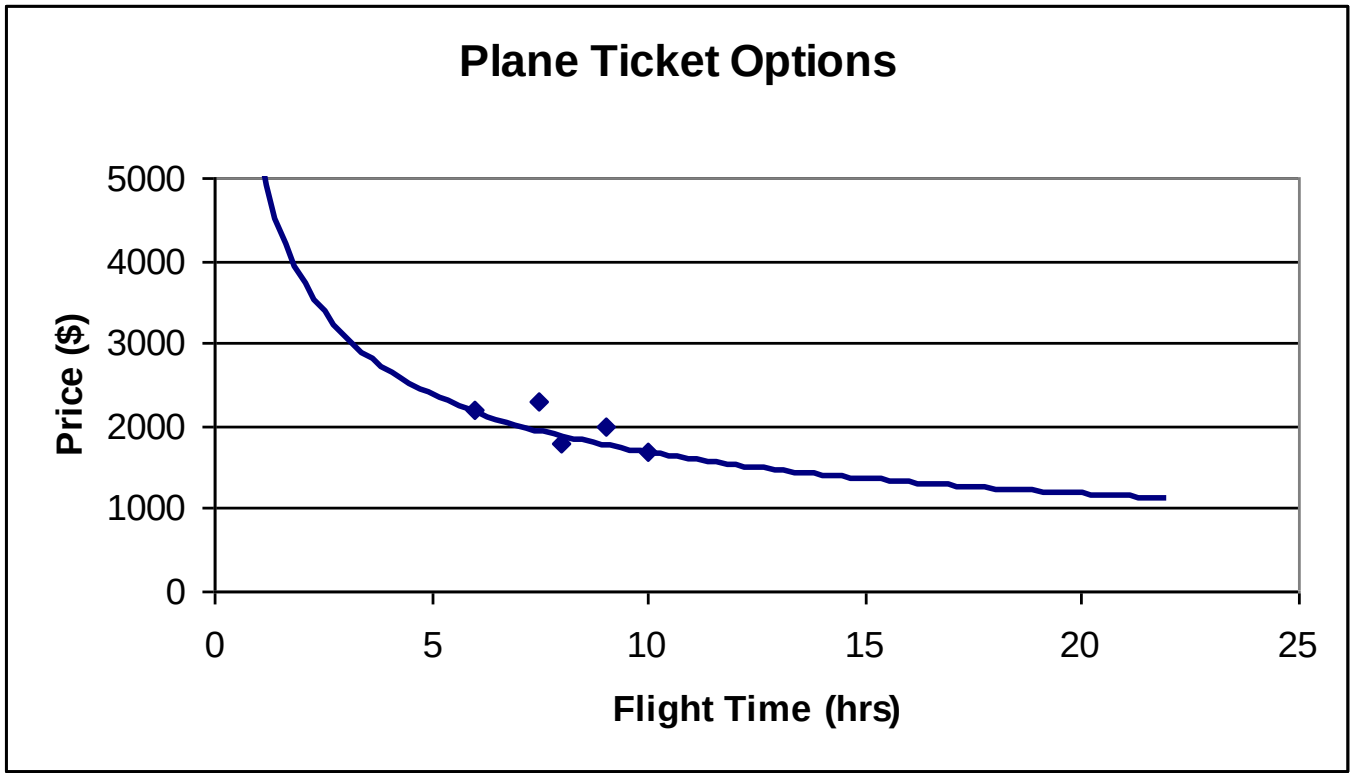
Random Forest

Diversity Management via Multi-objective Optimization

- Diversity, Accuracy, Regularization with Generalization
- Multi-objective Optimization to optimize the trade-off

Multi-objective Optimization (MOO)

- Multi-objective optimization is the optimization of conflicting objectives.



Difference Between Optimization and Learning

- Optimization: optimal solutions
- Learning: generalized solutions
- Generalized solutions require proper model formulations, e.g. regularization to control model complexity

Ensemble with MOO

- Objective of Performance $\sum_{n=1}^{N}(f_i(\mathbf{x}_n) - y_n)^2$

  This objective measures the empirical mean square error based on the training set.

- Objective of Correlation $-\sum_{n=1}^{N}(f_i(\mathbf{x}_n) - f_{ens}(\mathbf{x}_n))^2$

  This objective measures the amount of variability among the ensemble member and this term can also be treated as the diversity measure

- Objective of Regularization $\mathbf{w}_i^T\mathbf{w}_i = \sum_j w_j^2$

  This objective causes the weights to converge to smaller absolute values than they otherwise would.

Evolutionary Neural Network Operators

- **Component networks**
  - Radial basis function (RBF) network
  - multilayer perceptron
- **Crossover Operators**
  - Exchange the basis functions
  - Exchange the topology
- **Mutation Operators**
  - Deleting a weight connection
  - Adding a weight connection
  - Deleting one basis function.
  - Adding one basis function.

