CSE 512 - Machine Learning HW 3

Mihir Chakradeo 111462188

I. Boosting

ML-HW3

[I] Boosting

(1) Show That: $\epsilon_{Train} = \frac{1}{N}\sum_{j=1}^{N} \delta(H(x^j) \neq y^j) \leq \frac{1}{N}\sum_{j=1}^{N} \exp(-f(x^j)y^j)$

LHS = $\epsilon_{Train} = \frac{1}{N}\sum_{j=1}^{N} \delta(H(x^j) \neq y^j)$

It is given that, $\delta(H(x^j) \neq y^j) = \begin{cases} 1, & \text{if } H(x^j) \neq y^j \\ 0, & \text{otherwise} \end{cases}$

$\therefore \epsilon_{Train} = \frac{1}{N}\sum_{j=1}^{N} \begin{cases} 1, & \text{if } H(x^j) \neq y^j \\ 0 & \text{otherwise} \end{cases}$

We also know that $H(x^j) = \text{sign}\{f(x^j)\}$ & $y^j \in \{-1, 1\}$

- If we have true $y^j = -1$ & we predict $f(x^j) = +1$. then $y^j f(x^j) = -1$.
- when true $y^j = +1$ & $f(x^j) = -1$, $y^j f(x^j) = -1$.
- when true $y^j = -1$ & $f(x^j) = -1$, $y^j f(x^j) = 1$
- When $y^j = +1$ & $f(x^j) = +1$, $y^j f(x^j) = 1$.

We can see, that $y^j f(x^j) \geq 0$ means no error and $y^j f(x^j) < 0$ means there is some error.

$\therefore \epsilon_{Train} = \frac{1}{N}\sum_{j=1}^{N} \begin{cases} 1, & \text{if } y^j f(x^j) \leq 0 \\ 0, & \text{Otherwise.} \end{cases}$

Now, for some z, $\exp(-z) \geq 1$ if $z \leq 0$

~~and exp(-z) ≤~~

$\therefore \exp(-y^j f(x^j)) \geq 1$ when $y^j f(x^j) \leq 0$.

$\therefore \epsilon_{Train} \leq \frac{1}{N}\sum_{j=1}^{N} \exp(-y^j f(x^j))$

(2) Show that: $\frac{1}{N}\sum_{j=1}^{N} \exp(-f(x^j)y^j) = \prod_{t=1}^{T} Z_t$.

→ It is given that:

$$w_j^{(t+1)} = \frac{w_j^{(t)} \exp(-\alpha_t y^j h_t(x^j))}{Z_t}$$

where $Z_t = \sum_{j=1}^{N} w_j^{t} \exp(-\alpha_t y^j h_t(x^j))$

Let us consider 1 sample.

As y^j and $h_t(x^j) \in \{-1, 1\}$.

$$w_j^{(t+1)} = w_1^{(t)} \cdot \frac{\exp(-\alpha_1 y^j h_1(x^j))}{Z_1} \cdots \frac{\exp(-\alpha_T y^j h_T(x^j))}{Z_T}$$

But, we initialize $w_1^{(t)} = \frac{1}{N}$.

$$\therefore w_j^{(t+1)} = \frac{1}{N} \cdot \frac{\exp(-\alpha_1 \cdot y^j h_1(x^j))}{Z_1} \cdots \frac{\exp(-\alpha_T y^j h_T(x^j))}{Z_T}.$$

$$= \frac{1}{N} \frac{\exp(-y^j \sum_{t=1}^{T} \alpha_t h_t(x^j))}{\prod_{t=1}^{T} Z_t}$$

Given that,

$$f(x^j) = \sum_{t=1}^{T} \alpha_t h_t(x^j).$$

$$\therefore w_j^{(t+1)} = \frac{1}{N} \frac{\exp(-y^j f(x^j))}{\prod_{t=1}^{T} Z_t}.$$

Now, let us extend for N data points:

$$\sum_{j=1}^{N} w_j^{(t+1)} = \left[\frac{1}{N}\right]\left[\frac{\sum_{j=1}^{N} \exp(-y^j f(x^j))}{\prod_{t=1}^{T} Z_t}\right]$$

Now, $w_j^{(t+1)}$ is a normalizing factor. i.e $\sum_{j=1}^{N} w_j^{(t+1)} = 1$.

$$\therefore\ 1 = \frac{1}{N} \frac{\sum_{j=1}^{N} \exp(-y^j f(x^j))}{\prod_{j=1}^{T} Z_t}$$

$$\text{i.e. } \frac{1}{N} \sum_{j=1}^{N} \exp(-f(x^j) y^j) = \prod_{j=1}^{T} Z_t$$

---

(3)

(a) Find the value of α_t that minimizes Z_t, and then ST: $Z_t^{opt} = 2\sqrt{\epsilon_t(1-\epsilon_t)}$.

