Colearning

Exploiting Task-Relatedness in the Expert Framework

David Ilstrup

University of California - Santa Cruz

June 11, 2007 - CMPS290C
Colearning Outline

1 Quick Review (lec.5&6)

2 Experimental Setup

3 Experimental Results

4 Summary
Colearning

- Other settings for colearning exist in the literature (we ignore here)
- Focus on colearning in the expert framework
- Use the expert framework to exploit task relatedness
- Same setting as Abernethy, Bartlett and Rakhlin [ABR 2007]
Setting

- $K$ tasks
- $\kappa(t)$ is current task provided to prediction algorithm
- $N$ experts $f_i$ each predict value $f_{i,t} \in [0,1]$ at time $t$
- $\hat{y}_t \in [0,1]$ is algorithm prediction
- $y_t \in \{0,1\}$ is actual (discrete valued) label
- $\ell(pred, y_t)$ is loss - convex in prediction argument
Intuitive Task Relatedness

- Experts do well on more than one task
- Can small set of $m$ experts do (nearly) as well as $K$ “task-specialist” experts?
Formal Task Relatedness

- Let $S_m$ represent the set of all size-$m$ subsets of experts
- Let $L_{i,k}^T = \sum_{t: \kappa(t) = k} \ell(f_i, t, y_t)$

Change notation slightly to make dependence on $m$ explicit

$$L_{m^*}^T = \min_{S \in S_m} \sum_{k=1}^{K} \min_{i \in S} L_{i,k}^T$$

If $L_{m^*}^T$ doesn’t increase (much) as $m$ decreases, we have task relatedness
Method to get a (infeasible) bound - Hyperexperts

- \( \binom{N}{m} \) elements in \( S_m \)
- Can assign to tasks in \( m^K \) ways, but duplicates between subsets
- \( m^K \) assignments without repetition, but too few maps
- So \( \approx m^K \binom{N}{m} \) hyperexperts
- A weight per hyperexpert: bound terms \( \propto K \log(m) + m \log(N/m) \)
- Problem: Too many weights!
Can we achieve the bound with fewer weights?

- For $L_{1*}$ we only need the best expert
- $N$ weights and “static experts” algorithm suffice
- For $m = 1$
  $$K \log(m) + m \log(N/m) = \log(N)$$
- $K \times N$ weights and “static experts” for each task
- For $m = K$
  $$K \log(m) + m \log(N/m) = K \log(N)$$
- Suggestive, but tasks $\neq$ experts!
Terminology [ABR 2007]

Minimum number of shifts

**Definition (sequential multi-task problem)**
Either $\kappa(t+1) = \kappa(t)$, or $\forall s > t : \kappa(s) \neq \kappa(t)$

Maximum number of shifts

**Definition (shifting multi-task problem)**
$\forall t : \kappa(t+1) \neq \kappa(t)$
Algorithm 1: [ABR 2007]

Multitask Mixing Algorithm

1: Input: \( \eta \)
2: \( \tilde{w}_k^0 := \frac{1}{N} \mathbf{1} \) for all \( k \in [N] \)
3: for \( t := 1 \) to \( T \) do
4: \( k := \kappa(t) \), (current task)
5: Choose a distribution \( \beta_t \) over all tasks
6: \( \tilde{z}_t := \sum_{k'=1}^{K} \beta_t(k') \tilde{w}_{k'}^{t-1} \)
7: \( \hat{p}_t = z_t \cdot f_t \)
8: \( \tilde{w}_{k,i}^t := (\tilde{z}_i^t e^{-\eta l_i^t}) / (\sum_{j=1}^{N} \tilde{z}_j^t e^{-\eta l_j^t}) \) for all \( i \in [N] \)
9: \( \tilde{w}_{k'}^t := w_{k'}^{t-1} \) for any \( k' \neq k \)
MTM Achieves bound for sequential multi-task problem

- Bound terms in $O(m \log(N/m) + K \log(m))$
- General bound for MTM depends on number of shifts
- Basic problem: Conceptually problem depends on tasks, but bounds must sum over time
Choose a distribution $\beta_t$ over all tasks

- Can clever choices of $\beta_t$ mixing over tasks instead of priors achieve the desired bounds?
- Do some of the mixing schemes over priors in [BW 2002] already work? (We try some below)
- Optimal mixing parameters can depend on knowledge of $T$ or $m$ (e.g. MTM $\alpha = 1 - 2/m$)
- Can we use schemes similar to the doubling trick to adapt these as we go?

