Boosting with Online Binary Learners for the Multiclass Bandit Problem

Shang-Tse Chen
Hsuan-Tien Lin
Chi-Jen Lu

Georgia Tech National Taiwan University Academia Sinica

June 28, 2012
Multiclass Bandit Problem - Motivation

- online advertising
Multiclass Bandit Problem - Motivation

- online advertising
Multiclass Bandit Problem - Motivation

- online advertising
Multiclass Bandit Problem - Motivation

- online advertising

query → ad

Google

← click or not
Multiclass Bandit Problem - Motivation

- online advertising

- feedbacks for the unchosen ads remain unknown

- exploitation and exploration trade-off
For $t = 1, 2, \ldots, T$

1. Receive $x_t \in \mathbb{R}^d$ (query)

2. Predict $\hat{y}_t \in \{1, \ldots, k\}$ (ad)

3. Pay $1[y_t \neq \hat{y}_t]$ ($y_t$ is not revealed) (click or not)
### Multiclass Bandit Problem - Our Approach

#### Previous works
- Adapt from full-information algorithms
- Most of them are linear models
  - Banditron, Newton, …

#### This work:

Simple Algorithms (Weak Learners) $\rightarrow$ **Boosting** $\rightarrow$ **Strong Bandit Learner**
Brief Review – Batch Boosting

training data

\[(x_1, y_1) \quad (x_2, y_2) \quad \ldots \quad (x_T, y_T)\]
Brief Review – Batch Boosting

Training data

\[ w_1^{(1)} \quad w_2^{(1)} \quad \ldots \quad w_T^{(1)} \]

\[ (x_1, y_1) \quad (x_2, y_2) \quad \ldots \quad (x_T, y_T) \]
Brief Review – Batch Boosting

better than random guessing over $w^{(1)}$

Better than random guessing over $w^{(2)}$

Better than random guessing over $w^{(N)}$

Strong Learner
Brief Review – Batch Boosting

\[ (x_1, y_1) \quad (x_2, y_2) \quad \ldots \quad w_1^{(2)} \quad w_2^{(2)} \ldots \quad w_T^{(2)} \]

\[ (x_T, y_T) \]

\[ \alpha^{(1)} \quad h^{(1)} \]

\[ \alpha^{(1)} \quad h^{(1)} \]

\[ \alpha^{(1)} \quad h^{(1)} \]

\[ \alpha^{(1)} \quad h^{(1)} \]
Brief Review – Batch Boosting

better than random guessing over $w^{(2)}$

$\alpha^{(2)}$ $h^{(2)}$

$\alpha^{(1)}$ $h^{(1)}$

$w^{(2)}_1$ $w^{(2)}_2$ $\cdots$ $w^{(2)}_T$

$w^{(1)}_1$ $w^{(1)}_2$ $\cdots$ $w^{(1)}_T$

$(x_1, y_1)$ $(x_2, y_2)$ $\cdots$ $(x_T, y_T)$

training data
Brief Review – Batch Boosting

\[ \alpha^{(N)} \cdot h^{(N)} \]

\[ w_1^{(N)} \quad w_2^{(N)} \quad \cdots \quad w_T^{(N)} \]

\[ \alpha^{(2)} \cdot h^{(2)} \]

\[ w_1^{(2)} \quad w_2^{(2)} \quad \cdots \quad w_T^{(2)} \]

\[ \alpha^{(1)} \cdot h^{(1)} \]

\[ w_1^{(1)} \quad w_2^{(1)} \quad \cdots \quad w_T^{(1)} \]

\[ (x_1, y_1) \quad (x_2, y_2) \quad \cdots \quad (x_T, y_T) \]

Training data
Brief Review – Batch Boosting

Strong Learner

$\alpha^{(N)} \quad h^{(N)}$

$\alpha^{(2)} \quad h^{(2)}$

$\alpha^{(1)} \quad h^{(1)}$

$w_1^{(N)} \quad w_2^{(N)} \quad \ldots \quad w_T^{(N)}$

$w_1^{(2)} \quad w_2^{(2)} \quad \ldots \quad w_T^{(2)}$

$w_1^{(1)} \quad w_2^{(1)} \quad \ldots \quad w_T^{(1)}$

$(x_1, y_1) \quad (x_2, y_2) \quad \ldots \quad (x_T, y_T)$

training data
At each round $t$:

$$\alpha_t^{(N)}$$

$$h_t^{(N)}$$

$$\alpha_t^{(2)}$$

$$h_t^{(2)}$$

$$\alpha_t^{(1)}$$

$$h_t^{(1)}$$

Goal: minimize error rate:

$$\frac{1}{T} \sum_{t=1}^{T} \left[ H_t(x_t), y_t \right]$$
At each round $t$:
1. receive $x_t$