MOO Algorithm & Fitness Evolution

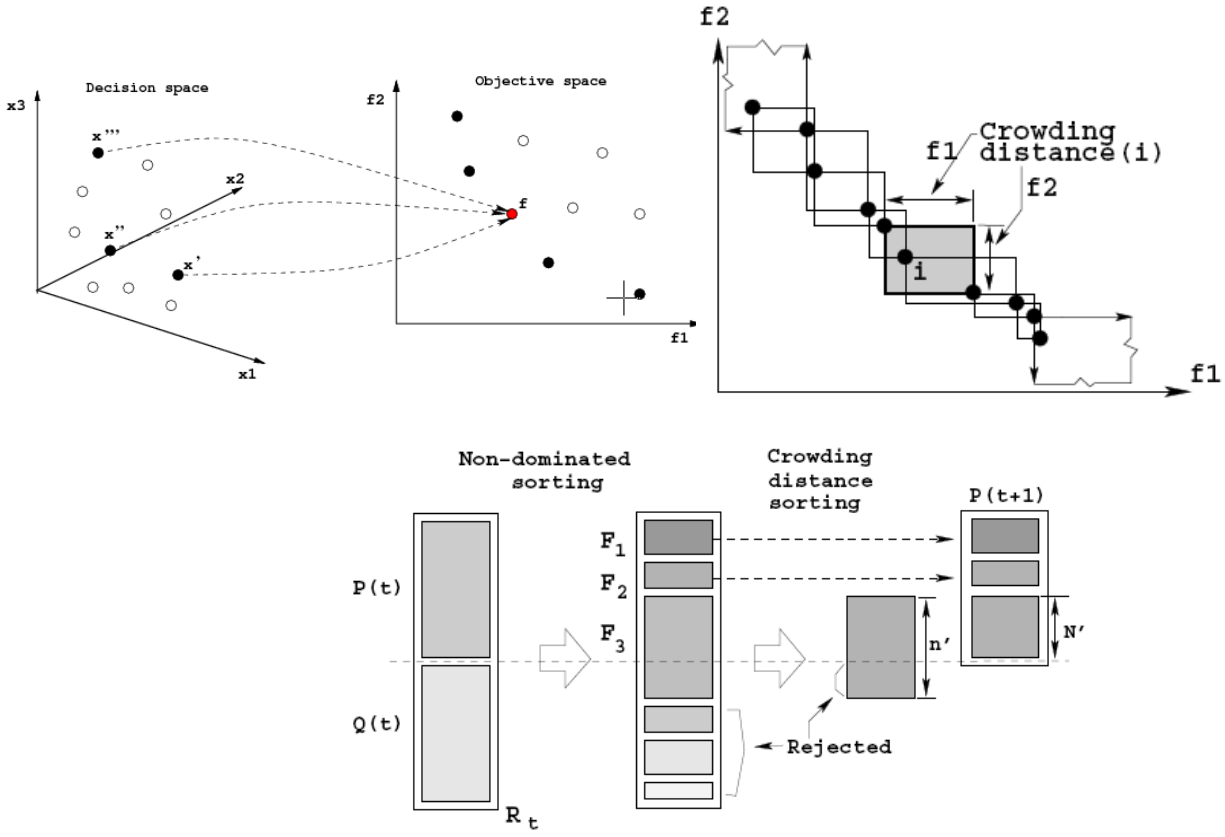
State-of-the-art MOO algorithms

- Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D) [Zhang07]
  - Decomposes a MOO problem into a number of different single objective optimization subproblems
  - Use neighbourhood relations among these subproblems

- NSGA II and fitness sharing [Srinivas95]
  - nondominated sorting algorithm
  - fitness sharing for more diversity in the search.

