Data format

{id: 11, category: 100, imgs:[ {picture: b’xxxx’}, {picture: b’xxxx’},{picture: b’xxxxx’} ] }

Similarity:

1. All have huge amount of images. ImageNet has over 1,500 million images. The ILSVRC competition hold every year has 1.2 million images, and about 5000 labels. Cdiscount has around 15 million, much less than ImageNet, but more than the ILSVRC competition. The labels for Cdiscount is around 5000.
2. Resolution. The average image resolution on ImageNet is bigger , but it is common to crop the original image to 256\*256 for better speed. Cdiscount images are 180\*180. So it is comparable in resolution.

So from the comparison, the Cdiscount image classification is very similar to ILSVRC competition, but bigger in image size.

A somewhat different aspect is that Cdiscount has multiple image for one product, but as I checked, it is not frequent. Average pic/product is 1.3. So it might not have a major impact in how we deal with the challenge.

Given the high similarity between Cdiscount and ILSVRC competition, it’s naturally to look into the history of ILSVRC for ideas.

History of ILSVRC:

|  |  |  |  |
| --- | --- | --- | --- |
| Year |  | Top-5 error rate(%) | depth |
| 2012 | AlexNet | 16.3 | 8 |
| 2014 | VGGNet |  | 19 |
| 2014 | Inception Net v1 | 6.67 | 22 |
| 2015 | ResNet | 3.57 | 152 |
| 2016 | CUImage(ensemble) |  |  |

Time estimate:

GTX 1080

Cloud GPU computing resource selection:

Amazon: pricy and slow for on demand GPU instance(0.90/hour for Nvidia K80). 0.27 for Spot Instance.

Paperspace: 0.6 for Nvidia P5000

Floyd: pricy but the interface is really simple, only 2 hour of free GPU time.

Google: $0.74/hour for Nvidia K80. $300 of credit for use.

Use your own machine: a desktop with GTX1080 costs $1000+.

Conclusion: use Google.

Code:

Steps for deep learning

1, use bson.decode\_file\_iter to get a generator for all train/test files.

2, use itertools.islice() to generate batches.

3, output batches of (image,category) data.

Note batch is fixed for sample id, but sample have various number to pictures. So the number of (image,category) data is different.

K80 totalMemory: 11.17GiB freeMemory: 11.09GiB

Cdc-vgg kernel

# without 4th conv

Batch size=16 epoch time =94 hour total batch 695885

Batch size=64 epoch time=69 hour total batch 173971

Batch size=128 OOM error

OOM when allocating tensor with shape[61952,4096]

Number of parameters: 299 767 254

Flatten shape `(?, 61952)`

Output shape `(?, 5270)

#with 4th conv

Batch size 16 epoch time=81 h

Batch size 64 epoch time = 69 h

Batch size 128 epoch time= 73 h

Number of parameters: 105 520 086

Flatten shape `(?, 12800)`

#adding a maxpool [2,2] before the net.

Number of parameters 61479894

Flatten shape (?, 2048)

Batch size 128 epoch time= 19h

Batch size 300 epoch time =18h shows Allocator (GPU\_0\_bfc) ran out of memory

First try.

#adding a maxpool [2,2]. So I am using an image size 90\*90

//resizing to [3,3] is not helping running time.

Learning rate 0.01

Epoch 1

Batch size 128.

Epoch time=18 h.

Prediction speed.

About 800/min.

So about 50000/h

Takes about 2 hours. The count goes up to 9XXXX for batch size of 32.

Exception ignored in: <bound method BaseSession.\_\_del\_\_ of <tensorflow.python.client.session.Session object at 0x7fa8c4c44b70>>hon/client/session.py", line 696, in \_\_del\_\_Traceback (most recent call last):TypeError: 'NoneType' object is not callable