Deep Learning: Homework 1

Instructed by *Yi Wu*

Due on Mar 10, 2021

Runlong Zhou YaoClass 82 2018011309

1.1

Answer:

- **Convolutional layer.**

$$\frac{\mathrm{d}(\mathrm{weight}(j,k)\star z(k))(h,w)}{\mathrm{d}z(a,b,c)}=\mathbb{I}\{k=a,\ h\le b<h+k_H,\ w\le c<w+k_W\}\mathrm{weight}(j,k,b-h,c-w),$$

hence,

$$\begin{aligned}\frac{\mathrm{d}y(j,h,w)}{\mathrm{d}z(a,b,c)}&=\sum_{k=0}^{C_{\mathrm{in}}-1}\mathbb{I}\{k=a,\ h\le b<h+k_H,\ w\le c<w+k_W\}\mathrm{weight}(j,k,b-h,c-w)\\&=\mathbb{I}\{h\le b<h+k_H,\ w\le c<w+k_W\}\mathrm{weight}(j,a,b-h,c-w).\end{aligned}$$

Further,

$$\begin{aligned}\frac{\mathrm{d}L}{\mathrm{d}z(a,b,c)}&=\sum_{j,h,w}\frac{\mathrm{d}L}{\mathrm{d}y(j,h,w)}\cdot\frac{\mathrm{d}y(j,h,w)}{\mathrm{d}z(a,b,c)}\\&=\sum_{j=0}^{C_{\mathrm{out}}-1}\sum_{h=b-k_H+1}^{b}\sum_{w=c-k_W+1}^{c}\mathrm{weight}(j,a,b-h,c-w)\frac{\mathrm{d}L}{\mathrm{d}y(j,h,w)}.\end{aligned}$$

$$\frac{\mathrm{d}y(j,h,w)}{\mathrm{d}\mathrm{weight}(j,a,b,c)}=z(a,h+b,w+c),$$

hence,

$$\frac{\mathrm{d}L}{\mathrm{d}\mathrm{weight}(j,a,b,c)}=\sum_{h,w}z(a,h+b,w+c)\frac{\mathrm{d}L}{\mathrm{d}y(j,h,w)}.$$

$$\frac{\mathrm{d}L}{\mathrm{d}\mathrm{bias}(j)}=\sum_{h,w}\frac{\mathrm{d}L}{\mathrm{d}y(j,h,w)}.$$

- **Max-pooling layer.**

$$\frac{\mathrm{d}y(j,h,w)}{\mathrm{d}z(a,b,c)}=\mathbb{I}\left\{j=a,\ (b,c)=\arg\max_{(m,n)\le(k_H-1,k_W-1)}z(j,\mathrm{stride}[0]\times h+m,\mathrm{stride}[1]\times w+n)\right\},$$

hence,

$$\begin{aligned}
&\frac{\mathrm{d}L}{\mathrm{d}z(a,b,c)} \\
&= \sum_{j,h,w} \frac{\mathrm{d}L}{\mathrm{d}y(j,h,w)} \cdot \frac{\mathrm{d}y(j,h,w)}{\mathrm{d}z(a,b,c)} \\
&= \sum_{h,w} \mathbb{I}\left\{(b,c) = \arg \max_{(m,n)\leq(k_H-1,k_W-1)} z(a, \mathrm{stride}[0]\times h+m, \mathrm{stride}[1]\times w+n)\right\} \frac{\mathrm{d}L}{\mathrm{d}y(a,h,w)}.
\end{aligned}$$

- **Tanh.**

$$\frac{\mathrm{d}y}{\mathrm{d}z} = \frac{(\mathrm{e}^z+\mathrm{e}^{-z})^2-(\mathrm{e}^z-\mathrm{e}^{-z})^2}{(\mathrm{e}^z+\mathrm{e}^{-z})^2} = 1-y^2,$$

hence,

$$\frac{\mathrm{d}L}{\mathrm{d}z} = \frac{\mathrm{d}L}{\mathrm{d}y}\cdot\frac{\mathrm{d}y}{\mathrm{d}z} = (1-y^2)\frac{\mathrm{d}L}{\mathrm{d}y}.$$

2

Answer:

