Probability Estimates for Multi-class Classification by Pairwise Coupling

Ting-Fan Wu*, Chih-Jen Lin* and Ruby C. Weng**
*Department of Computer Science, National Taiwan University, **Department of Statistics, National Chengchi University
Taipei, Taiwan

Multi-class Probability Estimation

- Given k classes of training data
- Pairwise probability estimates r_{ij} known
  Any new $\mathbf{x}$,
  $$r_{ij} \approx \mu_{ij} \equiv p(y = i \mid y = i \text{ or } j, \mathbf{x}) \text{ available}$$
- Goal: estimate
  $$p_i = p(y = i \mid \mathbf{x}), i = 1, \dots, k$$

Motivation

- Provide multiclass probability outputs for SVM
  By pairwise coupling (like 1vs1 voting for multi-class classification
- Applicable to other two-class methods
  Probability estimates:
  - Make post-processing possible
  - May help multi-class classification

Review of Existing Methods

- Voting:
  $$p_i = 2 \sum_{j:j \neq i} \frac{I_{\{r_{ij} > r_{ji}\}}}{k(k-1)}$$
  Problem: $p_i \leq 2/k, \forall i$, may not be good estimate
- [Refregier and Vallet 1991]:
  $$\frac{r_{ij}}{r_{ji}} \approx \frac{\mu_{ij}}{\mu_{ji}} = \frac{p_i}{p_j}, \ \forall i, j$$
  A k-variable linear system
  $$\begin{cases} \text{any } k-1 \text{ equations: } r_{ji} p_i = r_{ij} p_j \\ \sum_{i=1}^k p_i = 1 \end{cases}$$
  Problem: $\mathbf{p}$ depends on selecting of $k-1$ equations
- [Price, Kner, Personnaz, and Dreyfus 1995]
  $$\sum_{j=1}^{l} p_j = 1 \implies (\sum_{j:j \neq i} p_i + p_j) - (k-2) p_i = 1$$
  $$r_{ij} \approx \mu_{ij} = \frac{p_i}{p_i + p_j}$$
  $$\sum_{j:j \neq i} \frac{1}{r_{ij}} - (k-2) \approx \frac{1}{p_i}$$
  Solve $\mathbf{p}$, then normalize it
  Problem: a bit heuristic; no comparison so far
- [Hastie and Tibshirani, 1998]: Minimizing Kullback-Leibler distance between r_{ij} and μ_{ij}:
  $$\min_{\mathbf{p}} \sum_{i \neq j} n_{ij} r_{ij} \log(r_{ij}/\mu_{ij})$$
  $$\text{subject to } \begin{array}{l} \sum_{i=1}^k p_i = 1 \\ p_i \geq 0, \ i = 1, \dots, k \\ \mu_{ij} = p_i/(p_i + p_j) \end{array}$$
- A nonlinear system
  If $r_{ij} > 0, \forall i \neq j$, unique global minimum
  An iterative procedure to find the solution $\mathbf{p}^*$
- p_i^* the same order as $\tilde{p}_i$
  $$p_i^* > p_j^* \Leftrightarrow \tilde{p}_i > \tilde{p}_j, \text{ where } \tilde{p}_j = \frac{2 \sum_{s:s \neq j} r_{js}}{k(k-1)}$$
- $\tilde{\mathbf{p}}$ from
  $$\begin{aligned} p_i &= \sum_{j:j \neq i} (\frac{p_i + p_j}{k-1})(\frac{p_i}{p_i + p_j}) \\ &= \sum_{j:j \neq i} (\frac{p_i + p_j}{k-1}) \mu_{ij} \approx \frac{2 \sum_{s:s \neq j} r_{js}}{k(k-1)} \end{aligned} \quad (1)$$
  replacing $p_i + p_j$ with $2/k$, and μ_{ij} with r_{ij}
  Problem: if $p_i + p_j \not\approx 2/k$?

