Geometric Programming for Aggregation of Binary Classifiers

Sunho Park and Seungjin Choi

Department of Computer Science
Pohang University of Science and Technology, Korea
{titan,seungjin}@postech.ac.kr
http://mlg.postech.ac.kr

May 27, 2011

Outline

- Multiclass learning
  - Direct vs binary decomposition
  - Aggregation of binary problems
- Geometric programming for aggregation of binary classifiers
  - Softmax model
  - ℓ_1-norm regularized maximum likelihood estimation
  - Geometric programming formulation
- Experiments
- Conclusions and future work

Outline

- Multiclass learning
  - Direct vs binary decomposition
  - Aggregation of binary problems
- Geometric programming for aggregation of binary classifiers
  - Softmax model
  - ℓ_1-norm regularized maximum likelihood estimation
  - Geometric programming formulation
- Experiments
- Conclusions and future work

Outline

- Multiclass learning
  - Direct vs binary decomposition
  - Aggregation of binary problems
- Geometric programming for aggregation of binary classifiers
  - Softmax model
  - ℓ_1-norm regularized maximum likelihood estimation
  - Geometric programming formulation
- Experiments
- Conclusions and future work

Outline

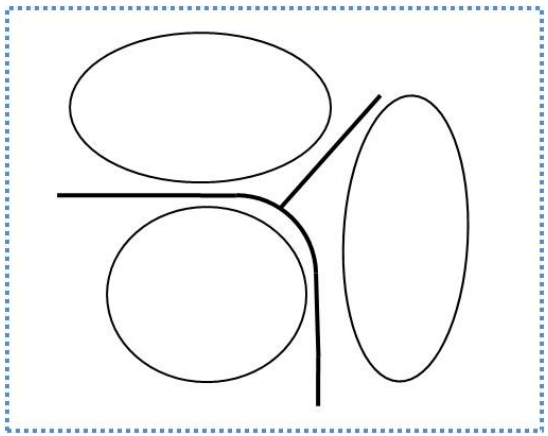
- Multiclass learning
  - Direct vs binary decomposition
  - Aggregation of binary problems
- Geometric programming for aggregation of binary classifiers
  - Softmax model
  - ℓ_1-norm regularized maximum likelihood estimation
  - Geometric programming formulation
- Experiments
- Conclusions and future work

Outline

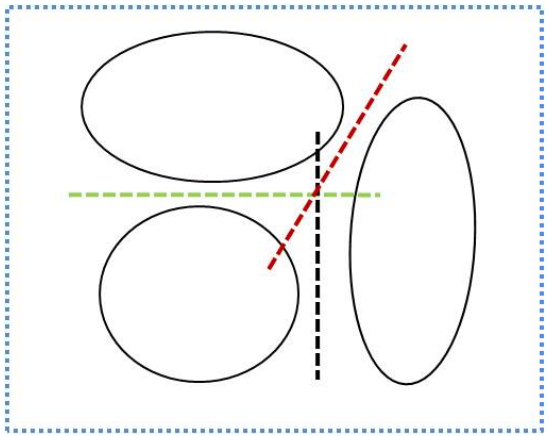
- Multiclass learning
  - Direct vs binary decomposition
  - Aggregation of binary problems
- Geometric programming for aggregation of binary classifiers
  - Softmax model
  - ℓ_1-norm regularized maximum likelihood estimation
  - Geometric programming formulation
- Experiments
- Conclusions and future work

Multiclass Learning

- Multiclass learning ($K \geq 3$): Assign class label $y \in \{1, ..., K\}$ to data $\mathbf{x}$.
- Direct method vs. Binary decomposition method



(a) Direct method



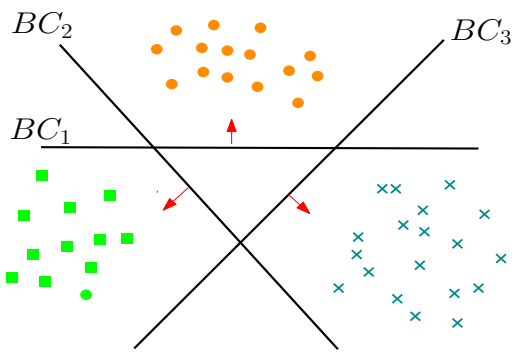
(b) All-pairs (APs)

