From Ordinal Ranking to Binary Classification

Hsuan-Tien Lin

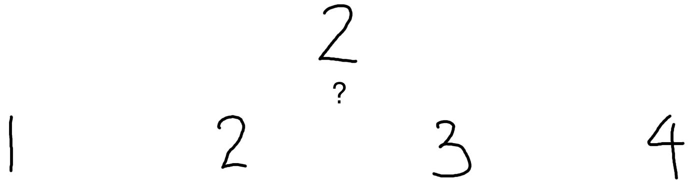
Department of Computer Science and Information Engineering
National Taiwan University

Talk at NTUST
March 16, 2009

Joint work with Dr. Ling Li at Caltech (ALT'06, NIPS'06)

Outline

Which Digit Did You Write?



one (1) two (2) three (3) four (4)

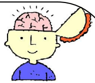
How can machines learn to classify?

Supervised Machine Learning from Examples

Parent

(picture, label) pairs

Kid's brain



$\rightarrow$ good decision function

possibilities

Truth $f(x)$ + noise $e(x)$

examples (picture x_n, label y_n)

learning algorithm

$\rightarrow$ good decision function $g(x) \approx f(x)$

learning model $\{g_\alpha(x)\}$

challenge:

see only $\{(x_n, y_n)\}$ without knowing $f(x)$ or $e(x)$

$\stackrel{?}{\Longrightarrow}$ **generalize** to unseen (x, y) w.r.t. $f(x)$

Some Classical Machine Learning Problems

- classification: discrete y_n
  - $\{\text{one}, \text{two}, \text{three}, \text{four}\}$
  - $\{\text{apple}, \text{orange}, \text{banana}\}$
  - $\{\text{yes}, \text{no}\}$: **binary classification**
- regression: numerical y_n ($\in \mathbb{R}$)
  - stock prices
  - students' scores

Truth $f(x)$ + noise $e(x)$

↓

examples (picture x_n, label y_n)

↓

learning algorithm  → good decision function $g(x) \approx f(x)$

↑

learning model $\{g_\alpha(x)\}$

new types of machine learning problems keep coming from new applications

Outline

Which Age-Group?

2

infant (1)

child (2)

teen (3)

adult (4)

rank: a finite ordered set of labels $\mathcal{Y} = \{1, 2, \cdots, K\}$

Properties of Ordinal Ranking (1/2)

ranks represent **order** information

	<		<		<	
infant (1)		child (2)		teen (3)		adult (4)

general classification cannot properly use order information

Hot or Not?

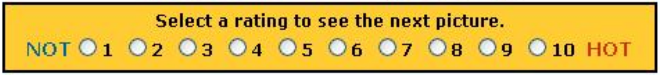
http://www.hotornot.com

rank: natural representation of human preferences

Properties of Ordinal Ranking (2/2)

ranks do **not** carry numerical information

- rating 9 not 2.25 times "hotter" than rating 4



- actual metric hidden

infant (ages 1–3)

child (ages 4–12)

teen (ages 13–19)

adult (ages 20–)

general regression deteriorates without correct numerical information

How Much Did You Like These Movies?

http://www.netflix.com

goal: use “movies you’ve rated” to automatically predict your preferences (ranks) on future movies

Ordinal Ranking Setup

Given

N examples (input x_n, rank y_n) $\in \mathcal{X} \times \mathcal{Y}$

- `age-group`: $\mathcal{X}$ = encoding(human pictures), $\mathcal{Y} = \{1, \cdots, 4\}$
- `hotornot`: $\mathcal{X}$ = encoding(human pictures), $\mathcal{Y} = \{1, \cdots, 10\}$
- `netflix`: $\mathcal{X}$ = encoding(movies), $\mathcal{Y} = \{1, \cdots, 5\}$