Our First Approach

- Not replacing $p_i + p_j$ with $k/2$ in (1)
  $$p_i = \sum_{j:j \neq i} (\frac{p_i + p_j}{k-1}) r_{ij}, \forall i, \text{ with } p_i \geq 0, \forall i, \ \sum_{i=1}^k p_i = 1 \quad (2)$$
- (2) : a linear system
  $$Q\mathbf{p} = \mathbf{p}, Q_{ij} = \begin{cases} r_{ij}/(k-1) & \text{if } i \neq j, \\ \sum_{s:s \neq i} r_{si}/(k-1) & \text{if } i = j. \end{cases} \quad (3)$$
- A 3×3 example of Q
  $$\begin{bmatrix} r_{12}/2 + r_{13}/2 & r_{12}/2 & r_{13}/2 \\ r_{21}/2 & r_{21}/2 + r_{23}/2 & r_{23}/2 \\ r_{31}/2 & r_{32}/2 & r_{31}/2 + r_{32}/2 \end{bmatrix}$$
- A finite Markov Chain with transition matrix Q
  $\rightarrow$ Can be easily solved
  If $r_{ij} > 0$, $i \neq j$, (3) a unique solution $\mathbf{p}$ with $0 < p_i < 1, \forall i$
- An optimization point of view
  $$\begin{aligned} (2) \ &\Rightarrow \ (k-1) p_i = \sum_{j:j \neq i} (p_i + p_j) r_{ij} \\ &\Rightarrow \ \sum_{j:j \neq i} r_{ji} p_i - \sum_{j:j \neq i} r_{ij} p_j = 0, \forall i \end{aligned}$$
- A convex problem with unique solution
  $$\min_{\mathbf{p}} \sum_{i=1}^k (\sum_{j:j \neq i} r_{ji} p_i - \sum_{j:j \neq i} r_{ij} p_j)^2$$
  $$\text{subject to } \sum_{i=1}^k p_i = 1, p_i \geq 0, \forall i$$

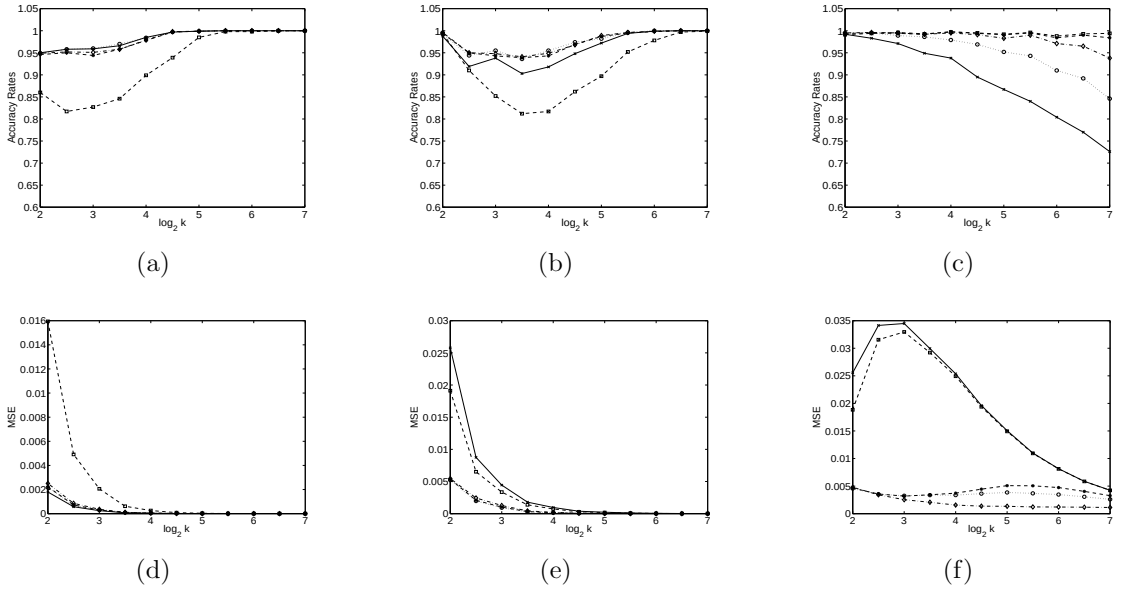


Figure 1: Accuracy: δ_{HT} (solid, $\times$), δ_V (dash, $\square$), δ_1 (dotted, $\circ$), δ_2 (dashed, $*$) and δ_{PKPD} (dashed, $\diamondsuit$)