- Advantages of binary decomposition over direct methods
  - Easy and simple to learn classifiers
  - Extensive studies and implementations
  - Better suited to parallel computation

Binary Decomposition

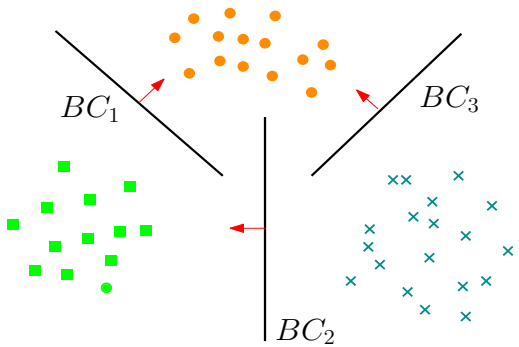
Binary decomposition = Binary encoding

✠ One versus all (OVA)



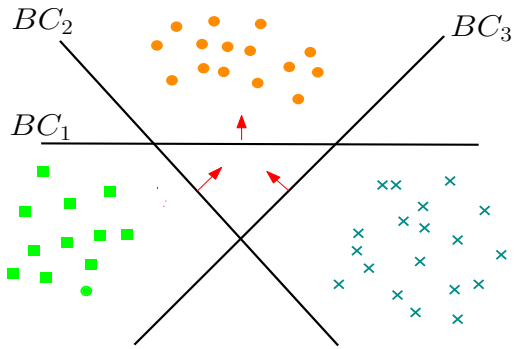
	c1	c2	c3
BC_1	1	0	0
BC_2	0	1	0
BC_3	0	0	1

✠ All-pairs (APs)



	c1	c2	c3
BC_1	1	0	△
BC_2	△	1	0
BC_3	1	△	0

✠ ECOC (complete code)



	c1	c2	c3
BC_1	1	0	0
BC_2	1	0	1
BC_3	1	1	0

Code matrix: Codewords in column and binary problems in row

Binary Decomposition: Training

- Examples: OVA

$$\mathbf{C} = \begin{array}{|c|c|c|c|} \hline & \text{class 1} & \text{class 2} & \text{class 3} \\ \hline BC_1 & 1 & 0 & 0 \\ BC_2 & 0 & 1 & 0 \\ BC_3 & 0 & 0 & 1 \\ \hline \end{array}$$

  - each training example produces M binary target values

$$\text{e.g., } (\mathbf{x}_i, y_i = 2) \Rightarrow (\mathbf{x}_i, C_{1,y_i} = 0), (\mathbf{x}_i, C_{2,y_i} = 1), (\mathbf{x}_i, C_{3,y_i} = 0).$$

- Training binary classifiers
  - The jth binary classifier is trained on $\{\mathbf{x}_i, C_{j,y_i}\}_{i=1}^N$:

$$BC_1\text{: } \{\mathbf{x}_i, C_{1,y_i}\}_{i=1}^N, \ BC_2\text{: } \{\mathbf{x}_i, C_{2,y_i}\}_{i=1}^N, \cdots, \ BC_M\text{: } \{\mathbf{x}_i, C_{M,y_i}\}_{i=1}^N,$$

- Probability estimates determined by binary classifiers (for instance, one uses SVM with Platt's sigmoid model for binary classifers)

$$Q_{j,i} \triangleq P(C_{j,y_i} = 1 \,|\, \mathbf{x}_i), \quad \mathbf{q}_i = [Q_{1,i}, ..., Q_{M,i}]^\top \in \mathbb{R}^M.$$

Aggregation of Binary Classifiers: Decoding

Aggregation of binary classifiers: Combine solutions to binary problems to determine a final answer to multiclass problems.

- Heuristics
  - APs: majority voting
  - OVA: max win
- Hard decoding: Finds a codeword which best matches a collection of predictive results computed by binary classifiers

$$\hat{y}_i = \arg\min_k \rho_{avg}(\mathbf{c}_k, \mathbf{q}_i), \quad \left(\triangleq \frac{1}{M} \sum_{j=1}^{M} d(C_{j,k}, Q_{j,i}) \right)$$

where d is a discrepancy measure, e.g., Hamming loss or exponential loss.

- Probabilistic decoding:
  - Class membership probabilities $\{P_{k,i} \triangleq p(y_i = k \mid \mathbf{x}_i)\}_{k=1}^{K}$.
  - Prediction: $\hat{y}_i = \arg\max_k P(y_i = k|\mathbf{x}_i)$.