→ We know: $Z_t = (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$

(i) Finding optimal α_t.

$$\frac{\partial Z_t}{\partial \alpha} = 0 \Rightarrow 0 = -(1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t \exp(\alpha_t)$$

$$(1-\epsilon_t)\exp(-\alpha_t) = \epsilon_t \exp(\alpha_t)$$

$$\frac{(1-\epsilon_t)}{\epsilon_t} = (\exp(\alpha_t))^2$$

$$\therefore\ \alpha_t = \ln\sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$$

(ii) Subs. optimal α_t in Z_t to get Z_t^{opt}.

$$\therefore\ Z_t^{opt} = (1-\epsilon_t)\exp\left(-\ln\sqrt{\frac{1-\epsilon_t}{\epsilon_t}}\right) + \epsilon_t \exp\left(\ln\sqrt{\frac{1-\epsilon_t}{\epsilon_t}}\right)$$

$$= \frac{(1-\epsilon_t)\sqrt{\epsilon_t}}{\sqrt{1-\epsilon_t}} + \epsilon_t\sqrt{\frac{1-\epsilon_t}{\epsilon_t}}$$

$$= \frac{(1-\epsilon_t)\epsilon_t + \epsilon_t(1-\epsilon_t)}{(\sqrt{1-\epsilon_t})(\sqrt{\epsilon_t})}$$

$$Z_t^{opt} = \frac{2\epsilon_t(1-\epsilon_t)}{\sqrt{\epsilon_t(1-\epsilon_t)}} = 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

(b) From part (a)

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

But, $\epsilon_t = \frac{1}{2} - \gamma_t$.

$$\therefore Z_t = 2 \cancel{\sqrt{\gamma_t(1-\gamma_t)}} \sqrt{\left(\tfrac{1}{2}-\gamma_t\right)\left(1-\tfrac{1}{2}+\gamma_t\right)}$$

$$= 2\sqrt{\left(\tfrac{1}{2}-\gamma_t\right)\left(\tfrac{1}{2}+\gamma_t\right)}$$

Now $(a+b)(a-b) = a^2 - b^2$.

$$\therefore Z_t = 2\sqrt{\left(\tfrac{1}{2}\right)^2 - \gamma_t^2}$$

$$= 2\sqrt{(1/4) - (\gamma_t^2)}$$

$$= \sqrt{\tfrac{4}{4} - 4\gamma_t^2}$$

$$= \sqrt{1 - 4\gamma_t^2}.$$

But, $\log(1-x) \leq -x$. for $0 \leq x < 1$

$$\therefore 1-x \leq e^{-x}.$$

$$\therefore \sqrt{1-x} \leq \sqrt{e^{-x}}$$

$$\therefore \sqrt{1-x} \leq e^{-x/2}$$

$$\therefore \sqrt{1-4\gamma_t^2} \leq e^{-\frac{4\gamma_t^2}{2}}$$

$$\leq \exp(-2\gamma_t^2)$$

$$\therefore \underline{Z_t \leq \exp(-2\gamma_t^2)}.$$

(c) we have:

$$\epsilon_{train} \leq \exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right)$$

we want to show that if each classifier is better than random, then: $\epsilon_{train} \leq \exp(-2T\gamma^2)$ $\left\{ \begin{array}{l} \gamma_t \geq \gamma, \forall t \\ \gamma > 0 \end{array} \right.$

→ We know that $\gamma_t > 0$ implies better than random.

i.e $\gamma_t \geq \gamma$, where $\gamma > 0$. $\forall t$.

i.e $\gamma_t^2 \geq \gamma^2$.

Now, we want an upper bound on ϵ_{train}.

∴ we can replace γ_t^2 by γ^2.

i.e $\epsilon_{train} \leq \exp\left(-2\sum_{t=1}^{T}\gamma^2\right)$

$$\underline{\epsilon_{train} \leq \exp(-2T\gamma^2)}.$$

II. Action Recognition with CNN

Submitted ipython notebook in zip

1. **First TODO: define model** - in notebook

2. **Second TODO: define loss function and optimizer** - in notebook

3. **Third TODO: Train model and show accuracy-** in notebook

4. **Fourth TODO: Tell us what you did:**
   I tried the VGG-19 from ResNet paper: https://arxiv.org/pdf/1512.03385.pdf.The convolutional layers have a kernel_size of 3 throughout. After successive conv, activations, there are maxpool layers, with a kernel size of 2, which reduce the size of output by 2. After every Max pooling layer, the number of filters is doubled. At the end I have used three fully connected layers

5. **Fifth TODO: Submit score to Kaggle**
   Kaggle Score: 61.03975

6. **Sixth TODO: Flatten 3d -** In notebook

7. **Seventh TODO: Design a network using 3D convolution on videos for video classification -** In notebook

8. **Eighth TODO: Tell us what you did:**
   I have used a two layer network with a fixed filter size of 3. The number of filters for 1st conv layer is 8 with padding 2, and then it is doubled (16) for the next conv layer, again with padding 2. After each convolutional layer, I have used BatchNorm. After RELU activations, I have used Max Pooling with a kernel size of 2 and stride 2.

   Kaggle Score: 58.10397