Goal

an ordinal ranker (decision function) $r(x)$ that "closely predicts" the ranks y associated with some **unseen** inputs x

ordinal ranking: a hot and important research problem

Importance of Ordinal Ranking

- relatively new for machine learning
- connecting classification and regression
- matching human preferences—many applications in social science, information retrieval, psychology, and recommendation systems

Ongoing Heat: Netflix Million Dollar Prize

Ongoing Heat: Netflix Million Dollar Prize (since 10/2006)

Given

each user u (480,189 users) rates N_u (from tens to thousands) movies x—a total of $\sum_u N_u = 100{,}480{,}507$ examples

Goal

personalized ordinal rankers $r_u(x)$ evaluated on 2,817,131 "unseen" queries (u, x)

Leaderboard

Display top 3 leaders.

Rank	Team Name	Best Score	% Improvement	Last Submit Time
--	No Grand Prize candidates yet	--	--	--
Grand Prize - RMSE <= 0.8563				
1	When Gravity and Dinosaurs Unite	0.8686	8.70	2008-02-12 12:03:24
2	BellKor	0.8686	8.70	2008-02-26 23:26:28
3	Gravity	0.8708	8.47	2008-02-06 14:12:44

the first team being 10% **better than original Netflix system gets a million USD**

Formalizing (Non-)Closeness: Cost

- ranks carry no numerical information: how to say "close"?
- artificially quantify the **cost** of being wrong

  e.g. loss of customer loyalty when the system says ★★★★★ but you feel ★★☆☆☆

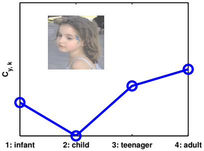
- cost vector **c** of example $(x, y, \mathbf{c})$:
  $\mathbf{c}[k]$ = cost when predicting (x, y) as rank k
  e.g. for (`Sweet Home Alabama` , ★★☆☆☆), a proper cost is $\mathbf{c} = (1, 0, 2, 10, 15)$

closely predict: small cost during testing

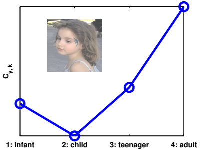
Ordinal Cost Vectors

For an ordinal example $(x, y, \mathbf{c})$, the cost vector $\mathbf{c}$ should

- be consistent with rank y: $\mathbf{c}[y] = \min_k \mathbf{c}[k] \, (= 0)$
- respect order information: V-shaped (**ordinal**) or even convex (**strongly ordinal**)



V-shaped: pay more when predicting further away



convex: pay **increasingly** more when further away

$\mathbf{c}[k] = [\![y \neq k]\!]$	$\mathbf{c}[k] = \lvert y - k \rvert$	$\mathbf{c}[k] = (y - k)^2$
classification: ordinal	absolute: strongly ordinal	squared: strongly ordinal
$(1, 0, 1, 1, 1)$	$(1, 0, 1, 2, 3)$	$(1, 0, 1, 4, 9)$

Our Contributions

a theoretical and algorithmic foundation of ordinal ranking, which reduces ordinal ranking to binary classifcaction, and ...

- provides a methodology for designing new ordinal ranking algorithms with **any** ordinal cost **effortlessly**
- takes many existing ordinal ranking algorithms as **special cases**
- introduces **new theoretical guarantee** on the generalization performance of ordinal rankers
- leads to **superior experimental results**

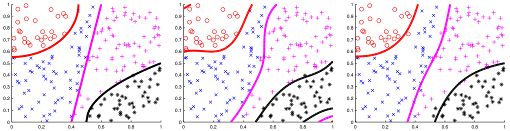


Figure: truth; traditional algorithm; our algorithm

Central Idea: Reduction

(iPod)

complex ordinal ranking problems

(adapter)

(reduction)

(cassette player)

simpler binary classification problems with well-known results on models, algorithms, and theories

