0️⃣ Import Libraries & Defining Constants

Mount GoogleDrive

[ ] ↳ *1 Zelle ausgeblendet*

Import the usual suspects ... 🤡🤦🥷

[ ] ↳ *1 Zelle ausgeblendet*

And some new friends ... 🧞🧚🦄

```
import os
from fastai.metrics import error_rate, accuracy
import random
import shutil
import pathlib
import numpy as np
```

Defining Constants 🗿🪨

```
# Defining Constants
BASE_FOLDER = "/content/gdrive/My Drive/fastai/unpackai/flowers"
PATH_TO_TRAIN = BASE_FOLDER+"/training/"
PATH_TO_TEST = BASE_FOLDER+"/testing/"
PATH_TO_MODELS = BASE_FOLDER+"/models/"
CATEGORIES = ["daisy","dandelion","rose","sunflower","tulip"]
```

1️⃣ Some Prework

Building a Test Dataset

In `Flower Categorization.ipynb` I missed a build to test dateset. I will do this now by manually creating two folders `training` and `testing` within my `flowers` folder on my *Google Drive*, I moved all the folders containing the flower images into the training folder, built the same flower categories as folders within the `testing` folder. I then moved a sample of all the images from my training folder to the testing folder with the following code:

```
def create_testset(src_folder ,dest_folder,test=0.2):
  files = os.listdir(src_folder)
  no_of_files = int(len(files) * test)

  #Move files
  for file_name in random.sample(files, no_of_files):
    shutil.move(os.path.join(src_folder, file_name), dest_folder)

for flower in CATEGORIES:
  src = PATH_TO_TRAIN+flower
  dst = PATH_TO_TEST+flower
  create_testset(src, dst,test=0.2)
```

```
print("TRAINING\n=======")
```

```
print("TRAINING\n=======")
for flower in CATEGORIES:
  src = PATH_TO_TRAIN + flower
  print(flower,": ",len(Path(src).ls()))

print("\nTESTING\n=======")
for flower in CATEGORIES:
  src = PATH_TO_TEST + flower
  print(flower,": ",len(Path(src).ls()))
```

```
TRAINING
=======
daisy :  616
dandelion :  787
rose :  628
sunflower :  586
tulip :  780

TESTING
=======
daisy :  154
dandelion :  196
rose :  156
sunflower :  146
tulip :  195
```

Load the **old** Training Results

Zum Bearbeiten doppelklicken (oder Eingabe)

```
# I saved the training results but forgot to copy it to my GDrive 😉.
#Anyhow, the best Accuracy I got was around 85% - This is the benchmark I need to beat 💪

#learn4.load('flower-stage-4')
```

2️⃣ Getting the Data

```
#PATH_TO_TRAIN
path = Path(PATH_TO_TRAIN)
path.ls()
```

```
(#6) [Path('/content/gdrive/My Drive/fastai/unpackai/flowers/training/rose'),Path('/content/gdrive/My Drive/fa
```

```
for flower in CATEGORIES:
  x = os.listdir(PATH_TO_TRAIN+flower)
  print(flower,": ",len(x))
```

```
daisy :  616
dandelion :  787
rose :  628
sunflower :  586
tulip :  780
```

3️⃣ Prepare for Training

```
# CREATE LIST OF PATH TO TRAINING IMAGES
fns = get_image_files(PATH_TO_TRAIN)
fns
```

```
(#3393) [Path('/content/gdrive/My Drive/fastai/unpackai/flowers/training/rose/16051111039_0f0626a241_n.jpg'),P
```

```
%%time
# CHECK IMAGES FOR BROKEN FILES
# ⏳ Takes longer to run
#failed = verify_images(fns)
#failed
```

```
CPU times: user 3 µs, sys: 0 ns, total: 3 µs
Wall time: 5.48 µs
```

DataBlock

```
flowers = DataBlock(
    blocks=(ImageBlock, CategoryBlock),
    get_items=get_image_files,
    splitter=RandomSplitter(valid_pct=0.25, seed=42),
    get_y=parent_label,
    item_tfms=Resize(128))
```

DataLoader

```
dls = flowers.dataloaders(PATH_TO_TRAIN)
dls_test = flowers.dataloaders(PATH_TO_TEST)
```

Investigate Dataset

```
#Show labels
dls.vocab
```

```
['daisy', 'dandelion', 'rose', 'sunflower', 'tulip']
```

```
# Size of Datasets
print("TRAINING TEST SIZES:")
print("Total: ", dls.n)
print("Train: ", dls.n-dls.valid.n)
print("Validation: ", dls.valid.n)
```

```
TRAINING TEST SIZES:
Total:  2545
Train:  1697
Validation:  848
```

```
dls.valid.show_batch(max_n=8, nrows=1)
```