For each round $t$:
- Receive $x_t$
- Predict $H^t(x_t)$
- Receive $y_t$
- Update $h_t$ and $\alpha_t$ for $\alpha_t^{(1)}$, $\alpha_t^{(2)}$, and $\alpha_t^{(N)}$

Goal: minimize error rate:
$$\frac{1}{T} \sum_{t=1}^{T} \left[ H^t(x_t), y_t \right]$$
Online Boosting (Binary)

At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$

$H_t(x_t) = \text{sign}(\sum_{i=1}^{N} \alpha_t^{(i)} h_t^{(i)}(x_t))$

$w(t) = \alpha_t^{(1)} + \alpha_t^{(2)} + \cdots + \alpha_t^{(N)}$

$t - 1$  \hspace{1cm}  $t$  \hspace{1cm}  $t + 1$
Online Boosting (Binary)

At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$

$$H_t(x_t) = \text{sign} \left( \sum_{i=1}^{N} \alpha_t^{(i)} h_t^{(i)}(x_t) \right)$$
Online Boosting (Binary)

At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$
At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h^{(i)}_t$ and $\alpha^{(i)}_t$

**Online Boosting (Binary)**

$$\begin{align*}
\alpha^{(N)}_t \rightarrow h^{(N)}_t \\
\alpha^{(2)}_t \rightarrow h^{(2)}_t \\
\alpha^{(1)}_t \rightarrow h^{(1)}_t \\
\end{align*}$$

**Online Learning**

$$\begin{align*}
x_t, y_t & \quad w^{(1)}_t = 1 \\
_t & \quad \alpha^{(1)}_t \rightarrow h^{(1)}_{t+1} \\
_t & \quad \alpha^{(1)}_t \rightarrow h^{(1)}_{t+1} \\
_{t+1} & \quad \alpha^{(N)}_t \rightarrow h^{(N)}_t \\
_{t+1} & \quad \alpha^{(2)}_t \rightarrow h^{(2)}_t \\
_{t+1} & \quad \alpha^{(1)}_t \rightarrow h^{(1)}_t \\
\end{align*}$$
Online Boosting (Binary)

At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$

**Online Learning**

- $x_t, y_t$
- $t - 1$, $t$, $t + 1$
At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$

Goal: minimize error rate:

$$\sum_{t=1}^{T} \left[ H_t(x_t), y_t \right]$$
Online Boosting (Binary)

At each round \( t \):
1. receive \( x_t \)
2. predict \( H_t(x_t) \)
3. receive \( y_t \)
4. update each \( h_t^{(i)} \) and \( \alpha_t^{(i)} \)

**Goal**: minimize error rate:
\[
\sum_{t=1}^{T} \left[ H_t(x_t), y_t \right]
\]
At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$
At each round $t$:
1. receive $x_t$
2. predict $H_t(x_t)$
3. receive $y_t$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$

**Goal:** minimize error rate:

$$\frac{1}{T} \sum_{t=1}^{T} \mathbf{1}[H_t(x_t) \neq y_t]$$
At each round $t$:
1. receive $x_t$
2. predict $\hat{y}_t$
3. receive $1[y_t \neq \hat{y}_t]$
4. update each $h_t^{(i)}$ and $\alpha_t^{(i)}$

Challenges in this setting:
- Use a single bit of feedback to
  1. ensure good weak learners (with proper assumption)
  2. estimate example weights
  3. estimate voting weights
Challenge 1 - What is our online weak learner

What kind of weak learners should we use?

- Multiclass Bandit Weak Learner
  - More difficult to design
  - Weak learners are not in the standard bandit setting
    \[ \text{(predict } \hat{y}_t^{(i)} \text{ but receive } 1[y_t \neq \hat{y}_t]) \]

- Binary (full-information) Weak Learner
  - simplest learner, many existing algorithms
  - use one-vs-rest reduction
    \[ (x_t, y_t) \rightarrow ((x_t, k), y_{tk}), \quad k = 1, \ldots, K \]
  - Call the weak learner only when \( y_{tk} \) is known
What is our online weak learner? What kind of weak learners should we use?

- Multiclass Bandit Weak Learner
  - More difficult to design
  - Weak learners are not in the standard bandit setting
    
    \[ \text{(predict } \hat{y}_t^{(i)} \text{ but receive } 1[y_t \neq \hat{y}_t]\) \]

- Binary (full-information) Weak Learner
  - simplest learner, many existing algorithms
  - use one-vs-rest reduction
    
    \[ (x_t, y_t) \rightarrow ((x_t, k), y_{tk}), \quad k = 1, \ldots, K \]
  - Call the weak learner only when \( y_{tk} \) is known
What’s the assumption on the online binary weak learner?