If I have seen further it is by standing on the shoulders of Giants—I. Newton

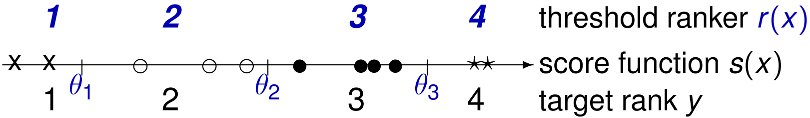
Outline

Outline

1. Machine Learning Setup
2. Ordinal Ranking Setup
3. **The Reduction Framework**
   - **Key Ideas**
   - Important Properties
   - Algorithmic Usefulness
   - Theoretical Usefulness
4. Experimental Results
5. Conclusion

Threshold Ranker

- if getting an ideal score $s(x)$ of a movie x, how to construct the discrete $r(x)$ from an analog $s(x)$?



quantize $s(x)$ by **ordered** (non-uniform) thresholds θ_k

- commonly used in previous work:
  - threshold perceptrons (PRank, Crammer and Singer, 2002)
  - threshold hyperplanes (SVOR, Chu and Keerthi, 2005)
  - threshold ensembles (ORBoost, Lin and Li, 2006)

threshold ranker: $r(x) = \min\{k\colon s(x) < \theta_k\}$

Key Idea: Associated Binary Queries

getting the rank using a threshold ranker

1. is $s(x) > \theta_1$? Yes
2. is $s(x) > \theta_2$? No
3. is $s(x) > \theta_3$? No
4. is $s(x) > \theta_4$? No

generally, how do we query the rank of a movie x?

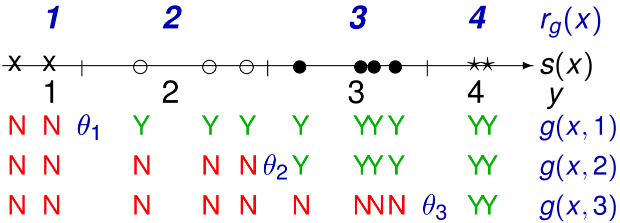
1. is movie x better than rank 1? Yes
2. is movie x better than rank 2? No
3. is movie x better than rank 3? No
4. is movie x better than rank 4? No

associated binary queries:

is movie x better than rank k?

More on Associated Binary Queries

say, the machine uses $g(x, k)$ to answer the query
"*is movie x better than rank k?*"
e.g. for threshold ranker: $g(x, k) = \text{sign}(s(x) - \theta_k)$



associated binary examples:

$$\left(\underbrace{(x, k)}_{k\text{-th associated binary query}} , \underbrace{(z)_k}_{\text{desired answer}} \right)$$

Computing Ranks from Associated Binary Queries

when $g(x, k)$ answers "*is movie x better than rank k?*"

Consider $\big(g(x, 1), g(x, 2), \cdots, g(x, K-1)\big)$,

- consistent predictions: (Y, Y, N, N, N, N, N)
- extracting the rank from consistent predictions:
  - minimum index searching: $r_g(x) = \min\{k \colon g(x, k) = \text{N}\}$
  - counting: $r_g(x) = 1 + \sum_k [\![g(x, k) = \text{Y}]\!]$
- two approaches equivalent for consistent predictions
- mistaken/inconsistent predictions? e.g. (Y, N, Y, Y, N, N, Y)

counting: simpler to analyze and robust to mistake

The Counting Approach

Say $y = 5$, i.e., $\big((z)_1, (z)_2, \cdots, (z)_7\big) = $ (Y, Y, Y, Y, N, N, N)

- if $g_1(x, k)$ reports consistent predictions (Y, Y, N, N, N, N, N)
  - $g_1(x, k)$ made 2 binary classification errors
  - $r_{g_1}(x) = 3$ by counting: the absolute cost is 2

absolute cost = # of binary classification errors

- if $g_2(x, k)$ reports inconsistent predictions (Y, N, Y, Y, N, N, Y)
  - $g_2(x, k)$ made 2 binary classification errors
  - $r_{g_2}(x) = 5$ by counting: the absolute cost is 0

absolute cost $\leq$ # of binary classification errors

If $(z)_k$ = desired answer & r_g computed by counting,

$$\big|y - r_g(x)\big| \leq \sum_{k=1}^{K-1} [\![(z)_k \neq g(x, k)]\!] .$$

Binary Classification Error v.s. Ordinal Ranking Cost

Say $y = 5$, i.e., $\big((z)_1, (z)_2, \cdots, (z)_7\big) = $ (Y, Y, Y, Y, N, N, N)