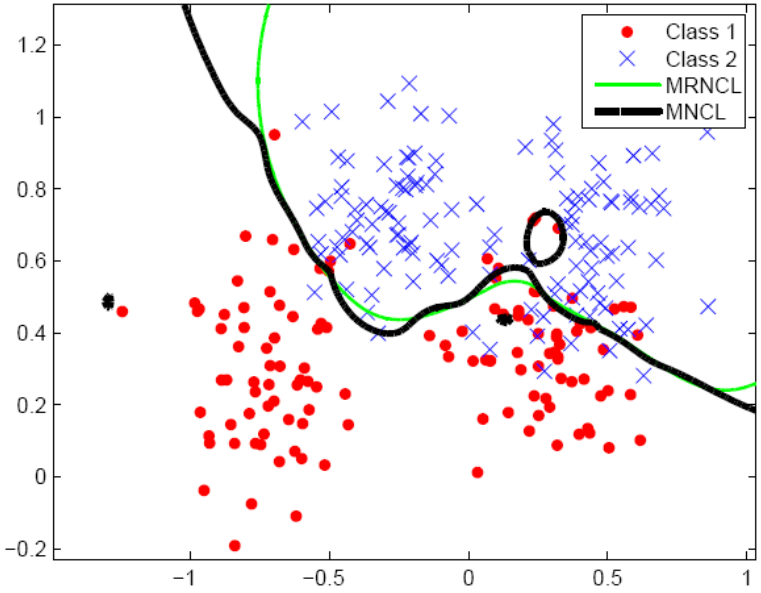
NSGA-II

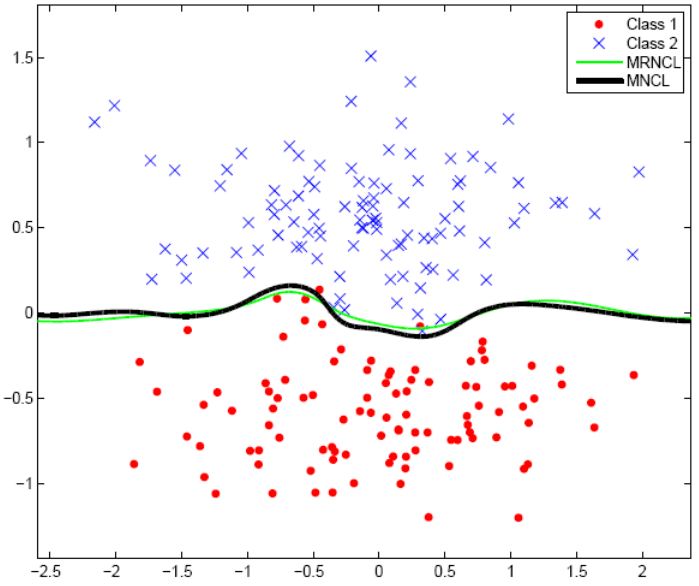
- Pareto front for multiple optimal sets
- Use of ‘crowding’ for well-spread set of non-dominated gene sets
- Find smallest set of genes that classify samples in training set and extrapolate to a hold-out set
- Various MOEA variants since developed by others



- See: Kangal resources, incl. K. Debs’ web-pages http://www.iitk.ac.in/kangal/pub.htm

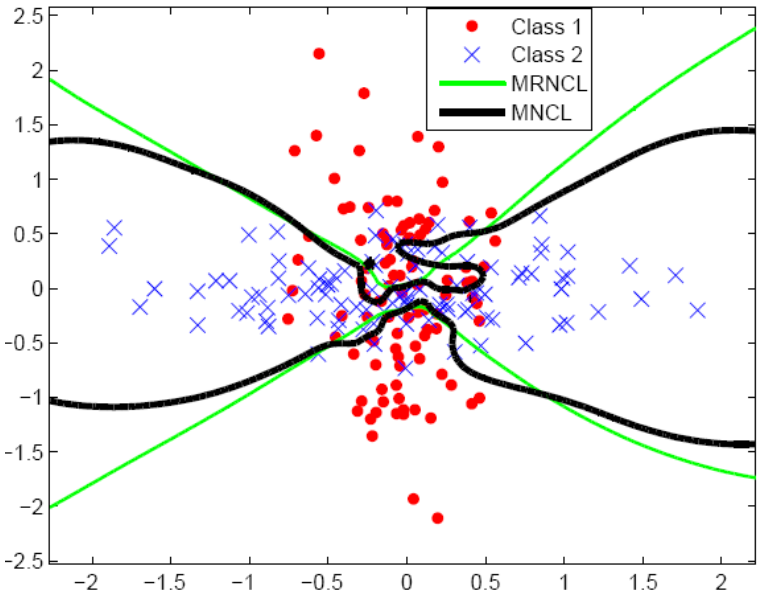
Experiments: Synthetic Data Sets (1)

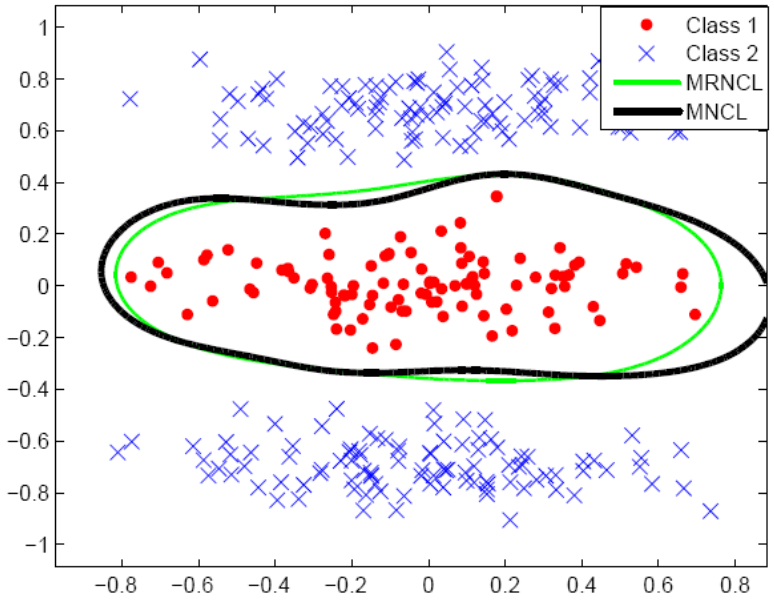




- Comparison of MRNCL and MNCL on four synthetic classification data sets. Two classes are shown as crosses and dots.
- LEFT: Synth problem
- RIGHT: Overlap problem

