
Notes

Release alpha

Sep 22, 2020

Contents

1	c++ notes	1
1.1	1. Why should use member initialization list?	1
2	floating point number	3
3	ieee 规定的 16, 32, 64, 128 比特的浮点数格式列表	5
4	cuda related	7
4.1	how to install nvidia driver in linux?	7
4.2	download nvidia driver	8
5	Backpropagation 的推导	9
5.1	约定	9
5.2	公式推导	9
5.3	BP 算法总结	10
6	convolution arithmetic	11
6.1	1. convolution	11
6.2	2. pooling	11
6.3	3. tansposed convolution	12
6.4	4. dilated convolution	12
7	tensorrt 学习笔记	13
8	cmake notes	15
9	git tricks	17
9.1	How to delete all commit history in github?	17
10	linux notes	19

10.1	linux installer shell maker	19
10.2	How to disable gui when ubuntu desktop version booting	19
10.3	How to disable guest session	20
10.4	ubuntu 18.04 disable xorg using nvidia card!	20
10.5	fix time difference in Ubuntu & Windows Dual Boot	21
11	reStructuredText notes	23
11.1	headings	23
12	Indices and tables	25

1.1 1. Why should use member initialization list?

For POD class members, it makes no difference.

For class members which are classes, then it avoids an unnecessary call to a default constructor.

Furthermore, if a class doesn't have a default constructor, or you have a const member variable, you must use an initializer list:

CHAPTER 2

floating point number

浮点数最新标准为 IEEE 754-2019

浮点数格式如下：

S(sign)	E (biased exponent)	T (trailing significand field)
1 bit	w bits	t bits, $t = p - 1$

具有如下关系：

$$\begin{aligned}e &= E - bias \\e_{max} &= bias = 2^{w-1} - 1 \\e_{min} &= 1 - e_{max}\end{aligned}$$

关于 biased E 的说明：

1. normal number: $[1, 2^w - 2]$, 值为 $(-1)^s \times 2^{E-bias} \times (1 + 2^{1-p} \times T)$
2. 0, 当 $T=0$ 表示 ± 0 ; 当 $T \neq 0$ 表示 subnormal number, 值为 $(-1)^s \times 2^{e_{min}} \times (0 + 2^{1-p} \times T)$
3. $2^w - 1$ (二进制全部为 1), 当 $T=0$, 表示 $\pm\infty$; 当 $T \neq 0$, 表示 NaN.

ieee 规定的 16, 32, 64, 128 比特的浮点数格式列表

参数	binary16	binary32	binary64	binary128
指数位数	5	8	11	15
emax/bias	15	127	1023	16383
小数位数	10	23	52	112

4.1 how to install nvidia driver in linux?

Problem: NVIDIA kernel module 'nvidia-drm' already be loaded in kernel

Solution (in root):

1. goto true tty3, pressing **Ctrl+Alt+F2**
2. disable the graphical target:

```
systemctl isolate multi-user.target
```

3. unload the Nvidia drivers:

```
modprobe -r nvidia-drm
```

4. install cuda driver
5. start the GUI again:

```
systemctl start graphical.target
```

6. confirm your driver version:

```
nvidia-smi
```

4.2 download nvidia driver

website: <https://www.nvidia.com/Download/Find.aspx>

Backpropagation 的推导

5.1 约定

$$z_j^{l+1} = \sum_k w_{jk}^l a_k^l + b_j^l, \quad a_j^l = \sigma(z_j^l) \quad (5.1)$$

其中, z_j^l 表示未激活前第 l 层、第 j 个神经元的值, w_{jk}^l 为连接第 l 层第 j 个神经元和第 $l+1$ 层第 k 个神经元的权重, a_k^l 表示激活后的第 l 层第 k 个神经元的值, b_j^l 为偏移量 bias, σ 为激活函数。

注意和书籍 <http://neuralnetworksanddeeplearning.com/chap2.html> 中公式 (23) 的约定有区别, 我们把 weight 和 bias 和神经元的值放到同一层中。

公式 (5.1) 写成矩阵形式为:

$$z^{l+1} = w^l a^l + b^l, \quad a^l = \sigma(z^l)$$

5.2 公式推导

我们约定 C 为损失函数 (loss function), 并记:

$$\delta^l = \frac{\partial C}{\partial z^l}$$

约定 **Hadamard product** 或者 elementwise 相乘为 (重复指标不求和):

$$u \odot v = u_i * v_i$$

根据公式 (5.1) 可以直接得出对偏移量 b 的偏导数 (梯度):

$$\frac{\partial C}{\partial b_j^l} = \sum_i \frac{\partial C}{\partial z_i^{l+1}} \frac{\partial z_i^{l+1}}{\partial b_j^l} = \frac{\partial C}{z_j^{l+1}} = \delta_j^{l+1}$$

上式写成矩阵形式为:

$$\frac{\partial C}{\partial b^l} = \delta^{l+1}$$

对权重 w 的求导为:

$$\frac{\partial C}{\partial w_{jk}^l} = \sum_i \frac{\partial C}{\partial z_i^{l+1}} \frac{\partial z_i^{l+1}}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^{l+1}} a_k^l = \delta_j^{l+1} a_k^l$$