- if $g_1(x, k)$ reports consistent predictions (Y, Y, N, N, N, N, N)
  - $g_1(x, k)$ made 2 binary classification errors
  - $r_{g_1}(x) = 3$ by counting: the **squared** cost is 4
- if $g_3(x, k)$ reports consistent predictions (Y, N, N, N, N, N, N)
  - $g_3(x, k)$ made 3 binary classification errors
  - $r_{g_3}(x) = 2$ by counting: the **squared** cost is 9

1 error in binary classification
$\Longrightarrow$ 5 cost in ordinal ranking

Importance of Associated Binary Examples

$(z)_k$	Y	Y	Y	Y	N	N	N	
$g_1(x,k)$	Y	Y	N	N	N	N	N	$\mathbf{c}\left[r_{g_1}(x)\right] = \mathbf{c}[3] = 4$
$g_3(x,k)$	Y	N	N	N	N	N	N	$\mathbf{c}\left[r_{g_3}(x)\right] = \mathbf{c}[2] = 9$
$(w)_k$	7	**5**	3	1	1	3	5	

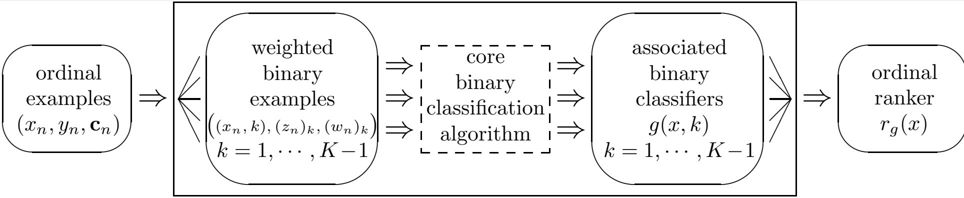
- $(w)_k \equiv \Big|\mathbf{c}[k+1] - \mathbf{c}[k]\Big|$: the importance of $\big((x,k),(z)_k\big)$

per-example cost bound (Li and Lin, 2007):
for **consistent predictions** or **strongly ordinal costs**

$$\mathbf{c}\left[r_g(x)\right] \le \sum_{k=1}^{K-1} (w)_k \left[\!\left[(z)_k \neq g(x,k)\right]\!\right]$$

Outline

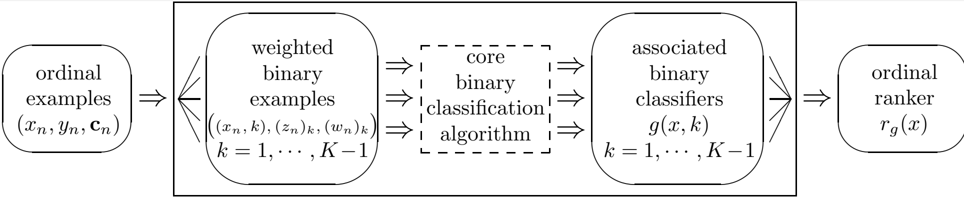
The Reduction Framework (1/2)



1. transform ordinal examples $(x_n, y_n, \mathbf{c}_n)$ to weighted binary examples $\big((x_n, k), (z_n)_k, (w_n)_k\big)$
2. use your favorite algorithm on the weighted binary examples and get $K-1$ binary classifiers (i.e., one big joint binary classifier) $g(x, k)$
3. for each new input x, predict its rank using $r_g(x) = 1 + \sum_k [\![g(x, k) = \mathrm{Y}]\!]$

the reduction framework:
systematic & easy to implement

The Reduction Framework (2/2)