Probabilistic Decoding: Bradley-Terry Models

- Most of existing methods make use of (generalized) Bradley-Terry models to relate binary predictions with class membership probabilities.
- Class membership probabilities are treated as parameters
  e.g. 3 class problem with APs binary decomposition (decoding: the pairwise coupling (=Bradley-Terry) model (Hastie and Tibshirani, 1998))

$$\pi_{1,*} = p(y_* = 1 | y_* = 1 \text{ or } y_* = 2) = \frac{P_{1,*}}{P_{1,*} + P_{2,*}},$$

$$\pi_{2,*} = p(y_* = 1 | y_* = 1 \text{ or } y_* = 3) = \frac{P_{1,*}}{P_{1,*} + P_{3,*}},$$

$$\pi_{3,*} = p(y_* = 2 | y_* = 2 \text{ or } y_* = 3) = \frac{P_{2,*}}{P_{2,*} + P_{3,*}},$$

  - $\{Q_{j,*}\}$ are the probability estimates obtained by binary classifiers
  - Class membership probabilities $\{P_{k,*}\}$ are estimated by minimizing KL divergence between $\{Q_{j,*}\}$ and $\{\pi_{j,*}\}$

Issues in Existing Probabilistic Decoding Methods

- In methods based on *Bradley-Terry models*, class membership probabilities are treated as parameters. Thus the number of parameters grows with examples in the training set.
- Predictions by unreliable binary classifiers degrade the overall classification performance.
- Recent effort for optimal aggregation (N. Yukinawa, et al., 2009)
  - High-dimensional optimization problem: Both aggregation weights and class membership probabilities are treated as parameters.
  - Non convex optimization problem: Does not guarantee a global solution.

Our Approach Here

- Aggregation model: Softmax model (Park and Choi, ICDM 2010)
  - Softmax function is used for modeling class membership probabilities.
  - Aggregation weights are only parameters to be tuned.
- Estimation of aggregation weights: ℓ_1-norm regularized maximum likelihood estimation
  - Convex optimization: Guarantees a global solution.
  - This optimization problem is easily solved by geometric programming.

