From Ordinal Ranking to Binary Classification

Hsuan-Tien Lin

Department of Computer Science and Information Engineering
National Taiwan University

Talk at Microsoft Research Asia
February 18, 2009

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

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

How Much Did You Like These Movies?

http://www.netflix.com

rank: natural representation of human preferences

Properties of Ordinal Ranking (2/2)

ranks do **not** carry numerical information

- ★★★★★ not 2.5 times “better” than ★★☆☆☆
- actual metric may be hidden

infant
(ages 1–3)

child
(ages 4–12)

teen
(ages 13–19)

adult
(ages 20–)

general regression deteriorates without correct numerical information

Ordinal Ranking

Setup

input space $\mathcal{X}$; rank space $\mathcal{Y}$ (a finite ordered set)

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

Given

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

Goal

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

How to say closely predict?

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 ★★☆☆☆

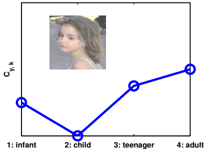
- cost vector $\mathbf{c}$ of example $(x, y, \mathbf{c})$:
  $\mathbf{c}[k]$ = cost when predicting (x, y) as rank k
  e.g. for (`Sweet Home Alabama` , ★★☆☆☆), a proper cost is $\mathbf{c} = (1, 0, 2, 10, 15)$

closely predict: small cost during testing

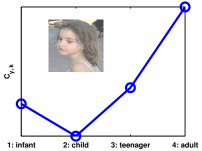
Ordinal Cost Vectors

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

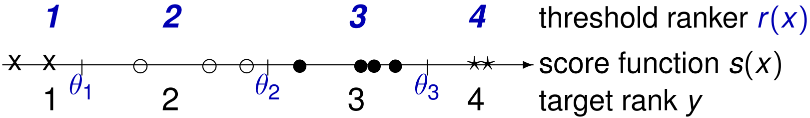
a theoretical and algorithmic foundation of ordinal ranking, which reduces ordinal ranking to binary classifaction, and ...

- 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**

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

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?

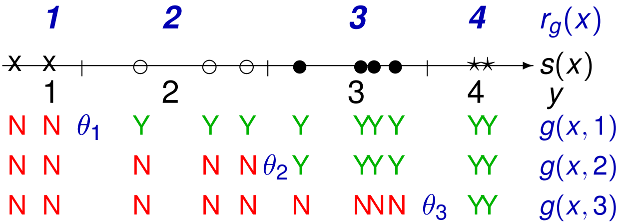
1. is movie x better than rank 1? Yes
2. is movie x better than rank 2? No
3. is movie x better than rank 3? No
4. is movie x better than rank 4? No

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

are all associated examples of the same importance?

Importance of Associated Binary Examples

- given movie x with rank $y = 2$, and $\mathbf{c} = (y - k)^2$

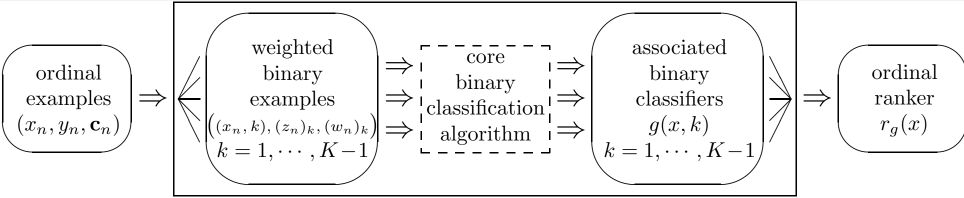
	g_1	g_2	g_3	g_4
is x better than rank 1?	N	Y	Y	Y
is x better than rank 2?	N	N	Y	Y
is x better than rank 3?	N	N	N	Y
is x better than rank 4?	N	N	N	N
$r_g(x)$	1	2	3	4
$\mathbf{c}\left[r_g(x)\right]$	1	0	1	4

- 3 more for answering query 3 wrong; only 1 more for answering query 1 wrong
- $(w)_k \equiv \Big|\mathbf{c}[k+1] - \mathbf{c}[k]\Big|$: the importance of $\big((x, k), (z)_k\big)$

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

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

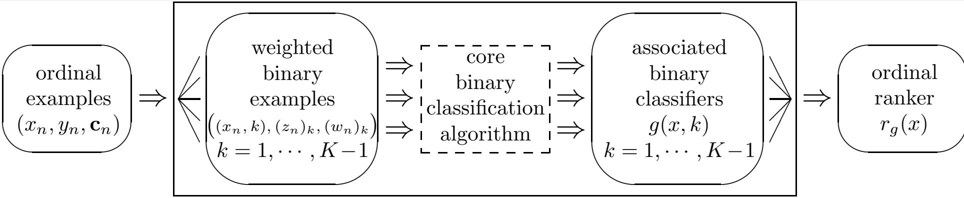
The Reduction Framework (1/2)



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 $r_g(x) = 1 + \sum_k [\![g(x, k) = \text{Y}]\!]$

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

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)

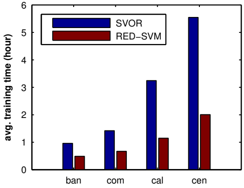
variants of existing algorithms can be designed quickly by tweaking reduction

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

Proving New Generalization Theorems

Ordinal Ranking (Li and Lin, 2007)

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

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

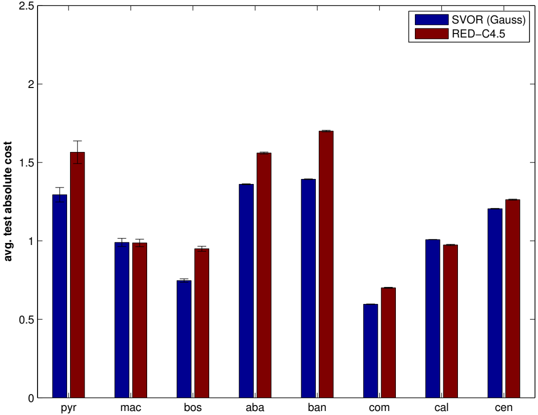
Bi. Cl. (Bartlett and Shawe-Taylor, 1998)

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

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

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

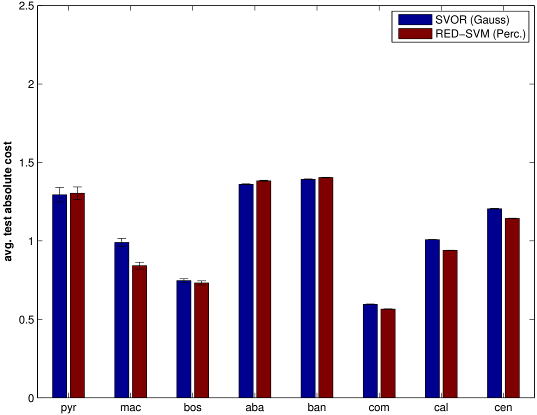
Reduction-C4.5 v.s. SVOR



- 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

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