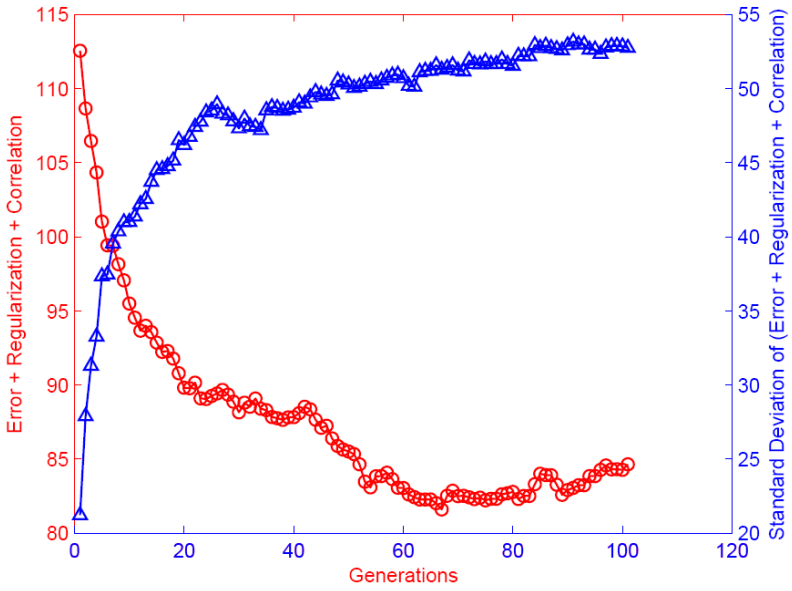
Experiments: Synthetic Data Sets (2)

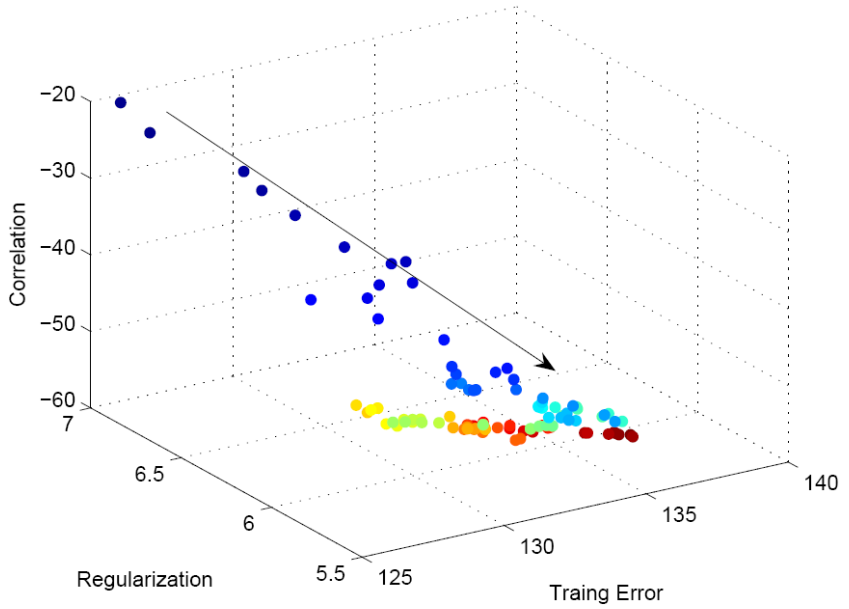




- LEFT: Bumpy problem
- Right: Relevance problem
- The separating lines were obtained by projecting test data over a grid. The lines in green(thin) and black(thick) were obtained by MRNCL and MNCL, respectively.

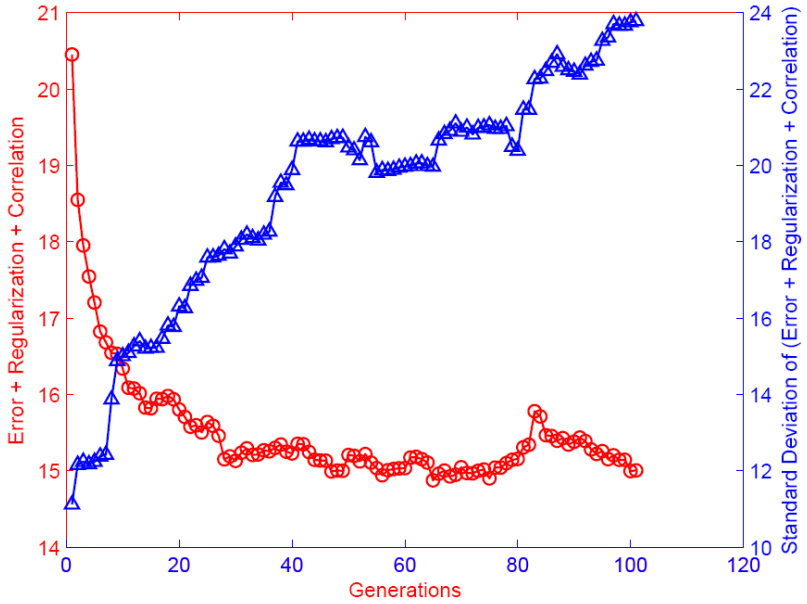
Evolutionary Information (Banana)

