Imbalanced RankBoost for Efficiently Ranking Large-Scale Image/Video Collections

Michele Merler†, Rong Yan* and John R. Smith*
†Columbia University, *IBM TJ Watson Research Center

Email: mmerler@cs.columbia.edu, {yanr, jsmith}@us.ibm.com

Introduction

- Efficiently ranking and retrieving relevant data is increasingly important for organizing / managing large-scale image/video collections
- Vast majority of users prefer to browse only a limited number of top ranked examples, while completely ignore the rest
- Imbalanced RankBoost (rank learning algorithm)
  - merges RankBoost and iterative thresholding into unified loss optimization framework
  - distinguishes between top and bottom ranked data

Learning to Rank

- Goal: produce a ranked list of originally unordered examples $X$, so that the relevant ones are placed as close as possible to the top
- RankBoost
  - Combine a pool of K “weak” ranking features $h_i(x)$ into composite ranking function $F(x)$
  - Exponential loss function minimization (~minimize # of ranking errors)
  - Joint selection of $h_i(x)$ and optimization of combination weights $\alpha_k$

No distinction between top- and bottom-ranked examples

Problem: Suboptimal Ranking Process

Solution: Imbalanced RankBoost

- More ranking features for top ranked data
- Truncate ranking feature computation for the data ranked below learnt cutoff thresholds
- More efficient ranking process

Imbalanced RankBoost

- Bipartite setting: relevant and non-relevant examples $x$ are grouped in disjoint sets $X_0$ and $X_1$
- Ranking function $F(x)$ contains cut-off thresholds $\theta_i$
  \[ F(x) = f(x) + \sum_i \alpha_i h_i(x) \cdot I(F_i(x) \leq \theta_i) \]
- Goal: minimize exponential rank misclassification error wrt
  \[ L = \sum_i \exp(-f(x_i) - f_0(x_i) + \theta_i h_i(x_i) \cdot I(x_i \in X_0) - \theta_i h_i(x_i) \cdot I(x_i \in X_1) - \lambda \Omega(\alpha)) \]

Learning Process

- Initialize pool of ranking features
- Select optimal ranking feature
- Select optimal cutoff threshold
- Compute optimal weights
- Update ranking function
- Output final ranking function

Imbalanced RankBoost

- Imbalanced RankBoost automatically emphasizes top ranked data and truncates computation for less important bottom-ranked ones
- Ranking process more efficient and effective wrt traditional RankBoost
- Future work: incorporate processing time of ranking features into learning process

Conclusions

Experiments

- TRECVID 2007 collection: 52,347 keyframes, 20 concepts
- 200 Ranking Features: RBF kernel SVM trained on bags of data and global features (color histogram, edge histogram, etc.)

RESULTS

- 6-fold speed up in ranking process
- 7% to 21% relative MAP improvement
- Benefit more relevant on top ranked samples

Average precision for single concepts computed as function of the ranking time at full depth