上式写成矩阵形式为:

$$\frac{\partial C}{\partial w^l} = \delta^{l+1} (a^l)^T$$

l 层 δ^l 和 $l+1$ 层的 δ^{l+1} 的关系为:

$$\frac{\partial C}{\partial z_j^l} = \sum_{i,k} \frac{\partial C}{\partial z_i^{l+1}} \frac{\partial z_i^{l+1}}{\partial a_k^l} \frac{\partial a_k^l}{\partial z_j^l} = \sum_i \delta_i^{l+1} w_{ij}^l \sigma'(z_j^l)$$

上式写成矩阵形式为:

$$\delta^l = (w^l)^T \delta^{l+1} \odot \sigma'(z^l)$$

可以看出:

$$\nabla_a C = (w^l)^T \delta^{l+1}$$

5.3 BP 算法总结

BP 算法可以概括为以下四个关系式:

$$\begin{aligned} \delta^l &= \frac{\partial C}{\partial z^l} = \nabla_z C \\ \frac{\partial C}{\partial w^l} &= \delta^{l+1} (a^l)^T \\ \frac{\partial C}{\partial b^l} &= \delta^{l+1} \\ \delta^l &= (w^l)^T \delta^{l+1} \odot \sigma'(z^l) \end{aligned}$$

可以看出, 可以从 δ^{l+1} 的推导出对第 l 层的权重和偏移量的偏导, 以及第 l 层的未激活前的神经元的偏导。

reference

- https://github.com/vdumoulin/conv_arithmetic
- <https://arxiv.org/abs/1603.07285>

6.1 1. convolution

Set input data size i , convolution kernel size k , stride size s , and zero padding size p . Then the output size o is:

$$o = \left\lfloor \frac{i + 2p - k}{s} \right\rfloor + 1. \quad (6.1)$$

The floor function $\lfloor \cdot \rfloor$ can found at https://en.wikipedia.org/wiki/Floor_and_ceiling_functions.

6.2 2. pooling

According to (6.1), pooling output size is:

$$o = \left\lfloor \frac{i - k}{s} \right\rfloor + 1. \quad (6.2)$$

6.3 3. transposed convolution

explanation The convolution operation can be rewritten to matrix multiplication.

6.4 4. dilated convolution

The dilation “rate” is controlled by an additional hyperparameter d . A kernel of size k dilated by a factor d has an effective size:

$$\hat{k} = k + (k - 1)(d - 1).$$

Combined with (6.1) the output size is:

$$o = \left\lfloor \frac{i + 2p - k - (k - 1)(d - 1)}{s} \right\rfloor + 1. \quad (6.3)$$

1. `nvinfer1::Dims` 表示的是 CHW 的各个纬度，而不是 NCHW!
2. `ModelImporter` 类实现了 `nvonnxparser::IParser`
3. onnx parser 中增加对插件的支持，需要修改 `builtin_op_importers.cpp`
4. 增加插件后，需要在文件 `InferPlugin.cpp` 注册插件

CHAPTER 8

cmake notes

1. cmake command line option for x64 architecture:

```
cmake -A x64 ..
```


9.1 How to delete all commit history in github?

1. Checkout

```
git checkout --orphan latest_branch
```

2. Add all files

```
git add -A
```

3. Commit the changes

```
git commit -am "commit message"
```

4. Delete the branch

```
git branch -D master
```

5. Rename the current branch to master

```
git branch -m master
```

6. Finally, force update your repository

```
git push -f origin master
```

10.1 linux installer shell maker

<https://github.com/megastep/makeself>

10.2 How to disable gui when ubuntu desktop version booting

1. modify grub:

```
sudo vim /etc/default/grub
```

change

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
```

to

```
GRUB_CMDLINE_LINUX_DEFAULT="text"
```

update grub

```
sudo update-grub
```

2. disable lightdm service

```
sudo systemctl disable lightdm.service
```

3. If you want to start desktop

```
sudo service lightdm start
```

4. To enable lightdm service, systemd has bug! ref: <https://bugs.launchpad.net/ubuntu/+source/systemd/+bug/1595454> solution(root user or use sudo):

```
systemctl enable lightdm  
ln -s /lib/systemd/system/lightdm.service /etc/systemd/system/display-manager.service
```

10.3 How to disable guest session

```
sudo vim /usr/share/lightdm/lightdm.conf.d/50-ubuntu.conf
```

add:

```
allow-guest=false
```

10.4 ubuntu 18.04 disable xorg using nvidia card!

ref: <https://askubuntu.com/questions/1061551/how-to-configure-igpu-for-xserver-and-nvidia-gpu-for-cuda-work>

1. create file /etc/X11/xorg.conf with the following content:

```
Section "Device"  
    Identifier      "intel"  
    Driver          "intel"  
    BusId           "PCI:0:2:0"  
EndSection  
  
Section "Screen"  
    Identifier      "intel"  
    Device          "intel"  
EndSection
```


10.5 fix time difference in Ubuntu & Windows Dual Boot

Ubuntu use the hardware clock (RTC, real time clock) in universal time (UTC) by default while Windows use the clock in local time.

easy solution in ubuntu

```
$ sudo timedatectl set-local-rtc 1
```


11.1 headings

- # H1, with overline, for parts
- * H2, with overline, for chapters
- = H3, for sections
- - H4, for subsections
- ^ H5, for subsubsections
- ” H6, for paragraphs

CHAPTER 12

Indices and tables

- `genindex`
- `modindex`