On Batch Learning in a Binary Weight Setting

Shao C. Fang and Santosh S. Venkatesh[1]
Department of Electrical Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA

Abstract — **We consider the problem of inferring a finite binary sequence $\boldsymbol{w}^* \in \{-1,1\}^n$ from a random sequence of half-space data $\{\mathbf{u}^{(t)} \in \{-1,1\}^n : \langle \boldsymbol{w}^*, \mathbf{u}^{(t)}\rangle \geq 0, t \geq 1\}$. In this context, we show that a previously proposed randomised on-line learning algorithm dubbed Directed Drift [1] has minimal space complexity but an expected mistake bound exponential in n. We show that batch incarnations of the algorithm allow of massive improvements in running time. In particular, using a batch of $\frac{1}{2}\pi n \log n$ examples at each update epoch reduces the expected mistake bound to $\mathcal{O}(n)$ in a single bit update mode, while using a batch of $\pi n \log n$ examples at each update epoch in a multiple bit update mode lead to convergence to $\boldsymbol{w}^*$ with a constant (independent of n) expected mistake bound.**

I. Introduction

Write $\mathbb{B} \triangleq \{-1,1\}^n$ for simplicity and let $\mathbb{B}^n \triangleq \{-1,1\}^n$ denote the vertices of the binary n-cube. Let $\boldsymbol{w}^* \in \mathbb{B}^n$ be some fixed (but unknown) vertex. Suppose we are provided with a random labelled sequence of positive examples $\{\mathbf{u}^{(t)}, t \geq 1\}$ of $\boldsymbol{w}^*$ obtained by independent sampling from the uniform distribution on the positive half-space of vertices

$$\mathbb{B}^n_+(\boldsymbol{w}^*) \triangleq \{\mathbf{u} \in \mathbb{B}^n : \langle \boldsymbol{w}, \mathbf{u}\rangle \geq 0\}.$$

Our goal is to infer the finite binary sequence $\boldsymbol{w}^*$ in an efficient (on-line) manner from the sample $\{\mathbf{u}^{(t)}\}$.

II. Directed Drift

Directed Drift[1] is a randomised, on-line learning algorithm with minimal space complexity.

Algorithm D (*Directed Drift*). Given a confidence parameter $\delta > 0$ and a sample of positive examples $\{\mathbf{u}^{(t)}, t \geq 1\}$ generated by independent sampling from the uniform distribution on $\mathbb{B}^n_+(\boldsymbol{w}^*)$, the algorithm generates a hypothesis $\boldsymbol{w}$ which, with confidence in excess of $1-\delta$, coincides with the concept $\boldsymbol{w}^*$.

D1. [Initialise.] Set epoch $t \leftarrow 1$, confidence counter $T \leftarrow 0$, and select an arbitrary initial hypothesis $\boldsymbol{w} \in \mathbb{B}^n$.

D2. [Is the hypothesis consistent on the example?] Set $Y \leftarrow \langle \boldsymbol{w}, \mathbf{u}^{(t)}\rangle$.

D3. [If it ain't broke, don't fix it.] If $Y \geq 0$, increment the confidence counter $T \leftarrow T+1$: if $T \geq \sqrt{\frac{\pi n}{2}} \log \delta^{-1}$, output the hypothesis $\boldsymbol{w}$ and terminate the algorithm; else go to step D5.

D4. [Update hypothesis.] Else (if $Y < 0$) set $T \leftarrow 0$, $J \leftarrow \{j : w_j \neq u_j^{(t)}\}$ and pick a random index j from the uniform distribution on J. Set $w_j \leftarrow -w_j$ and leave the other components of $\boldsymbol{w}$ unchanged.

D5. [Increment time and iterate.] Set $t \leftarrow t+1$ and go back to step D2.

By a consideration of the equilibrium probability distribution of the states of the finite Markov chain which represents the system we show that the algorithm has minimal space complexity $2n$ and exponential time complexity $\Omega\left(e^{0.139n}\right)$.[2]

Massive improvements in running time result if the algorithm is modified to run in batch mode. In a single bit update batch mode, Step D4 is replaced by

D4′ [Update hypothesis.] Else (if $Y < 0$) set $T \leftarrow 0$ and call an additional $m-1$ examples $\mathbf{u}^{(t+1)}, \ldots, \mathbf{u}^{(t+m-1)}$. Define the indicator functions

$$I_k^{(s)} = \begin{cases} 1 & \text{if } w_k \neq u_k^{(s)}, \\ 0 & \text{if } w_k = u_k^{(s)}, \end{cases}$$

and select the index j garnering the most votes: $j \leftarrow \arg\max_k \sum_{s=t}^{t+m-1} I_k^{(s)}$. Set $w_j \leftarrow -w_j$ and leave the other components of $\boldsymbol{w}$ unchanged. Set $t \leftarrow t+m-1$.

In a multiple bit update batch mode, Step D4 is replaced by

D4″ [Update hypothesis.] Else (if $Y < 0$) set $T \leftarrow 0$ and call an additional $m-1$ examples $\mathbf{u}^{(t+1)}, \ldots, \mathbf{u}^{(t+m-1)}$. Define the indicator functions

$$I_k^{(s)} = \begin{cases} 1 & \text{if } w_k \neq u_k^{(s)}, \\ 0 & \text{if } w_k = u_k^{(s)}. \end{cases}$$

Tally the votes $b_k = \sum_{s=t}^{t+m-1} I_k^{(s)}$ and order the indices such that $b_{j_1} \geq b_{j_2} \geq \cdots \geq b_{j_n}$. Set $w_j \leftarrow -w_j$ if $j \in \{j_1, \ldots, j_{\lfloor (1-Y)/2 \rfloor}\}$ and leave the other components of $\boldsymbol{w}$ unchanged. Set $t \leftarrow t+m-1$.

Relatively small batch sizes m are needed. We show that in a single bit update batch mode, a batch size of $m = \frac{1}{2}\pi n \log n$ reduces the time complexity of the algorithm to $\mathcal{O}(n)$ while in a multiple bit update batch mode, a batch size of $m = \pi n \log n$ reduces the time complexity of the algorithm to $\mathcal{O}(1)$, independent of n.

References

[1] S. S. Venkatesh, "Directed Drift: a new linear threshold algorithm for learning binary weights on-line," *J. Comp. Sys. Sciences*, vol. 46, pp. 198–217, 1993.

[1]This work was supported by the Air Force Office of Scientific Research under grants F49620-93-1-0120 and F49620–92–J–0344.

[2]We use the number of bits of buffer memory needed as a measure of space complexity and the expected mistake bound of the algorithm, i.e., the expected number of epochs when an example is misclassified by the current hypothesis, as a measure of the algorithm's time complexity.