Our Second Approach

- $p_i/(p_i+p_j) \approx r_{ij} \Rightarrow r_{ji}p_i \approx r_{ij}p_j$

$$\min_{\mathbf{p}} \frac{1}{2}\sum_{i=1}^{k}\sum_{j:j\neq i}(r_{ji}p_i - r_{ij}p_j)^2 \quad (4)$$
$$\text{subject to } \sum_{i=1}^{k} p_i = 1,\ p_i \geq 0,\ \forall i$$

- An improved version of [Refregier and Vallet]
- (4) equivalent to without $p_i \geq 0, \forall i$
- A linear-constrained convex quadratic program

$$\min_{\mathbf{p}} \frac{1}{2}\mathbf{p}^T Q\mathbf{p}, Q_{ij} = \begin{cases} \sum_{s:s\neq i} r_{si}^2 & \text{if } i=j \\ r_{ji}r_{ij} & \text{if } i\neq j \end{cases} \quad (5)$$
$$\text{subject to } \sum_{i=1}^{k} p_i = 1,\ p_i \geq 0,\ \forall i$$

- (5) solved by a linear system (KKT)

$$\begin{bmatrix} Q & \mathbf{e} \\ \mathbf{e}^T & 0 \end{bmatrix}\begin{bmatrix} \mathbf{p} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ 1 \end{bmatrix} \quad (6)$$

Generally, Q positive definite $\Longrightarrow$ (5) unique solution

$$\begin{bmatrix} Q+\Delta\mathbf{e}\mathbf{e}^T & \mathbf{e} \\ \mathbf{e}^T & 0 \end{bmatrix}\begin{bmatrix} \mathbf{p} \\ b \end{bmatrix} = \begin{bmatrix} \Delta\mathbf{e} \\ 1 \end{bmatrix} \quad (7)$$

$Q+\Delta\mathbf{e}\mathbf{e}^T$ PD and $\begin{bmatrix} Q & \mathbf{e} \\ \mathbf{e}^T & 0 \end{bmatrix}^{-1}$ exists. (7), equivalent to (6), by Gaussian elimination

$Q+\Delta\mathbf{e}\mathbf{e}^T$ symmetric and PD, Cholesky factorization saves the computational time.

- A fixed-point algorithm regardless existence of Q^{-1}

Algorithm 1

1. *Initial* $p_i \geq 0, \forall i$ *and* $\sum_{i=1}^{k} p_i = 1$.
2. *Repeat* $(t = 1, \ldots, k, 1, \ldots)$

$$p_t \leftarrow \frac{1}{Q_{tt}}[-\sum_{j:j\neq t} Q_{tj}p_j + \mathbf{p}^T Q\mathbf{p}]$$

normalize $\mathbf{p}$

until (6) is satisfied.