4️⃣ Train

Metrics = Accuracy, because all groups ca same size

🔍 Saved Models Overview

No	NAME	MODEL PARAMETER	DATABLOCK	BEST EPOCH	ACCURACY

No	NAME	MODEL PARAMETER	DATABLOCK	BEST EPOCH	ACCURACY
0	flower_1a	`arch=resnet34`		2	0.84
1	flower_1b	`arch=resnet50`		19	0.89
2		`arch=resnet50` `base_lr=lr_min`			0.86
3	flower_1c	`arch=resnet50`	item_tfms=Resize(256)	18	0.93
4	flower_1d	`arch=resnet50`	`item_tfms=(RandomResizedCrop(256, min_scale=0.35)),` `batch_tfms=aug_transforms(mult=2)`	19	0.95
5	flower_1e	`arch=resnet50`	Normalization	19	0.95

0️⃣ Old Training Cycle: resnet34 👍

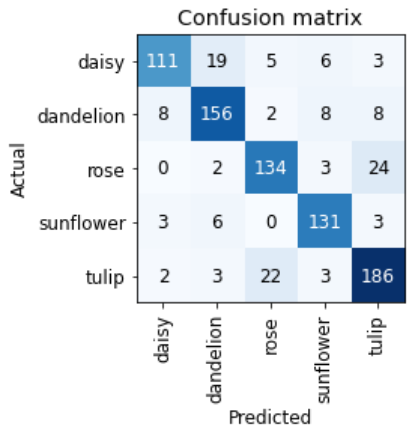
```
%%time
learn = cnn_learner(dls=dls, arch=resnet34, metrics=accuracy)
learn.fit_one_cycle(4)
```

epoch	train_loss	valid_loss	accuracy	time
0	1.540981	0.610043	0.773585	00:16
1	0.925402	0.512410	0.827830	00:16
2	0.654061	0.457857	0.843160	00:16
3	0.493504	0.456547	0.846698	00:16

```
CPU times: user 15.6 s, sys: 1.56 s, total: 17.1 s
Wall time: 1min 7s
```

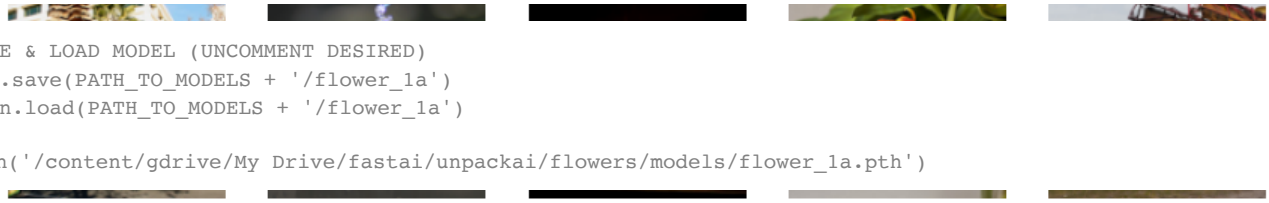
> 👆 This is the benchmark I have to beat 🎯🥊

```
interp = ClassificationInterpretation.from_learner(learn)
interp.plot_confusion_matrix()
```




```
# SHOW TOP 5 MISCLASSIFIED IMAGES
interp.plot_top_losses(5, nrows=1)
```

Zum Bearbeiten doppelklicken (oder Eingabe)



```
# SAVE & LOAD MODEL (UNCOMMENT DESIRED)
learn.save(PATH_TO_MODELS + '/flower_1a')
#learn.load(PATH_TO_MODELS + '/flower_1a')
```

```
Path('/content/gdrive/My Drive/fastai/unpackai/flowers/models/flower_1a.pth')
```

1️⃣ New Training Cycle: `resnet50` and longer training

```
# load resnet34 model
learn.load(PATH_TO_MODELS + '/flower_1a')
```

```
<fastai.learner.Learner at 0x7fbeaa232b50>
```

```
learn = cnn_learner(dls, resnet50, metrics=accuracy)
learn.fit_one_cycle(20) #I checked eopoch=20 but 19 was best
```

