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)
The Ordinal Ranking Problem

Which Age-Group?

rank: a finite ordered set of labels \( Y = \{1, 2, \cdots, K\} \)
The Ordinal Ranking Problem

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.
The Ordinal Ranking Problem

How Much Did You Like These Movies?

http://www.netflix.com

Get Recommendations (27) Rate Movies Movies You've Rated (5)

How much did you like these movies?

The Wedding Planner

How to Lose a Guy in 10 Days

Sweet Home Alabama

Pretty Woman

rank: natural representation of human preferences
The Ordinal Ranking Problem

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.
The Ordinal Ranking Problem

Ordinal Ranking

Setup

- input space $\mathcal{X}$; rank space $\mathcal{Y}$ (a finite ordered set)
  - age-group: $\mathcal{X} = \text{encoding(human pictures)}$, $\mathcal{Y} = \{1, \ldots, 4\}$
  - netflix: $\mathcal{X} = \text{encoding(movies)}$, $\mathcal{Y} = \{1, \ldots, 5\}$

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?
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 $(x, y, c)$: 
  $c[k] =$ cost when predicting $(x, y)$ as rank $k$

- e.g. for (Sweet Home Alabama, ★★★★★), a proper cost is $c = (1, 0, 2, 10, 15)$

- closely predict: small cost during testing
For an ordinal example \((x, y, c)\), the cost vector \(c\) should

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

\[
\begin{align*}
c[k] &= [y \neq k] \\
\text{classification:} & \quad \text{ordinal} \\
(1, 0, 1, 1, 1) \\
\end{align*}
\]

\[
\begin{align*}
c[k] &= |y - k| \\
\text{absolute:} & \quad \text{strongly ordinal} \\
(1, 0, 1, 2, 3) \\
\end{align*}
\]

\[
\begin{align*}
c[k] &= (y - k)^2 \\
\text{squared:} & \quad \text{strongly ordinal} \\
(1, 0, 1, 4, 9) \\
\end{align*}
\]
Our Contributions

- a theoretical and algorithmic foundation of ordinal ranking, which reduces ordinal ranking to binary classification, 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
Reduction from Ordinal Ranking to Binary Classification

**Key Ideas**

**Threshold Ranker**

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

\[
\begin{align*}
&\text{quantize } s(x) \text{ by ordered (non-uniform) thresholds } \theta_k \\
\text{threshold ranker: } r(x) = \min \{ k : s(x) < \theta_k \}
\end{align*}
\]

- 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)
Reduction from Ordinal Ranking to Binary Classification

Key Ideas

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 $k$?
say, the machine uses $g(x, k)$ to answer the query “is movie $x$ better than rank $k$?”
e.g. for threshold ranker: $g(x, k) = \text{sign}(s(x) - \theta_k)$
Computing Ranks from Associated Binary Queries

when \( g(x, k) \) answers “is movie \( x \) better than rank \( k \)?”

Consider \((g(x, 1), g(x, 2), \ldots, g(x, K-1))\),

- consistent predictions: \((Y, Y, N, N, N, N, N)\)
- extracting the rank from consistent predictions:
  - minimum index searching: \( r_g(x) = \min \{ k : g(x, k) = N \} \)
  - counting: \( r_g(x) = 1 + \sum_k [g(x, k) = Y] \)
- 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 \( c = (y - k)^2 \)

<table>
<thead>
<tr>
<th>Is ( x ) better than rank 1?</th>
<th>N</th>
<th>Y</th>
<th>Y</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is ( x ) better than rank 2?</td>
<td>N</td>
<td>N</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>Is ( x ) better than rank 3?</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>Is ( x ) better than rank 4?</td>
<td>N</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

\[
\begin{array}{ccccc}
\text{\( r_g(x) \)} & 1 & 2 & 3 & 4 \\
\text{\( c \[ r_g(x) \] } & 1 & 0 & 1 & 4 \\
\end{array}
\]

- 3 more for answering query 3 wrong; 1 more for answering query 1 wrong

- \( (w)_k \equiv |c[k + 1] - c[k]| \): the importance of \((x, k), (z)_k\)

Per-example cost bound (Li and Lin, 2007):

For consistent predictions or strongly ordinal costs:

\[
c \left[ r_g(x) \right] \leq \sum_{k=1}^{K-1} (w)_k \; [ (z)_k \neq g(x, k)]
\]
The Reduction Framework (1/2)

1. Transform ordinal examples \((x_n, y_n, c_n)\) to weighted binary examples \(((x_n, k), (z_n)_k, (w_n)_k)\) where \(k = 1, \ldots, K-1\).