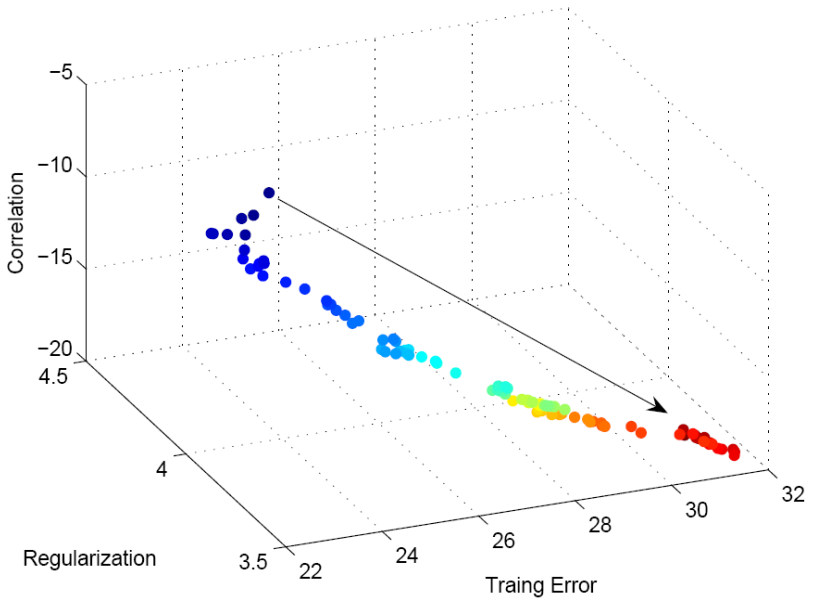




- Left: the summation of the mean of three objectives, training error, regularization and correlation in different generations. The right-y axis (blue line with triangles) is the standard deviation of the summation.
- Right: the mean value of these three objectives in different generations. The arrow points from the beginning (Generation = 1) to end (Generation = 100).

Evolutionary Information (Overlap)





- Although the direct summation of the three objectives into one term is an inaccurate estimation, as we do not consider the combination weights and we use the mean value instead of summation of the three objectives for every individual, the summation does reflect the tendency of MRNCL in the multiobjective algorithm.

Further Topics and Open Discussions

- Online Ensemble Learning
  - Online Bagging
  - Online Boosting
- Imbalanced Ensemble Learning

On-line Bagging

- When the number of examples tends to infinite, each base model contains *K* copies of each of the original training examples, where the distribution of *K* tends to a *Poisson(1)* distribution.

- Inputs:
  - Example *d* to be learnt.
  - Current ensemble **h**.
  - On-line learning algorithm *Lo*.

OnlineBagging$(\mathbf{h}, d)$

For each base model h_m, $(m \in \{1, 2, \ldots, M\})$ in the ensemble,

- Set k according to $Poisson(1)$.
- Do k times

  $h_m = L_o(h_m, d)$

Negative Correlation On-line Learning

From Off-line to On-line Boosting

Off-line

Input

- **set of labeled training samples**

$$\mathcal{X} = \{(\mathbf{x}_i, y_i)\},\ i = 1, ..., L,\ y_i = \pm 1$$

- **weight distribution over samples**

$$D_0 = 1/L$$

Algorithm

For n=1 to N

- **train a weak classifier using samples and weight distribution**

$$h_n^{weak}(\mathbf{x}) = \mathcal{L}(\mathcal{X}, D_{n-1})$$

- **calculate error** e_n
- **calculate confidence** $\alpha_n = f(e_n)$
- **update weight distribution** D_n