- Better than random guessing under any distribution
  - → IMPOSSIBLE  [Chen et al. ’12]
    - extreme case: only the first example has a non-zero weight
Challenge 1 - What is our online weak learner

What’s the assumption on the online binary weak learner?

- Better than random guessing under any distribution → IMPOSSIBLE [Chen et al. ’12]
  - extreme case: only the first example has a non-zero weight
- Relaxation: only deal with weights that satisfy

\[ \sum_{t,k} w_{t,k} = \Omega\left(\frac{KB}{\gamma^2}\right), \quad \text{where } B \geq \max_{t,k} w_{t,k} \]

---

Batch

- a fixed hypothesis \( h \in \mathcal{H} \)
- advantage \( 2\gamma > 0 \)

Online

- Linear Optimization
- regret = \( \sqrt{\frac{KB}{\sum_{t,k} w_{t,k}}} \)

---

Online

- a sequence of hypotheses \( h_1, ..., h_T \in \mathcal{H} \)
- advantage \( \gamma > 0 \)
Challenge 2 - How to set the example weight

Sum of weights $\sum_{t,k} w_{t,k}$ should be larger than $\Omega(KB/\gamma^2)$

- **Online SmoothBoost** [Chen et al. ’12] satisfies this condition (in the full-information setting). Let $\bar{w}_{tk}^{(i)}$ be such weights.
Challenge 2 - How to set the example weight

Sum of weights $\sum_{t,k} w_{t,k}$ should be larger than $\Omega(KB/\gamma^2)$

- **Online SmoothBoost** [Chen et al. ’12] satisfies this condition (in the full-information setting). Let $\bar{w}_{tk}^{(i)}$ be such weights.
- In the bandit setting, we can estimate it by

$$w_{tk}^{(i)} = \begin{cases} 
\bar{w}_{tk}^{(i)}/p_t(k) & \text{if } \hat{y}_t = y_t \\
0 & \text{otherwise}
\end{cases}$$

- Randomly predict with probability $\delta$ to ensure $|w_{tk}^{(i)}| \leq K/\delta$
- By Azuma’s Inequality

$$Pr \left[ \sum_t \left( \bar{w}_{tk}^{(i)} - w_{tk}^{(i)} \right) > \lambda T \right] \leq 2^{-\Omega(\lambda^2T/B)}$$
Choose the number of weak learners $N$ in the beginning

- using uniform voting weight, we can achieve error rate $O(K\delta)$ by combining at most $O(K/(\delta^2\gamma^2))$ weak learners
- this is only an upper bound of $N$
- we know the existence of a good combination of weak learners

$$\bar{\alpha} = \left(\frac{1}{m}, \ldots, \frac{1}{m}, 0, \ldots, 0\right), \quad m \leq N$$

- idea: use another online learning process to learn $\bar{\alpha}$
Dynamically set voting weights $\alpha_{tk}$ at each round

- Define loss $L_{tk}(\alpha) = \max \left\{ 0, \theta - y_{tk} \sum_{i=1}^{N} \alpha^{(i)} h_{tk}^{(i)}(x_t) \right\}$

- estimate the subgradient $\ell_{tk} = \begin{cases} \nabla L_{tk}(\alpha)/p_t(k) & \text{if } \hat{y}_t = y_t \\ 0 & \text{otherwise} \end{cases}$

- use multiplicative update

$$\alpha^{(i)}_{tk} = \alpha^{(i)}_{tk} \cdot e^{-\eta \ell_{tk}^{(i)}/Z_{(t+1)k}}$$
Put it all together

Main Theorem

Assumption:

- There is an online binary learner which can achieve an advantage $2\gamma > 0$ for any sequence of examples with the sum of weights larger than $\Omega(KB/\gamma^2)$
- $T = \Omega\left(\frac{K^2}{\delta^4} \log(K/\delta)\right)$

Then, the proposed bandit boosting algorithm achieves error rate $O(K\delta/\gamma)$ by using $O(K/\delta^2\gamma^2)$ weak learners.
Experiments

Dataset: **Reuters4**

Stable in the choice of exploration parameter $\delta$
Experiments

Dataset: Reuters4

Outperform other algorithms when $T$ is large
Conclusion

- We propose the first boosting algorithm in the multiclass bandit setting with nice theoretical properties by solving the 3 main challenges.
  - Determine the appropriate weak learner and assumption
  - Estimate example weights based on limited feedback
  - Combining weak learners based on limited feedback
- Promising empirical results on real-world data sets

Thank you. Questions?