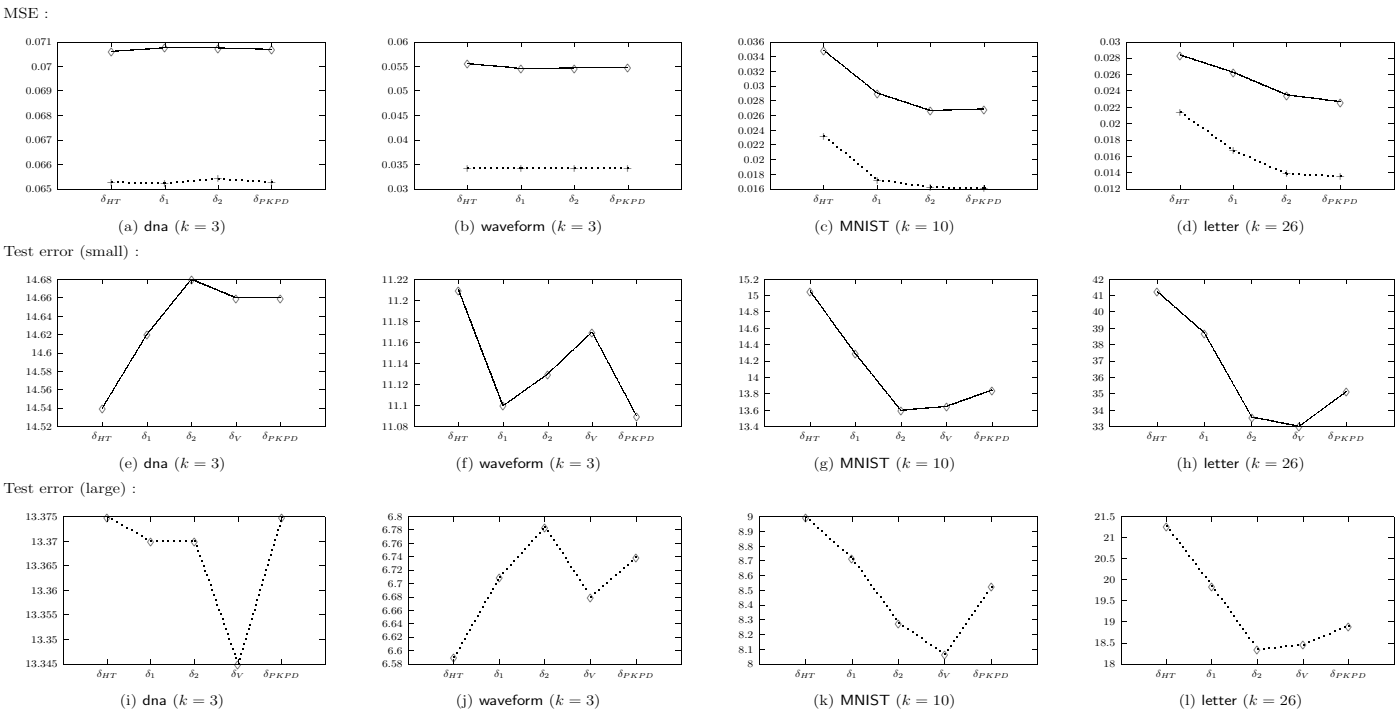
- Theorem : Converges if $r_{ij} > 0\ \forall i, j$
- Two proposed methods : linear systems $\Longrightarrow$ simpler than HT (nonlinear)

Relations Among Four Methods

- Under $\sum_{i=1}^{k} p_i = 1$ and $p_i \geq 0, \forall i$:

$$\delta_{HT}: \quad \min_{\mathbf{p}} \sum_{i=1}^{k}[\sum_{j:j\neq i}^{k}(r_{ij}\frac{1}{k} - \frac{1}{2}p_i)]^2$$
$$\delta_1: \quad \min_{\mathbf{p}} \sum_{i=1}^{k}\left[\sum_{j:j\neq i}^{k}(r_{ij}p_j - r_{ji}p_i)\right]^2$$
$$\delta_2: \quad \min_{\mathbf{p}} \sum_{i=1}^{k}\sum_{j:j\neq i}^{k}\left(r_{ij}p_j - r_{ji}p_i\right)^2$$
$$\delta_V: \quad \min_{\mathbf{p}} \sum_{i=1}^{k}\sum_{j:j\neq i}^{k}(I_{\{r_{ij}>r_{ji}\}}p_j - I_{\{r_{ji}>r_{ij}\}}p_i)^2$$