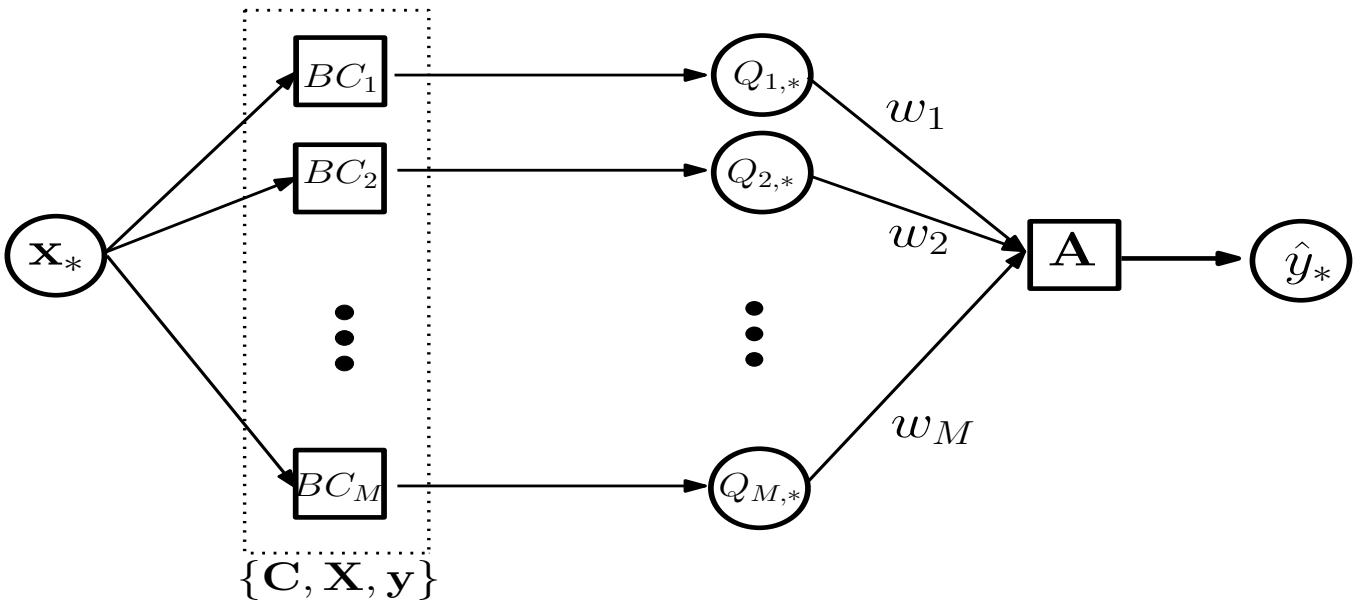


Figure: The pictorial view of aggregation of binary classifiers

Aggregation Model

Softmax model for class membership probabilities (Park and Choi, ICDM 2010)

$$p(y_i = k \mid \mathbf{w}, \mathbf{x}_i) = \frac{\exp\{-\rho_w(\mathbf{c}_k, \mathbf{q}_i)\}}{\sum_{j=1}^{K} \exp\{-\rho_w(\mathbf{c}_j, \mathbf{q}_i)\}},$$

where ρ_w is the weighted sum of discrepancies ($\mathbf{w} \in \mathbb{R}^M$, $w_j \geq 0$)

$$\rho_w(\mathbf{c}_k, \mathbf{q}_i) = \sum_{j=1}^{M} w_j \, d(C_{j,k}, Q_{j,i}),$$

where $d(C_{j,k}, Q_{j,i})$ is the cross-entropy error function:

$$d(C_{j,k}, Q_{j,i}) = -C_{j,k} \log Q_{j,i} - (1 - C_{j,k}) \log(1 - Q_{j,i}),$$

($C_{j,k} = \triangle : d(C_{j,k}, Q_{j,i}) = 0$).

A probabilistic extension of the loss-based decoding.

ℓ_1-Norm Regularized Maximum Likelihood Estimation

- Likelihood of training data

$$p(\mathbf{y}|\mathbf{w},\mathbf{X}) = \prod_{i=1}^{N}\prod_{k=1}^{K}\left(\frac{\exp\{-\rho_w(\mathbf{c}_k,\mathbf{q}_i)\}}{\sum_{j=1}^{K}\exp\{-\rho_w(\mathbf{c}_j,\mathbf{q}_i)\}}\right)^{\delta(k,y_i)},$$

$$= \prod_{i=1}^{N}\prod_{k=1}^{K}\left(\frac{1}{\sum_{j=1}^{K}\exp\left\{-\mathbf{w}^\top\boldsymbol{\varphi}_i^{j,k}\right\}}\right)^{\delta(k,y_i)},$$

where $[\boldsymbol{\varphi}_i^{j,k}]_l = d(C_{l,j},Q_{l,i}) - d(C_{l,k},Q_{l,i})$.

- Minimization of negative log-likelihood with ℓ_1 norm regularization

$$\mathcal{J} = -\log p(\mathbf{y}|\mathbf{w},\mathbf{X}) + \lambda\sum_{j=1}^{M}|w_j|$$

$$= \sum_{i=1}^{N}\log\left(\sum_{j=1}^{K}\exp\left\{-\mathbf{w}^\top\boldsymbol{\varphi}_i^{j,y_i}\right\}\right) + \boldsymbol{\lambda}^\top\mathbf{w} + \text{const},$$

subject to the constraints $w_j \geq 0$ for $j = 1,\ldots,M$.

Our Optimization

- Our minimization problem is given by

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, ..., M. \end{aligned}$$

- The log-sum-exp function, $\log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right)$, is convex $\Rightarrow$ Convex optimization problem.
- We convert this optimization to an equivalent optimization problem that has the form of geometric programming.