epoch	train_loss	valid_loss	accuracy	time
0	1.641093	0.688474	0.755896	00:18
1	1.015273	0.540087	0.813679	00:18
2	0.694151	0.524544	0.837264	00:18
3	0.510648	0.532615	0.838443	00:18
4	0.383808	0.556044	0.831368	00:18
5	0.317171	0.553483	0.844340	00:18
6	0.229155	0.480155	0.859670	00:18
7	0.175599	0.489543	0.864387	00:17
8	0.136901	0.539412	0.865566	00:18
9	0.107968	0.523926	0.860849	00:18
10	0.098762	0.515231	0.872642	00:18
11	0.080630	0.512159	0.869104	00:18
12	0.071790	0.525558	0.859670	00:18
13	0.060415	0.523310	0.871462	00:18
14	0.047943	0.510463	0.876179	00:18
15	0.040322	0.524173	0.864387	00:18
16	0.033364	0.512278	0.869104	00:18
17	0.028743	0.521003	0.869104	00:18
18	0.027532	0.507932	0.864387	00:18
19	0.030434	0.515206	0.864387	00:18

```
# SAVE MODEL (UNCOMMENT DESIRED)
learn.save(PATH_TO_MODELS + '/flower_1b')
```

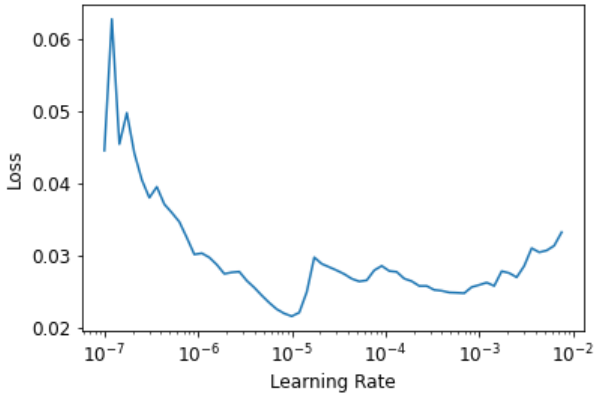
```
Path('/content/gdrive/My Drive/fastai/unpackai/flowers/models/flower_1b.pth')
```

2️⃣ New Training Cycle: Find optimal hyperparameter 👎

```
# LOAD PREVIOUS MODEL (UNCOMMENT DESIRED)
learn.load(PATH_TO_MODELS + '/flower_1b')
```

```
<fastai.learner.Learner at 0x7fbea2548d10>
```

```
lr_min, lr_steep = learn.lr_find()
```




```
print(f"Minim/10: {lr_min:.2e}, steepest point: {lr_steep:.2e}")
```

```
Minim/10: 1.00e-06, steepest point: 7.59e-07
```

```
# Finetune with minimum learning rate
learn.fine_tune(epochs=2, base_lr=lr_min)
```

epoch	train_loss	valid_loss	accuracy	time
0	0.029291	0.514919	0.866745	00:17

epoch	train_loss	valid_loss	accuracy	time
0	0.028583	0.528566	0.867925	00:18
1	0.027363	0.517406	0.859670	00:18

> This did not bring any improvment: let's keep the default learning rate!

3️⃣ New Training Cycle: change image size 👍

```
# LOAD PREVIOUS MODEL (UNCOMMENT DESIRED)
#learn.load(PATH_TO_MODELS + '/flower_1b')

# DataBlock
flowers = DataBlock(
    blocks=(ImageBlock, CategoryBlock),
    get_items=get_image_files,
    splitter=RandomSplitter(valid_pct=0.25, seed=42),
    get_y=parent_label,
    item_tfms=Resize(256)) # 👈 changed!
# DataLoader
dls = flowers.dataloaders(PATH_TO_TRAIN)
```

```
%%time
learn2 = cnn_learner(dls=dls, arch=resnet50, metrics=accuracy)
learn2.fit_one_cycle(20)
```

epoch	train_loss	valid_loss	accuracy	time
0	1.368746	0.425127	0.850236	00:35
1	0.735890	0.335120	0.882075	00:35
2	0.477259	0.331059	0.896226	00:35
3	0.332751	0.325810	0.903302	00:35
4	0.254248	0.360545	0.904481	00:35
5	0.190776	0.348506	0.899764	00:35
6	0.140587	0.367975	0.919811	00:35
7	0.097439	0.371187	0.904481	00:35
8	0.077251	0.354793	0.919811	00:35
9	0.063513	0.361446	0.915094	00:35
10	0.054430	0.336531	0.920991	00:35
11	0.044925	0.343764	0.918632	00:35
12	0.038827	0.334810	0.924528	00:35
13	0.032361	0.366777	0.915094	00:35
14	0.026138	0.346699	0.928066	00:35
15	0.022116	0.335553	0.922170	00:35
16	0.017763	0.343633	0.923349	00:35
17	0.013796	0.345700	0.924528	00:35
18	0.015290	0.341121	0.926887	00:35
19	0.013832	0.337118	0.922170	00:35

```
# SAVE & LOAD MODEL (UNCOMMENT DESIRED)
learn2.save(PATH_TO_MODELS + '/flower_1c')
```

```
Path('/content/gdrive/My Drive/fastai/unpackai/flowers/models/flower_1c.pth')
```