2. Use your favorite algorithm on the weighted binary examples and get \(K-1\) binary classifiers (i.e., one big joint binary classifier) \(g(x, k)\) where \(k = 1, \ldots, K-1\).

3. For each new input \(x\), predict its rank using \(r_g(x) = 1 + \sum_k [g(x, k) = Y]\).

The reduction framework:
- systematic & easy to implement
The Reduction Framework (2/2)

- **performance guarantee:**
  accurate binary predictions $\Rightarrow$ 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**
   \[ \implies \text{Absolutely good ranker? YES!} \]

**Error transformation theorem** (Li and Lin, 2007)

For **consistent predictions** or **strongly ordinal costs**, if \( g \) makes test error \( \Delta \) in the induced binary problem, then \( r_g \) pays test cost at most \( \Delta \) 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
   \[\implies\] absolutely good ranker? **YES!**

2. relatively good binary classifier
   \[\implies\] relatively good ranker? **YES!**

---

**regret transformation theorem** (Lin, 2008)

For *consistent predictions* or *strongly ordinal costs*,
if \(g\) is \(\epsilon\)-close to the optimal binary classifier \(g^*\),
then \(r_g\) is \(\epsilon\)-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
   \[\implies\] absolutely good ranker? \textbf{YES!}

2. relatively good binary classifier
   \[\implies\] relatively good ranker? \textbf{YES!}

3. algorithm producing relatively good binary classifier
   \[\iff\] algorithm producing relatively good ranker? \textbf{YES!}

**equivalence theorem** (Lin, 2008)

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

\textbf{ordinal ranking is equivalent to} binary classification
Unifying Existing Algorithms

 ordinal ranking = reduction + cost + binary classification

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

- 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

Algorithmic Usefulness

Reduction from Ordinal Ranking to Binary Classification

ordinal ranking = reduction + cost + binary classification

<table>
<thead>
<tr>
<th>ordinal ranking</th>
<th>cost</th>
<th>binary classification algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>RED-SVM</td>
<td>absolute</td>
<td>standard soft-margin SVM</td>
</tr>
<tr>
<td>RED-C4.5</td>
<td>absolute</td>
<td>standard C4.5 decision tree</td>
</tr>
<tr>
<td>(Li and Lin, 2007)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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. \( > 1 - \delta \),

\[
\text{expected test cost of } r \leq \frac{\beta}{N} \sum_{n=1}^{N} \sum_{k=1}^{K-1} \left[ \bar{\rho}(r(x_n), y_n, k) \leq \Phi \right]
\]

+ \( O \left( \text{poly} \left( K, \frac{\log N}{\sqrt{N}}, \frac{1}{\Phi}, \sqrt{\log \frac{1}{\delta}} \right) \right) \)

ambiguous training predictions w.r.t. criteria \( \Phi \)

deviation that decreases with stronger criteria or more examples

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

For SVM, with pr. \( > 1 - \delta \),

\[
\text{expected test err. of } g \leq \frac{1}{N} \sum_{n=1}^{N} \left[ \bar{\rho}(g(x_n), y_n) \leq \Phi \right]
\]

+ \( O \left( \text{poly} \left( \frac{\log N}{\sqrt{N}}, \frac{1}{\Phi}, \sqrt{\log \frac{1}{\delta}} \right) \right) \)

ambiguous training predictions w.r.t. criteria \( \Phi \)

deviation that decreases with stronger criteria or more examples

new ordinal ranking theorem

= reduction + any cost + bin. thm. + math derivation
Experimental Results

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
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