Geometric Programming

- The objective function and constraint functions have a special form
  - the standard form of geometric programming (in posynomial form):

$$\begin{array}{ll} \text{minimize} & g_0(\mathbf{a}) \\ \text{subject to} & g_j(\mathbf{a}) \leq 1, \quad j = 1, ..., M' \\ & e_j(\mathbf{a}) = 1, \quad j = 1, ..., M''. \end{array}$$

    where $g_0, ..., g_{M'}$ are posynomials and $e_1, ..., e_{M''}$ are monomials.
  - a monomial function ($g : \mathbb{R}^M \to \mathbb{R}$ with dorm $g = \mathbb{R}^M_{++}$):

$$g(\mathbf{a}) = c a_1^{\alpha_1} a_2^{\alpha_2} \cdots a_M^{\alpha_M},$$

    where $\alpha_j \in \mathbb{R}$ and $c > 0$.
  - a posynomial function:

$$g(\mathbf{a}) = \sum_{j=1}^{K} c_j a_1^{\alpha_{1,j}} a_2^{\alpha_{2,j}} \cdots a_M^{\alpha_{M,j}}.$$

- Global solution: geometric programming in posynomial form can be transformed into the equivalent convex optimization problem (geometric programming in convex form).

Geometric Programms for Aggregation

Our minimization problem

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, \ldots, M. \end{aligned}$$

strategy: original minimization problem

Geometric Programms for Aggregation

Our minimization problem

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, \ldots, M. \end{aligned}$$

strategy: original minimization problem $\Leftrightarrow$ **geometric programming in convex form**

Geometric Programms for Aggregation

Our minimization problem

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, \ldots, M. \end{aligned}$$

strategy: original minimization problem $\Leftrightarrow$ **geometric programming in convex form**
$\Leftrightarrow$ **geometric programming in posynomial form**

Geometric Programms for Aggregation

Our minimization problem

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log \left(\sum_{j=1}^{K} \exp \left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\} \right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, \ldots, M. \end{aligned}$$

strategy: original minimization problem $\Leftrightarrow$ **geometric programming in convex form**
$\Leftrightarrow$ **geometric programming in posynomial form**

Using $a_j = \exp\{w_j\}, s_i = \exp\{z_i\} \Rightarrow \prod_{l=1}^{M} a_l^{-[\boldsymbol{\varphi}_i^{j,y_i}]_l} = \exp\left\{ -\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i} \right\}, \prod_{i=0}^{N} s_i = \exp\left\{ \sum_{i=0}^{N} z_i \right\}.$

Geometric Programms for Aggregation

Our minimization problem

$$\begin{aligned} \text{minimize}_{\mathbf{w}} \quad & \sum_{i=1}^{N} \log\left(\sum_{j=1}^{K} \exp\left\{-\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i}\right\}\right) + \boldsymbol{\lambda}^\top \mathbf{w} \\ \text{subject to} \quad & w_j \geq 0, j = 1, \ldots, M. \end{aligned}$$

strategy: original minimization problem $\Leftrightarrow$ **geometric programming in convex form**
$\Leftrightarrow$ **geometric programming in posynomial form**

Using $a_j = \exp\{w_j\}, s_i = \exp\{z_i\} \Rightarrow \prod_{l=1}^{M} a_l^{-[\boldsymbol{\varphi}_i^{j,y_i}]_l} = \exp\left\{-\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i}\right\}, \prod_{i=0}^{N} s_i = \exp\left\{\sum_{i=0}^{N} z_i\right\}.$

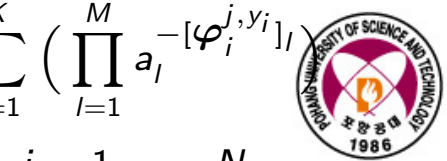
✠ Convex form

$$\begin{aligned} \text{minimize}_{\mathbf{w},\mathbf{z}} \quad & \sum_{i=0}^{N} z_i \\ \text{subject to} \quad & \mathbf{w}^\top \boldsymbol{\lambda} - z_0 \leq 0, \\ & h_i(\mathbf{w}) - z_i \leq 0, \ i = 1, \ldots, N, \\ & w_j \geq 0, \ j = 1, \ldots, M. \end{aligned}$$

where
$h_i(\mathbf{w}) = \log\left(\sum_{j=1}^{K} \exp\left\{-\mathbf{w}^\top \boldsymbol{\varphi}_i^{j,y_i}\right\}\right).$

✠ Posynomial form

$$\begin{aligned} \text{minimize}_{\mathbf{a},\mathbf{s}} \quad & \prod_{i=0}^{N} s_i \\ \text{subject to} \quad & s_0^{-1} \prod_{l=1}^{M} a_l^{\lambda} \leq 1, \\ & s_i^{-1} \sum_{j=1}^{K} \left(\prod_{l=1}^{M} a_l^{-[\boldsymbol{\varphi}_i^{j,y_i}]_l}\right) \\ & \leq 1, \quad i = 1, \ldots, N, \\ & a_l^{-1} \leq 1, \quad l = 1, \ldots, M. \end{aligned}$$