End

On-line

Input

- **ONE labeled training sample**

$$(\mathbf{x}, y),\ y = \pm 1$$

- **strong classifier to update**
- **initial sample importance** $\lambda = 1$

Algorithm

For n=1 to N

- **update weak classifier using samples and importance**

$$h_n^{weak}(\mathbf{x}) = \mathcal{L}(h_n^{weak}, (\mathbf{x}, y), \lambda)$$

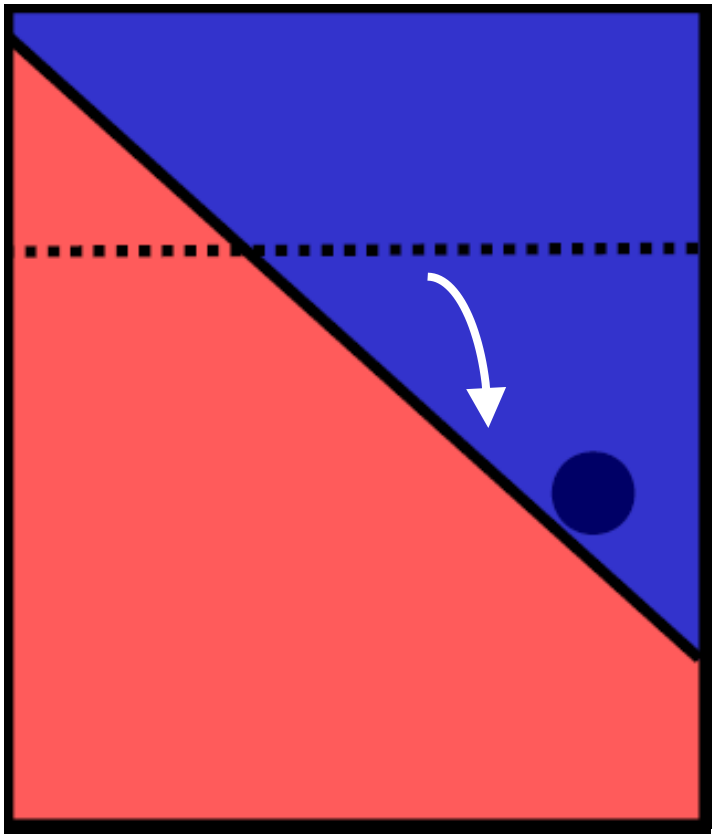
- **update error estimate** $\widehat{e}_n$
- **update confidence** $\alpha_n = f(\widehat{e}_n)$
- **update importance** λ

End

Online Boosting

[Oza, Russell 01]

Feature space



$= \alpha_1 \times h_1 + \alpha_2 \times h_2$

Input

- **ONE labeled training sample**
- **strong classifier**

Algorithm

- **initial sample importance**

for n=1 to N // number of weak classifiers

- **update weak classifier using sample and importance**
- **update error estimate**
- **update classifier weight**
- **update sample importance**

end

Result

$$H(\mathbf{x}) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n \, h_n(\mathbf{x})\right)$$

The Class Imbalance Problem I

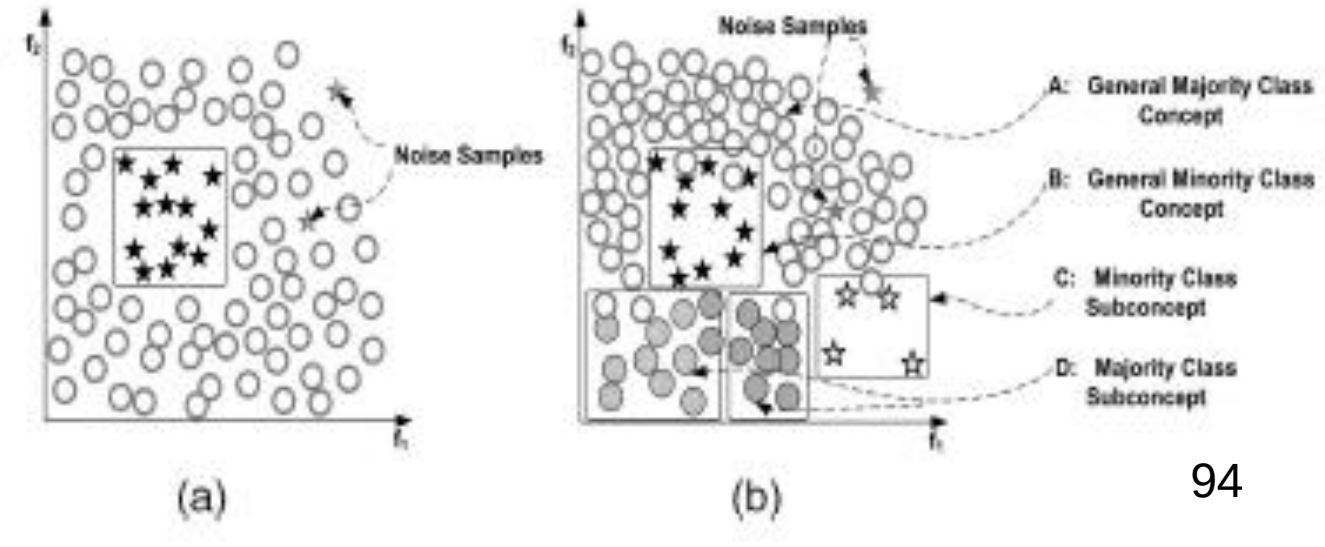
- Data sets are said to be balanced if there are, approximately, as many positive examples of the concept as there are negative ones.

