CS 596: Theoretical Machine Learning

Lecturer: Pranjal Awasthi
Scribe: Hui Qu

Lecture # 6
Oct 6, 2016

1 Randomized Multiplicative Weights (RMW) algorithm

In last lecture, we discussed the bound on the number of mistakes using Multiplicative Weights (MW). This bound can be improved using Randomized Multiplicative Weights (RMW.) The procedure of RMW is shown in algorithm 1, and it has two differences compared with MW:

- Treat weights as probabilities
- The rule of updating the weights, $w_h^{t+1} \leftarrow (1-\epsilon)w_h^t$

Algorithm 1 Randomized Multiplicative Weights (RMW)

initialize: $w_h^1 = 1$ for all $h \in H$
for $t = 1, 2, \cdots$ **do**
    on input x_t, predict $h(x_t)$ with probability w_h^t
    if h made a mistake **then**
        update the weight with the rule: $w_h^{t+1} \leftarrow (1-\epsilon)w_h^t$

Theorem 1. *Given a function class H, and any sequence of examples $S = \{x_1, x_2, \ldots, x_T\}$, let m be the number of mistakes of the best function in H, and M be the number of mistakes of RMW. Then $E[M] \leq (1+\epsilon)m + \frac{\log(|H|)}{\epsilon}$.*

Proof. Let's track the total weight of the system. W^t is the total weight on time t. Initially, $W^1 = |H|$. Let m be the number of mistakes that the best function in H has made so far $(t = 1, \ldots, T)$. Then

$$W^{T+1} \geq (1-\epsilon)^m \tag{1}$$

Let F^t denote the fraction of weight at time t on functions that make mistakes at time t. Then $W^{t+1} \leq (1-F^t)W^t + F^tW^t(1-\epsilon) = W^t(1-\epsilon F^t)$. Therefore,

$$W^{T+1} \leq |H|(1-\epsilon F^1)(1-\epsilon F^2)\cdots(1-\epsilon F^T) \tag{2}$$

Combining Equations (1) and (2),

$$\begin{aligned}
(1-\epsilon)^m &\leq |H|(1-\epsilon F^1)(1-\epsilon F^2)\cdots(1-\epsilon F^T) \\
m\log(1-\epsilon) &\leq \log(|H|) + \sum_{t=1}^{T}\log(1-\epsilon F^t) \quad [\text{ take log on both sides }] \\
&\leq \log(|H|) + \sum_{t=1}^{T} -\epsilon F^t \quad [\ \log(1-x) \leq -x\ ] \\
\Rightarrow \sum_{t=1}^{T}\epsilon F^t &\leq \log(|H|) + m\log(\frac{1}{1-\epsilon}) \\
\epsilon E[M] &\leq \log(|H|) + m(\epsilon+\epsilon^2) \quad [\ \log(\frac{1}{1-x}) \leq x + x^2\ ] \\
E[M] &\leq \frac{\log(|H|)}{\epsilon} + m(1+\epsilon)
\end{aligned} \tag{3}$$

□

From the conclusion of Theorem 1 we can derive the following

$$\begin{aligned} E[M] &\leq m + m\epsilon + \frac{\log(|H|)}{\epsilon} \\ &\leq m + \epsilon T + \frac{\log(|H|)}{\epsilon} \\ &\leq m + 2\sqrt{T\log(|H|)} \quad [\text{ set } \epsilon = \sqrt{\frac{\log(|H|)}{T}}] \\ \Rightarrow \frac{E[M]}{T} &\leq \frac{m}{T} + 2\sqrt{\frac{\log(|H|)}{T}} \end{aligned} \tag{4}$$

In equation (4), $\frac{m}{T}$ is the average number of mistakes if you know the future (means knowing all samples in the beginning), $\frac{E[M]}{T}$ is the average number of mistakes if you don't know the future, and $\frac{E[M]}{T} - \frac{m}{T}$ is defined as the *regret* of the algorithm, which measures how "sorry" the learner is not to have followed the predictions of the best function $h \in H$. It is clear that

$$regret = \frac{E[M]}{T} - \frac{m}{T} \leq 2\sqrt{\frac{\log(|H|)}{T}} \to 0 \quad as \quad T \to \infty \tag{5}$$

It tells us that if using RMW and the sample sequence is large enough ($T \to \infty$), one can almost do as well as the best function in H. This can be extended to online decision making.

2 Online Decision Making using RMW

2.1 Definition and examples

There are N experts (or actions, decisions, etc.). On each day t, pick an expert i_t to use (like choosing a function $h \in H$). Then the adversary/environment returns a loss vector $l^t = (l_1^t, l_2^t, \cdots, l_N^t)$ (corresponding to the true labels in previous model). After T days, your performance is $\frac{1}{T}\sum_{t=1}^{T} l_{i_t}^t$, and the performance of the best expert is $\min_{j\in[N]} \frac{1}{T}\sum_{t=1}^{T} l_j^t$. Similar to Theorem 1, we have the following theorem.