- **performance guarantee**:
  accurate binary predictions $\Longrightarrow$ correct ranks
- **wide applicability**:
  works with any ordinal **c** & any binary classification algorithm
- **simplicity**:
  mild computation overheads with $O(NK)$ binary examples
- **state-of-the-art**:
  allows new improvements in binary classification to be immediately inherited by ordinal ranking

Theoretical Guarantees of Reduction (1/3)

1. absolutely good binary classifier
   $\Longrightarrow$ absolutely good ranker? **YES!**

error transformation theorem (Li and Lin, 2007)

For **consistent predictions** or **strongly ordinal costs**,
if g makes test error Δ in the induced binary problem,
then r_g pays test cost at most Δ in ordinal ranking.

- a one-step extension of the per-example cost bound
- conditions: general and minor
- performance guarantee in the absolute sense

what if no "absolutely good" binary classifier?

Theoretical Guarantees of Reduction (2/3)

1. absolutely good binary classifier
   $\Longrightarrow$ absolutely good ranker? **YES!**
2. relatively good binary classifier
   $\Longrightarrow$ relatively good ranker? **YES!**

regret transformation theorem (Lin, 2008)

For **consistent predictions** or **strongly ordinal costs**,
if g is ϵ-close to the optimal binary classifier g_*,
then r_g is ϵ-close to the optimal ranker r_*.

"reduction to binary" sufficient for algorithm design,
but necessary?

Theoretical Guarantees of Reduction (3/3)

1. absolutely good binary classifier
   $\Longrightarrow$ absolutely good ranker? **YES!**
2. relatively good binary classifier
   $\Longrightarrow$ relatively good ranker? **YES!**
3. algorithm producing relatively good binary classifier
   $\Longleftrightarrow$ algorithm producing relatively good ranker? **YES!**

equivalence theorem (Lin, 2008)

For a general family of **ordinal costs**,
a good ordinal ranking algorithm exists
if & only if a good binary classification algorithm exists
for the corresponding learning model.

ordinal ranking is **equivalent to** binary classification

Outline

Unifying Existing Algorithms

ordinal ranking = reduction + cost + binary classification

ordinal ranking	cost	binary classification algorithm
PRank (Crammer and Singer, 2002)	absolute	modified perceptron rule
kernel ranking (Rajaram et al., 2003)	classification	modified hard-margin SVM
SVOR-EXP SVOR-IMC (Chu and Keerthi, 2005)	classification absolute	modified soft-margin SVM modified soft-margin SVM
ORBoost-LR ORBoost-All (Lin and Li, 2006)	classification absolute	modified AdaBoost modified AdaBoost

- development and implementation time could have been saved
- algorithmic structure revealed (SVOR, ORBoost)

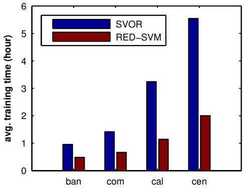
variants of existing algorithms can be designed quickly by tweaking reduction

Designing New Algorithms Effortlessly

ordinal ranking = reduction + cost + binary classification

ordinal ranking	cost	binary classification algorithm
RED-SVM	absolute	standard soft-margin SVM
RED-C4.5	absolute	standard C4.5 decision tree
(Li and Lin, 2007)		

SVOR (modified SVM) v.s. RED-SVM (standard SVM):



advantages of core binary classification algorithm inherited in the new ordinal ranking one

Outline

Proving New Generalization Theorems

Ordinal Ranking (Li and Lin, 2007)