- There exist many domains that do not have a balanced data set.

- ***Examples:***
  - Fault detection
  - Fraud detection

Nature of the Imbalanced learning problem(2/3)

- Relative imbalance and imbalance due to rare instances(absolute rarity)
- Data set complexity
  - Primary determining factor of classification deterioration
- Within-class imbalance



The Class Imbalance Problem

- The problem with class imbalances is that standard learners are often biased towards the majority class.
- That is because these classifiers attempt to reduce global quantities such as the error rate, not taking the data distribution into consideration.
- As a result examples from the overwhelming class are well-classified whereas examples from the minority class tend to be misclassified.

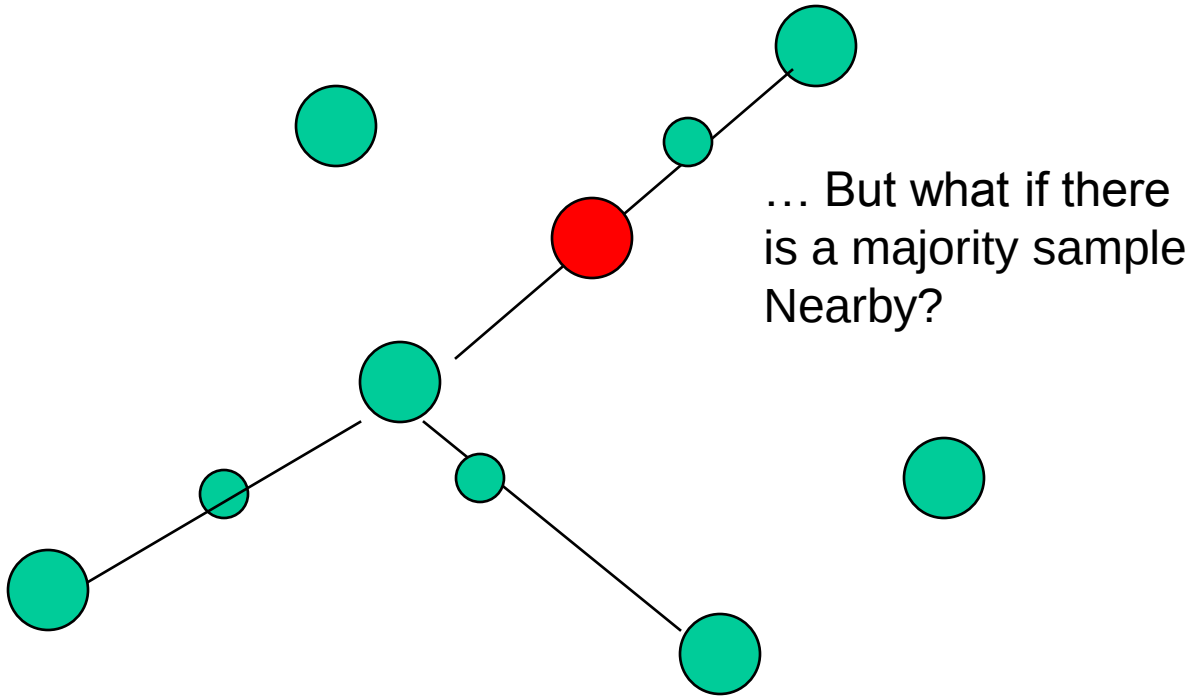
Supervised Classification Techniques

- Manipulating data records (oversampling / undersampling / generating artificial examples)
- Cost-sensitive classification techniques
- Ensemble based algorithms (SMOTEBoost, RareBoost

Manipulating Data Records

- **Over-sampling the rare class**
  - Make the duplicates of the rare events until the data set contains as many examples as the majority class => balance the classes
  - Does not increase information but increase misclassification cost
- **Down-sizing (undersampling) the majority class**
  - Sample the data records from majority class (Randomly, Near miss examples, Examples far from minority class examples (far from decision boundaries)
  - Introduce sampled data records into the original data set instead of original data records from the majority class
  - Usually results in a general loss of information and overly general rules
- **Generating artificial anomalies**
  - SMOTE (Synthetic Minority Over-sampling TEchnique) - new rare class examples are generated inside the regions of existing rare class examples
  - Artificial anomalies are generated around the edges of the sparsely populated data regions
  - Classify synthetic outliers vs. real normal data using active learning

SMOTE's Informed Oversampling Procedure



: Minority sample

: Synthetic sample

: Majority sample

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

		PREDICTED CLASS	
		Class=Yes	Class=No
ACTUAL CLASS	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

		PREDICTED CLASS	
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a (TP)	b (FN)
	Class=No	c (FP)	d (TN)

- Most widely-used metric:

$$\text{Accuracy} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class 1 example

Cost-Sensitive Measures

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Open Discussions

- How to further use diversity to encourage performance of ensemble?
- Is there any way that let us know more on ensemble?
- White box ensemble?

