Difficulty of Diagnosis

ConvNets helping to determine the
difficulty of diagnosis by image

VGG16 outputs the “prevalence”

How much does the CPU feel that *that* is the actual class

```
print(labels[str(i.item())])
#d = dir(i)
#for e in d: print(e)
```

```
tensor([-3.0172e+00, -1.1986e+00, -1.8998e+00, -1.1971e+00, -1.5325e+00,
        -1.5370e-02,  6.0900e-01, -1.6167e+00,  1.5567e+00, -2.5063e+00,
        -1.5624e+00, -2.3448e+00,  5.7230e-01, -1.7869e+00, -2.6640e+00,
         8.9858e-01, -2.4701e+00, -2.7066e+00, -3.2224e+00, -3.2978e+00,
        -3.8337e+00, -7.4573e-01, -1.0688e+00, -1.4380e+00,  7.7847e-01,
        -4.3500e+00, -4.5720e+00, -3.7380e+00, -4.3666e+00, -1.1037e+00,
         3.6357e-01, -5.5198e-01, -2.2667e-01, -9.6151e-02, -1.6193e+00,
        -1.7525e+00,  1.7974e+00, -1.5633e+00,  6.4028e-01,  1.2000e+00,
        -2.8107e+00, -1.8544e-01, -2.6185e+00,  3.9450e-01, -3.0935e-01,
        -1.8663e+00, -3.6287e-01, -8.3725e-01, -1.1526e+00, -2.2598e+00,
        -1.1135e+00, -1.3149e+00, -3.5583e+00, -2.6807e+00,  4.1116e-01,
        -1.3745e+00, -2.0054e+00, -9.5899e-01, -2.0634e+00, -9.5245e-01,
         6.5549e-02, -1.5371e+00, -2.0845e+00, -2.3176e+00, -1.7499e+00,
        -4.2196e+00,  3.5117e-01,  9.4370e-01, -2.5235e+00,  1.0221e+00,
        -4.1560e+00, -1.1523e+00, -2.4089e+00, -1.4440e+00, -1.5444e+00,
        -2.7348e+00,  1.9103e+00, -9.3136e-01, -4.0622e-01, -1.5044e-01,
        -3.4186e+00, -4.5549e+00,  8.9357e-01, -9.1193e-01, -5.2819e-01,
        -4.5147e-01,  5.4927e-02,  8.5835e-02,  1.3158e+00, -1.1495e+00,
         6.9001e-02, -5.5846e+00, -4.9061e+00, -9.8066e-01, -8.8657e-01,
```

Sorting by the class ID

- Just printing each class and its prevalence

```
#--list by class:
for i in range(N):
    print(i, end = "\t")
    print(labels[str(i)], end = "\t")
    print(cp[str(i)])

##--from now on, we only sort data and see which classes are 'winners':
```

```
0       ['n01440764', 'tench']  -1.0186396837234497
1       ['n01443537', 'goldfish']       2.167151689529419
2       ['n01484850', 'great_white_shark']      -3.485780954360962
3       ['n01491361', 'tiger_shark']    0.33351564407348633
4       ['n01494475', 'hammerhead']     -2.7718982696533203
5       ['n01496331', 'electric_ray']   0.3085658848285675
6       ['n01498041', 'stingray']       -3.0836760997772217
7       ['n01514668', 'cock']   0.7915195226669312
8       ['n01514859', 'hen']    2.153531551361084
9       ['n01518878', 'ostrich']        -2.5958735942840576
10      ['n01530575', 'brambling']      1.983551025390625
11      ['n01531178', 'goldfinch']      -0.343048095703125
12      ['n01532829', 'house_finch']    2.110304355621338
13      ['n01534433', 'junco']  -2.1741549968719482
14      ['n01537544', 'indigo_bunting'] -2.2756004333496094
15      ['n01558993', 'robin']  1.7396655082702637
16      ['n01560419', 'bulbul'] 0.4599645435810089
17      ['n01580077', 'jay']    -1.105634093284607
18      ['n01582220', 'magpie'] -2.240165948867798
```

Sorting by Prevalence

- Now we see the winner class and other classes

```
for i in range(N):
    print(sorted_cp[i], end = "\t")
    print(labels[sorted_cp[i]], end = "\t")
    print(cp[sorted_cp[i]])
```

```
281     ['n02123045', 'tabby']  15.74615478515625
285     ['n02124075', 'Egyptian_cat']   15.612914085388184
282     ['n02123159', 'tiger_cat']      15.478860855102539
287     ['n02127052', 'lynx']   11.005281448364258
283     ['n02123394', 'Persian_cat']    10.740737915039062
457     ['n02883205', 'bow_tie']        9.233245849609375
292     ['n02129604', 'tiger']  9.04011058807373
722     ['n03942813', 'ping-pong_ball'] 8.49628734588623
622     ['n03657121', 'lens_cap']       8.137709617614746
284     ['n02123597', 'Siamese_cat']    8.125133514404297
288     ['n02128385', 'leopard']        7.560521125793457
761     ['n04074963', 'remote_control'] 7.471639156341553
673     ['n03793489', 'mouse']  7.291515827178955
478     ['n02971356', 'carton'] 7.221921443939209
286     ['n02125311', 'cougar'] 6.845936298370361
289     ['n02128757', 'snow_leopard']   6.821112632751465
291     ['n02129165', 'lion']   6.516599655151367
539     ['n03223299', 'doormat']        6.441550254821777
720     ['n03937543', 'pill_bottle']    6.406008243560791
```