For RED-SVM/SVOR, with pr. $> 1 - \delta$,

$$\begin{aligned} &\text{expected test cost of } r \\ \le\quad & \underbrace{\frac{\beta}{N} \sum_{n=1}^{N} \sum_{k=1}^{K-1} [\![\bar{\rho}(r(x_n), y_n, k) \le \Phi]\!]}_{\substack{\text{ambiguous training} \\ \text{predictions w.r.t.} \\ \text{criteria } \Phi}} \\ +\quad & \underbrace{O\left(\text{poly}\left(K, \frac{\log N}{\sqrt{N}}, \frac{1}{\Phi}, \sqrt{\log \frac{1}{\delta}}\right)\right)}_{\substack{\text{deviation that decreases} \\ \text{with stronger criteria or} \\ \text{more examples}}} \end{aligned}$$

Bi. Cl. (Bartlett and Shawe-Taylor, 1998)

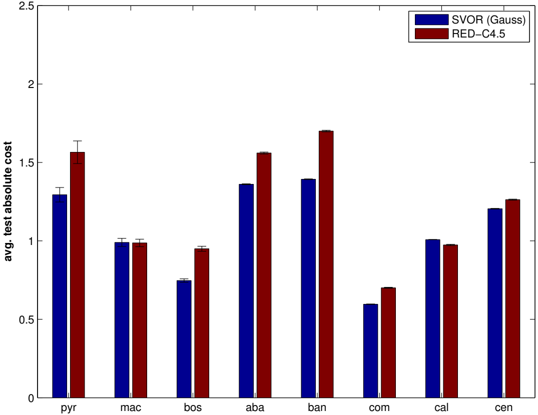
For SVM, with pr. $> 1 - \delta$,

$$\begin{aligned} &\text{expected test err. of } g \\ \le\quad & \underbrace{\frac{1}{N} \sum_{n=1}^{N} [\![\bar{\rho}(g(x_n), y_n) \le \Phi]\!]}_{\substack{\text{ambiguous training} \\ \text{predictions w.r.t.} \\ \text{criteria } \Phi}} \\ +\quad & \underbrace{O\left(\text{poly}\left(\frac{\log N}{\sqrt{N}}, \frac{1}{\Phi}, \sqrt{\log \frac{1}{\delta}}\right)\right)}_{\substack{\text{deviation that decreases} \\ \text{with stronger criteria or} \\ \text{more examples}}} \end{aligned}$$

new ordinal ranking theorem
= reduction + any cost + bin. thm. + math derivation

Outline

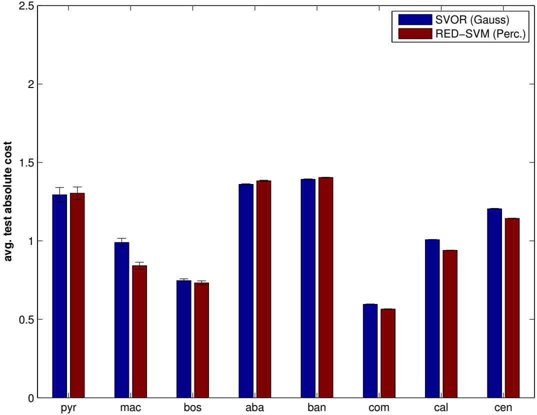
Reduction-C4.5 v.s. SVOR



- C4.5: a (too) simple binary classifier —decision trees
- SVOR: state-of-the-art ordinal ranking algorithm

even simple Reduction-C4.5 sometimes beats SVOR

Reduction-SVM v.s. SVOR



- SVM: one of the most powerful binary classification algorithm
- SVOR: state-of-the-art ordinal ranking algorithm extended from modified SVM

Reduction-SVM without modification often better than SVOR and faster

Outline

1. Machine Learning Setup
2. Ordinal Ranking Setup
3. The Reduction Framework
   - Key Ideas
   - Important Properties
   - Algorithmic Usefulness
   - Theoretical Usefulness
4. Experimental Results
5. **Conclusion**

Conclusion

- reduction framework: simple but useful
  - **establish** equivalence to binary classification
  - **unify** existing algorithms
  - **simplify** design of new algorithms
  - **facilitate** derivation of new theoretical guarantees
- **superior** experimental results:
  better performance and faster training time

reduction keeps ordinal ranking up-to-date with binary classification