CAUTION: In the fixed share algorithms of [BW 2002], $\alpha$ is the mixing value collectively applied to the past posteriors. In [ABR 2007], $\alpha$ is the mixing value applied to the current task prior.
Colearning Outline

1. Quick Review (lec.5&6)
2. Experimental Setup
3. Experimental Results
4. Summary
Data Generation

Same scheme as [ABR 2007]

- $K = 60$
- Tasks are randomly generated linear classifiers in $\mathbb{R}^4$
- $N = 100$
- 30 Experts are randomly generated linear classifiers
- 70 Experts predict randomly in the unit interval
Experiments

- **Experiment 1 - Sequential Multi-Task Problem**
  - 200 instances of each task presented sequentially
  - Each task presented once, 1200 trials total
- **Experiment 2 - Sequential Multi-Task Problem with Noise**
  - Same as experiment 1, but 30% label noise added to linear classifiers
- **Experiment 3 - Shifting Multi-Task Problem**
  - Task changes on every trial
  - 1200 trials total
- **Experiment 4 - Shifting Multi-Task Problem with Noise**
  - Same as experiment 3, but 30% label noise added to linear classifiers
Comparators and Algorithms

- Comparator: Best expert overall
- Comparator: Best expert at each task
- Algorithm: Static experts
- Algorithm: Fixed Share to Start Vector
- Algorithm: Static experts, per task
- Algorithm: Fixed Share to Start Vector, per task
- Algorithm: Multi-Task Mixing (Algorithm 1 above)
- Algorithm: Fixed Share to Decaying Past
1. Quick Review (lec.5&6)
2. Experimental Setup
3. Experimental Results
4. Summary
Experiment 1

Cumulative Losses for Algorithms and Comparitors

- Best Expert ($L_{1^*}$)
- K-Experts ($L_{60^*}$)
- Static Experts
- Fixed Share to Start Vector
- Static Experts, Per Task Weights
- Fixed Share to Start, Per Task Weights
- MT-Mixing ($\alpha = 0.97$)
- Fixed Share to Decaying Past

$\eta = 0.50, \alpha = 0.01$
Experiment 2

Cumulative Losses for Algorithms and Comparators

- Best Expert ($L_1^*$)
- K-Experts ($L_{60^*}$)
- Static Experts
- Fixed Share to Start Vector
- Static Experts, Per Task Weights
- Fixed Share to Start, Per Task Weights
- MT-Mixing ($\alpha=0.97$)
- Fixed Share to Decaying Past

$\eta = 0.50$, $\alpha = 0.01$
Experiment 3

Cumulative Losses for Algorithms and Comparitors

- Best Expert ($L_{1^*}$)
- K-Experts ($L_{60^*}$)
- Static Experts
- Fixed Share to Start Vector
- Static Experts, Per Task Weights
- Fixed Share to Start, Per Task Weights
- MT-Mixing ($\alpha=0.97$)
- Fixed Share to Decaying Past

$\eta = 0.50$, $\alpha = 0.01$
Experiment 4

Cumulative Losses for Algorithms and Comparitors

- Best Expert ($L_{1^*}$)
- K-Experts ($L_{60^*}$)
- Static Experts
- Fixed Share to Start Vector
- Static Experts, Per Task Weights
- Fixed Share to Start, Per Task Weights
- MT-Mixing ($\alpha=0.97$)
- Fixed Share to Decaying Past

$\eta = 0.50$, $\alpha = 0.01$
Colearning Outline

1. Quick Review (lec.5&6)
2. Experimental Setup
3. Experimental Results
4. Summary
Conclusion

- Algorithms using per task weights can do quite well
- Just a small experiment - what about in general?
- Want bounds with log terms free of dependence on number of task shifts or trials
What next?

- Experiment with generalized mixing schemes, such as MAX update and Projection update (cf. [BW 2002] sec. 5.2)
- Noisy task identification?
- Find mixing algorithm with desired bound
- Try to identify $m$-sized subset of experts?