Tutorial on Ensemble of Classifiers

- *Survey of Boosting from an Optimization Perspective.* Manfred K. Warmuth and S.V.N. Vishwanathan. ICML'09, Montreal, Canada, June 2009.
- *Theory and Applications of Boosting*. Robert Schapire. NIPS'07, Vancouver, Canada, December 2007.
- *From Trees to Forests and Rule Sets--A Unified Overview of Ensemble Methods*. Giovanni Seni and John Elder. KDD'07, San Jose, CA, August 2007.

References

- [Jordan94] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. Neural Computation, 6(2):181-214, 1994.
- [Breiman96] L. Breiman. Bagging predictors. Machine Learning, 24(2):123-140, 1996.
- [Breiman01] L. Breiman. Random forests. Machine Learning, 45(1):5-32, 2001.
- [Schapire98] R. E. Schapire, Y. Freund, P. Barlett, and W. S. Lee. Boosting the margin: A new explanation for the e?ectiveness of voting methods. The Annals of Statistics, 26(5):1651-1686, 1998.
- [Liu99] Y. Liu and X. Yao. Ensemble learning via negative correlation. Neural Networks, 12(10):1399-1404, 1999.
- [Krogh95] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In Advances in Neural Information Processing Systems 7, pages 231-238, 1995.
- [Yule1900] U. Yule. On the association of attributes in statistics. Philosophical Transactions of the Royal Society of London. Series A, 194:257-319, 1900.
- [Dietterich00] T. G. Dietterich. Ensemble methods in machine learning. Lecture Notes in Computer Science, 1857:1-15, 2000.
- [Sneath73] P. Sneath and R. Sokal. Numerical Taxonomy. W H Freeman & Co, 1973.
- [Ho98] T. K. Ho. The random subspace method for constructing decision forests. IEEE Transaction on Pattern Analysis and Machine Intelligence, 20(8):832-844, 1998.
- [Cunningham 00] P. Cunningham and J. Carney. Diversity versus quality in classification ensembles based on feature selection. In ECML'00: Proceedings of the 11th European Conference on Machine Learning, pages 109-116, 2000.

- [Kohavi 96] R. Kohavi and D. H.Wolpert. Bias plus variance decomposition for zero-one loss functions. In Proceedings of the Thirteenth International Conference on Machine Learning, pages 275-283, Morgan Kaufmann, 1996.
- [Hansen 90] L. K. Hansen and P. Salamon. Neural network ensembles. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(10):993-1001, 1990.
- [Partridge 97] D. Partridge and W. J. Krzanowski. Software diversity: Practical statistics for its measurement and exploitation. Information and Software Technology, 39:707-717, 1997.
- [Chen08:Thesis] H.Chen. Diversity and Regularization in Neural Network Ensembles. School of Computer Science, University of Birmingham.
- [Opitz99] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research, 11:169-198, 1999.
- [Opitz96]D. Opitz and J. Shavlik. Actively searching for an effective neural-network ensemble. Connection Science, 8(3/4), 337-353, 1996.
- [Carney 99]J. Carney and P. Cunningham. Tuning diversity in bagged neural network ensembles. Technical Report TCD-CS-1999-44, Trinity College Dublin, 1999.
- [Islam03] Md. M. Islam, X Yao, K. Murase."A constructive algorithm for training cooperative neural network ensembles. IEEE Transactions on Neural Networks, vol.14, no.4, pp. 820- 834, July 2003.
- [Ueda00] N. Ueda. Optimal linear combination of neural networks for improving classification performance. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(2):207-215, 2000.
- [Yao98] X. Yao and Y. Liu. Making use of population information in evolutionary artificial neural networks. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 28(3):417-425, 1998.

- [Zhang06] Yi Zhang, Samuel Burer, and W. Nick Street. 2006. Ensemble Pruning Via Semi-definite Programming. J. Mach. Learn. Res. 7, 1315-1338, December 2006.
- [Tipping01] M. E. Tipping. Sparse bayesian learning and the relevance vector machine. Journal of Machine Learning Research, 1(3):211-244, 2001.
- [Zhou02] Z. Zhou, J. Wu, and W. Tang. Ensembling neural networks: many could be better than all. Arti¯cial Intelligence, 137(1-2):239-263, 2002.
- [Zhang07] Q. Zhang, H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. IEEE Transactions on Evolutionary Computation, vol.11, no.6, pp.712-731, Dec. 2007
- [Srinivas95] N. Srinivas and K. Deb. Multiobjective function optimization using nondominated sorting genetic algorithms. Evolutionary Computation, 2(3):221-248, 1995.
- [Chandra06] Arjun Chandra and Xin Yao. Evolving hybrid ensembles of learning machines for better generalisation. Neurocomputing 69, 7-9, 686-700, 2006.

The End … is the Beginning!

Thanks for your attention!

Slides and more references available at

http://www.cs.bham.ac.uk/~hxc/tutorial/