Experimental Results

Experimental setting

- Binary decomposition methods: APs, OVA and ECOC ($K < 8$: complete code, $K \geq 8$: spare random coding)
- Binary classifier: linear SVM with Platt's sigmoid model

$$Q_{j,*} = \frac{1}{1 + \exp(-Af_j(\mathbf{x}_*) + B)},$$

- We use 'Mosek' to solve the geometric programming problems:

$$\widehat{\mathbf{w}} = \log(\widehat{\mathbf{a}}),$$

where $\widehat{\mathbf{a}}$ is a solution obtained by geometric programming.

- Comparison
  - Loss-based decoding (hard decoding) (E. L. Allwein, et al., 2000 jmlr)
  - WMAP (N. Yukinawa, et al., IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2009.)
    1. Probabilistic decoding method based on the *generalized Bradley-Terry model*
    2. Class membership probabilities are treated as parameters
       $\Rightarrow$ the number of parameters: M + NK (N: # training samples, K: # classes)
    3. Local solutions.

Experimental Results

Performance evaluation of three methods on UCI datasets

Table: Data description.

	# samples	# attributes	# classes
glass	214	9	7
satimage	6435	36	7
yeast	1484	8	10
pendigits	10992	16	10
vowel	990	10	11
isolet	7797	617	26
letter	20000	16	26

Experimental Results

Table: Performance of three methods on real-world datasets in terms of classification accuracy (10 fold cross-validation)

		Loss-based decoding	WMAP	Our method
glass	APs	0.662(0.104)	0.648(0.115)	**0.671**(0.096)
	OVA	0.619(0.159)	0.619(0.131)	**0.629**(0.152)
	ECOC	0.610(0.107)	0.614(0.113)	**0.657**(0.131)
sat-image	APs	0.862(0.013)	0.861(0.016)	**0.863**(0.013)
	OVA	0.770(0.014)	0.770(0.014)	**0.810**(0.010)
	ECOC	0.813(0.015)	0.812(0.015)	**0.852**(0.012)
yeast	APs	0.589(0.045)	0.588(0.047)	**0.595**(0.043)
	OVA	0.553(0.035)	0.553(0.035)	**0.555**(0.042)
	ECOC	0.520(0.052)	0.596(0.030)	**0.607**(0.021)
pen-digits	APs	0.972(0.005)	0.971(0.005)	**0.973**(0.005)
	OVA	0.848(0.007)	0.848(0.007)	**0.867**(0.006)
	ECOC	0.912(0.007)	0.912(0.005)	**0.943**(0.009)
vowel	APs	0.799(0.025)	0.783(0.035)	**0.805**(0.025)
	OVA	0.453(0.057)	0.453(0.057)	**0.478**(0.067)
	ECOC	0.526(0.048)	0.571(0.065)	**0.658**(0.041)
isolet	APs	**0.939**(0.010)	0.938(0.011)	**0.939**(0.010)
	OVA	0.780(0.009)	0.780(0.009)	**0.862**(0.011)
	ECOC	0.829(0.033)	0.853(0.015)	**0.875**(0.015)
letter	APs	0.818(0.005)	0.814(0.006)	**0.830**(0.004)
	OVA	0.534(0.014)	0.534(0.014)	**0.651**(0.009)
	ECOC	0.558(0.022)	0.567(0.028)	**0.611**(0.028)

Conclusions

Contributions

- Convex optimization for aggregation weights: ℓ_1-norm regularized maximum likelihood estimation
- Geometric programming formulation for solving the optimization problem

Comparison to WMAP

- Parameters to be tuned are only the aggregation weights.
- Geometric programming formulation yields the global solution.
- Class membership probabilities for test data are easily evaluated without further optimizations.

Conclusions

Contributions

- Convex optimization for aggregation weights: ℓ_1-norm regularized maximum likelihood estimation
- Geometric programming formulation for solving the optimization problem

Comparison to WMAP

- Parameters to be tuned are only the aggregation weights.
- Geometric programming formulation yields the global solution.
- Class membership probabilities for test data are easily evaluated without further optimizations.

Thank you for your attention!!