Sorting by Prevalence

- Now we see the winner class and other classes
- Checking only winner does not seem to be right

```
for i in range(N):
    print(sorted_cp[i], end = "\t")
    print(labels[sorted_cp[i]], end = "\t")
    print(cp[sorted_cp[i]])
```

```
281     ['n02123045', 'tabby']  15.74615478515625
285     ['n02124075', 'Egyptian_cat']   15.612914085388184
282     ['n02123159', 'tiger_cat']      15.478860855102539
287     ['n02127052', 'lynx']   11.005281448364258
283     ['n02123394', 'Persian_cat']    10.740737915039062
457     ['n02883205', 'bow_tie']        9.233245849609375
292     ['n02129604', 'tiger']  9.04011058807373
722     ['n03942813', 'ping-pong_ball'] 8.49628734588623
622     ['n03657121', 'lens_cap']       8.137709617614746
284     ['n02123597', 'Siamese_cat']    8.125133514404297
288     ['n02128385', 'leopard']        7.560521125793457
761     ['n04074963', 'remote_control'] 7.471639156341553
673     ['n03793489', 'mouse']  7.291515827178955
478     ['n02971356', 'carton'] 7.221921443939209
286     ['n02125311', 'cougar'] 6.845936298370361
289     ['n02128757', 'snow_leopard']   6.821112632751465
291     ['n02129165', 'lion']   6.516599655151367
539     ['n03223299', 'doormat']        6.441550254821777
720     ['n03937543', 'pill_bottle']    6.406008243560791
```

Sorting by Prevalence

- Now we see the winner class and other classes
- Checking only winner does not seem to be right
- Since the top three classes are quite close, we may want to think about all three

```
for i in range(N):
    print(sorted_cp[i], end = "\t")
    print(labels[sorted_cp[i]], end = "\t")
    print(cp[sorted_cp[i]])
```

```
281     ['n02123045', 'tabby']  15.74615478515625
285     ['n02124075', 'Egyptian_cat']   15.612914085388184
282     ['n02123159', 'tiger_cat']      15.478860855102539
287     ['n02127052', 'lynx']   11.005281448364258
283     ['n02123394', 'Persian_cat']    10.740737915039062
457     ['n02883205', 'bow_tie']        9.233245849609375
292     ['n02129604', 'tiger']  9.04011058807373
722     ['n03942813', 'ping-pong_ball'] 8.49628734588623
622     ['n03657121', 'lens_cap']       8.137709617614746
284     ['n02123597', 'Siamese_cat']    8.125133514404297
288     ['n02128385', 'leopard']        7.560521125793457
761     ['n04074963', 'remote_control'] 7.471639156341553
673     ['n03793489', 'mouse']  7.291515827178955
478     ['n02971356', 'carton'] 7.221921443939209
286     ['n02125311', 'cougar'] 6.845936298370361
289     ['n02128757', 'snow_leopard']   6.821112632751465
291     ['n02129165', 'lion']   6.516599655151367
539     ['n03223299', 'doormat']        6.441550254821777
720     ['n03937543', 'pill_bottle']    6.406008243560791
```

Sorting by Prevalence

- The more close values we have, the more difficult should be visual exam, too.
- Difficulty of diagnosis

```
for i in range(N):
    print(sorted_cp[i], end = "\t")
    print(labels[sorted_cp[i]], end = "\t")
    print(cp[sorted_cp[i]])
```

```
281     ['n02123045', 'tabby']  15.74615478515625
285     ['n02124075', 'Egyptian_cat']   15.612914085388184
282     ['n02123159', 'tiger_cat']      15.478860855102539
287     ['n02127052', 'lynx']   11.005281448364258
283     ['n02123394', 'Persian_cat']    10.740737915039062
457     ['n02883205', 'bow_tie']        9.233245849609375
292     ['n02129604', 'tiger']  9.04011058807373
722     ['n03942813', 'ping-pong_ball'] 8.49628734588623
622     ['n03657121', 'lens_cap']       8.137709617614746
284     ['n02123597', 'Siamese_cat']    8.125133514404297
288     ['n02128385', 'leopard']        7.560521125793457
761     ['n04074963', 'remote_control'] 7.471639156341553
673     ['n03793489', 'mouse']  7.291515827178955
478     ['n02971356', 'carton'] 7.221921443939209
286     ['n02125311', 'cougar'] 6.845936298370361
289     ['n02128757', 'snow_leopard']   6.821112632751465
291     ['n02129165', 'lion']   6.516599655151367
539     ['n03223299', 'doormat']        6.441550254821777
720     ['n03937543', 'pill_bottle']    6.406008243560791
```

Further study

- More relevant data
- Other models that VGG16
  - Other champions: VGG19, AlexNet, ResNet, LeNet, uNet, etc etc
  - Homemade architecture
- Definition of “difficult diagnosis”