- $\delta_1 \rightarrow \delta_{HT}$: letting $p_j = \frac{1}{k}$ and $r_{ij} = \frac{1}{2}$ Ignore differences between p_i
- $\delta_2 \rightarrow \delta_V$: replace r_{ij} with $I_{\{r_{ij}>r_{ji}\}}$ Enlarge differences between p_i
- $\uparrow$: tends to underestimate the differences between p_i's
- $\downarrow$: tends to overestimate the differences between p_i's



(a) dna ($k = 3$) (b) waveform ($k = 3$) (c) MNIST ($k = 10$) (d) letter ($k = 26$)

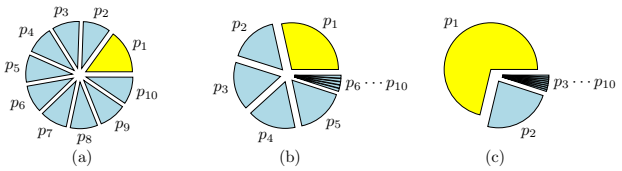
(e) dna ($k = 3$) (f) waveform ($k = 3$) (g) MNIST ($k = 10$) (h) letter ($k = 26$)

(i) dna ($k = 3$) (j) waveform ($k = 3$) (k) MNIST ($k = 10$) (l) letter ($k = 26$)

Figure 2: MSE (row 1) and Accuracy (row 2,3) of small (solid) and large (dotted) datasets. From left to right sorted by k: dna, waveform, MNIST, letter

Simulated Examples

Assuming p_i known with correct class p_1, three schemes:



$$\begin{cases} r_{ij} = \frac{p_i}{p_i+p_j} + 0.1 z_{ij} & \text{if } i > j, \\ r_{ji} = 1 - r_{ij} & \text{if } j > i, \end{cases}$$

$$z_{ij} : \text{Normal}(0, 1)$$

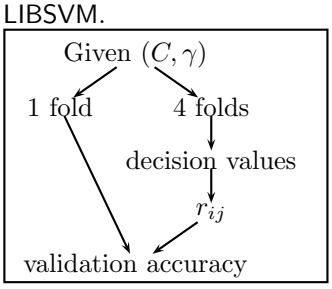
- (a) Small difference between p_i
  (b) Larger difference between $p_i \implies p_i + p_j \not\approx k/2$
  (c) Huge differences between p_i, extreme case of (b)
- Figure 1 :
  δ_1 and δ_2 less sensitive to p_i; overall fairly stable
  Agree with the relation from optimization formulations
- δ_v performs worse, especially MSE as $p_i \le 2/k, \forall i$
- δ_{HT}: Performance (c)<(b)<(a), since $p_i + p_j \approx k/2$ increasingly violated

Real Data

- Small (300 training and 500 testing) and Large (800 and 1000) data sets;
  Each problem : 20 training/testing pairs

dataset	dna	waveform	MNIST	letter
#class	3	3	10	26
#attribute	180	21	784	16

- SVM as binary classifier
  - RBF kernel $e^{-\gamma \|x_i - x_j\|^2}$
  - Parameter selection: five-fold CV on (C, γ) by LIBSVM.



  - Platt's probability estimates for r_{ij}:

$$r_{ij} = P(i \mid i \text{ or } j, \mathbf{x}) = \frac{1}{1 + e^{A\hat{f}+B}}$$

    A and B: minimizing the negative log-likelihood
- Random forest as binary classifier
  CV to select best mtry among $\{1, \sqrt{m}, m/3, m/2, m\}$

Real Data: Results

- k small: four methods similar
- k large: δ_{HT} less competitive
  δ_V fairly good accuracy, but bad in MSE
  Characteristics of real problems here closer to simulation Figure 1(c),(f), rather than Figure 1(a),(d)
- When k large, δ_2 comes closer to δ_V
  δ_1 and δ_2 in the between among four methods.
- Random forest similar

Conclusions

- Four methods connected by optimization formulations
- HT and Voting are extreme cases
  Two proposed methods are in the between $\implies$ tend to be more stable