Theorem 2. *If $l_i^t \in [0, \gamma]$, then $E[\frac{1}{T}\sum_{t=1}^{T} l_{i_t}^t] - \min_{j\in[N]} \frac{1}{T}\sum_{t=1}^{T} l_j^t \leq 2\gamma\sqrt{\frac{\log N}{T}}$*

It should be noted that if $l_i^t \notin \{0, 1\}$ in RMW, then the updating rule changes to $w_h^{t+1} \leftarrow (1 - \epsilon l_h^t) w_h^t$. Here are some examples of online decision making:

Example 1 (Page Replacement Policy)**.** *In a computer operating system that uses paging for virtual memory management, page replacement algorithms decide which memory pages to page out (swap out, write to disk) when a page of memory needs to be allocated.*

In this example, *experts* are different page replacement algorithms, like LRU (least recently used), FIFO (first in first out), and *loss* is cache miss or not.

Example 2 (Search Optimization)**.** *For a search engine, it aims to order different links such that the higher a link ranked on the search results page, the more times users will click it.*

In this example, *experts* are all orderings of search results, and *loss* is the depth of the link clicked.

Example 3 (Adaptive Routing)**.** *On the Internet, there are often some paths between the same start point and destination. A system needs to alter the path according to the conditions in order to achieve the best performance, such as lowest latency.*

In this example, *experts* are all possible paths from start to destination, and *loss* is the performance (latency) of the chosen path.

Example 4 (Robotics)**.** *A robot is at a certain grid of a maze, and there is gold at different grids of the maze. The robot aims to collect as much gold as possible within a period of time.*

In this example, *experts* are actions of the robot, $\{up, down, left, right\}$, and *loss* is if there is gold in the grid or not.

Example 5 (Clinical Trials)**.** *The doctor needs to test the effectiveness of some drugs.*

In this example, *experts* are different drugs, and *loss* is the result of taking a drug.

2.2 Bandit setting and EXP3 algorithm

In some situations, we cannot get the loss of all experts at time t. Is it possible for us to achieve zero regret? The answer is yes.

Bandit Setting: On each day t, we pick an expert i_t from N experts, but we can only get the loss of that expert $l_{i_t}^t$, instead of the loss of all experts. The process is illustrated in figure 1. By using RMW, the regret may not approach to 0 as $T \to \infty$. Intuitively this is because the overal best expert might make all its mistakes in the beginning. RMW will make its weight really small and from then on the algorithm will not pick it for a long time.

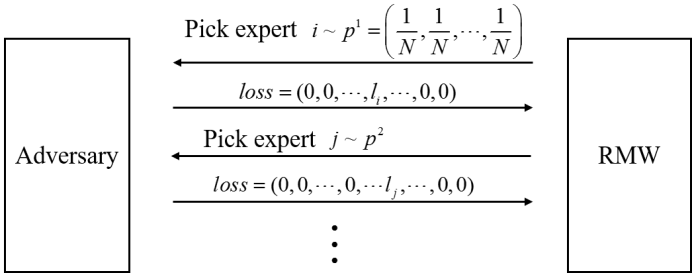

Figure 1: RMW in bandit setting

EXP3 Algorithm: The process of EXP3 algorithm is illustrated in figure 2, where the probability of picking each expert is $q^t = (1-\gamma)p^t + \gamma(\frac{1}{N}, \frac{1}{N}, \cdots, \frac{1}{N})$, and the loss returned to RMW is $\hat{l}_i^t = l_i^t / q_i^t$. With the EXP3 algorithm, it is guaranteed that $regret \to 0$ as $T \to \infty$. More specifically, we have the following theorem.

Theorem 3. *For any $T > 0$, assuming $l_i^t \in [0, 1]$, then $\exists \gamma$, $Regret(EXP3) \leq 2\sqrt{\frac{N \log N}{T}}$.*

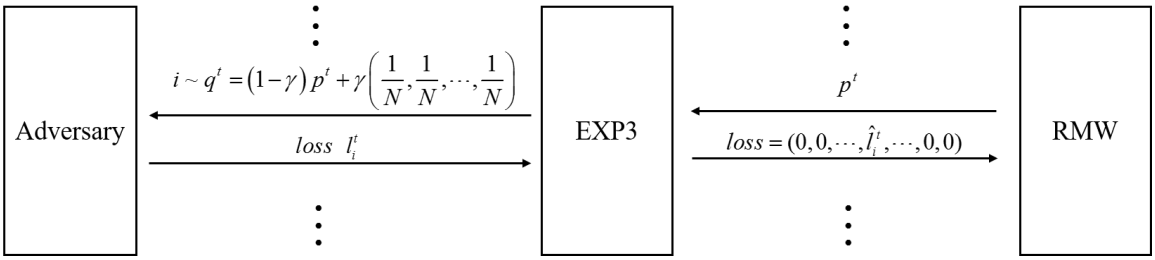

Figure 2: EXP3 algorithm