1. Three settings with smallest test losses are listed as follows:

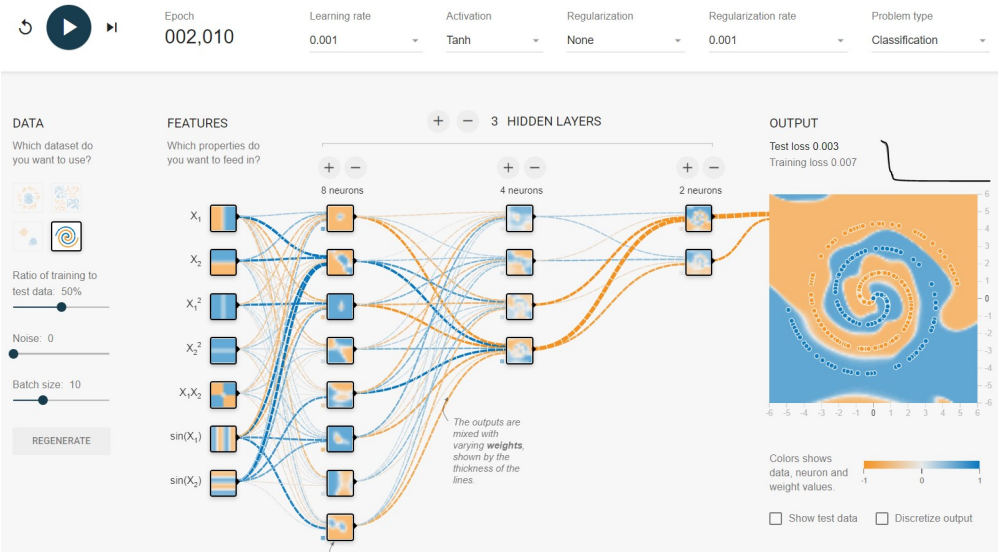

Figure 1: Test loss = 0.003

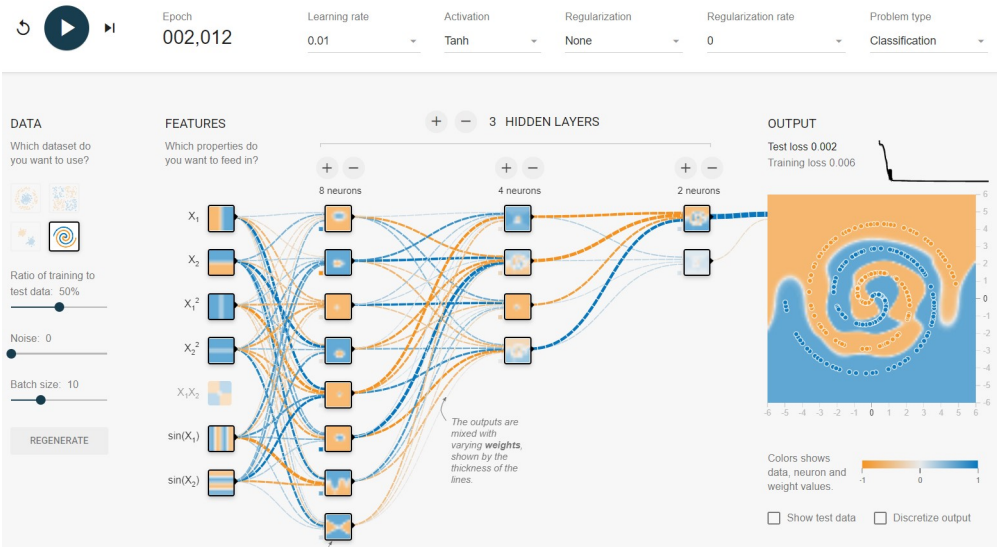

Figure 2: Test loss = 0.002

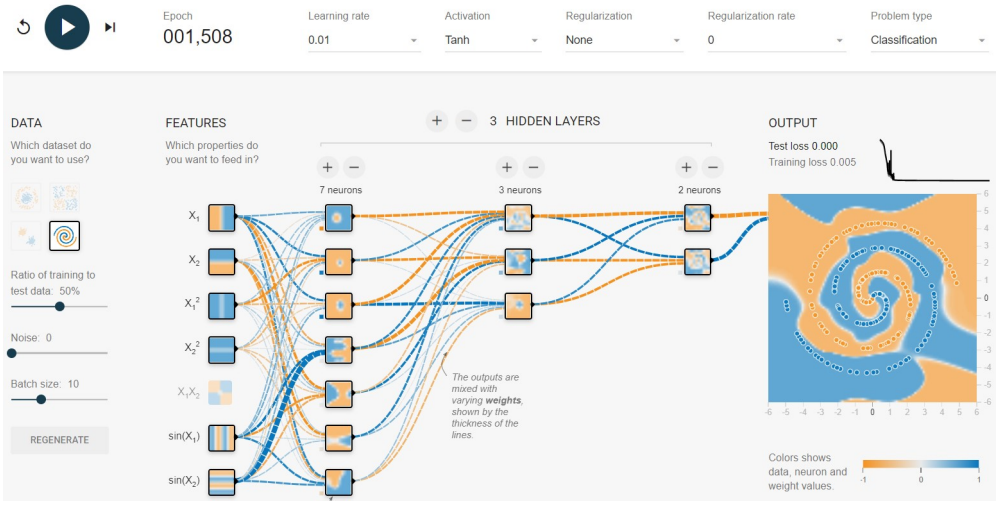

Figure 3: Test loss = 0.000

The first setting includes all the kernels so it works fairly well. By observing that x_1x_2 had diminishing weights during training, it was removed in the second setting. By cutting more neurons out, a lighter network (the last setting) is attained.

2. **Learning rate:** Higher learning rates make losses decrease faster in the beginning, but the losses are more likely to jitter afterwards. Lower learning rates give more smooth optimizing process, but the convergence rate is slower.

   Hidden nodes: More hidden nodes make convergence slower and do not necessarily mean lower test loss. Extremely small number of nodes cannot learn well either.

   Regularization: Higher regularization factor makes convergence faster, but may converge to a worse model since the some weights are suppressed to lower values, nullifying some nodes.