4️⃣ New Training Cycle: Data Augmentation 👍

```
# LOAD THE BEST MODEL
learn2.load(PATH_TO_MODELS + '/flower_1c')
```

```
<fastai.learner.Learner at 0x7fbea26f9490>
```

> Make sure that `learn2` is loaded
>
> - otherwise: go back to Chapter *Change Image Size* and reload `learn2`

```
# CHECK IF DATABLOCK AND DATALOADER ARE LOADED
if learn2: print("learn2 is loaded - you're ready to go ✅")
else: print("you need to load learn2 first 🚫")
```

```
learn2 is loaded - you're ready to go ✅
```

```
# add random sized cropped images

# Change DataBlock
flowers2=flowers.new(item_tfms=(RandomResizedCrop(256, min_scale=0.35)),
            batch_tfms=aug_transforms(mult=2))
# DataLoader
dls2 = flowers2.dataloaders(PATH_TO_TRAIN)
```

```
%%time
learn2 = cnn_learner(dls=dls2, arch=resnet50, metrics=accuracy)
learn2.fine_tune(20)
```

epoch	train_loss	valid_loss	accuracy	time
0	0.977287	0.523025	0.884434	00:37

epoch	train_loss	valid_loss	accuracy	time
0	0.443550	0.367064	0.908019	00:46
1	0.349229	0.355337	0.905660	00:46
2	0.283153	0.291493	0.916274	00:46
3	0.252086	0.349431	0.910377	00:46
4	0.239316	0.284095	0.929245	00:46
5	0.203785	0.337358	0.911557	00:46
6	0.170385	0.363783	0.906840	00:46
7	0.148083	0.300947	0.922170	00:46
8	0.149892	0.260880	0.936321	00:46
9	0.129948	0.358625	0.918632	00:46
10	0.123374	0.268963	0.932783	00:46
11	0.094072	0.257456	0.929245	00:46
12	0.075111	0.245667	0.936321	00:46
13	0.068691	0.239510	0.939858	00:46
14	0.054446	0.245523	0.937500	00:46
15	0.047843	0.227741	0.946934	00:46
16	0.038271	0.222068	0.944575	00:46
17	0.031211	0.221396	0.950472	00:46
18	0.030001	0.221863	0.945755	00:46
19	0.027054	0.215625	0.949292	00:46

```
CPU times: user 7min 26s, sys: 6min 41s, total: 14min 7s
Wall time: 16min 10s
```

```
# SAVE MODEL
learn2.save(PATH_TO_MODELS + '/flower_1d')
```

```
Path('/content/gdrive/My Drive/fastai/unpackai/flowers/models/flower_1d.pth')
```

5️⃣ New Training Cycle: Normalization Input Data 👎🏻

> "*... normalization becomes especially important when using pretrained models.*" (Jeremy Howard (2019): *Deep Learning for Coders with fastai and PyTorch*)

So I was wondering if **normalization** might increase the accuracy. I gave it a try:

```
def get_dls(bs, size):
    dblock = DataBlock(blocks=(ImageBlock, CategoryBlock),
                       get_items=get_image_files,
                   get_y=parent_label,
                   item_tfms=Resize(460),
                   batch_tfms=[*aug_transforms(size=size, min_scale=0.75),
```

```
                                Normalize.from_stats(*imagenet_stats)])
    return dblock.dataloaders(path, bs=bs)
dls3 = get_dls(64, 224)
```

```
%%time
learn3 = cnn_learner(dls=dls3, arch=resnet50, metrics=accuracy)
learn3.fine_tune(20)
```

epoch	train_loss	valid_loss	accuracy	time
0	0.895280	0.448588	0.892330	00:35

epoch	train_loss	valid_loss	accuracy	time
0	0.375619	0.286797	0.920354	00:42
1	0.268500	0.236564	0.924779	00:42
2	0.197797	0.240905	0.921829	00:42
3	0.162209	0.327786	0.912979	00:43
4	0.153562	0.276167	0.933628	00:42
5	0.156589	0.294993	0.932153	00:42
6	0.121960	0.242182	0.936578	00:42
7	0.089825	0.256572	0.929204	00:42
8	0.073802	0.219652	0.946903	00:42
9	0.066795	0.232735	0.936578	00:42
10	0.056784	0.226415	0.939528	00:42
11	0.047415	0.253357	0.938053	00:42
12	0.035931	0.201353	0.943953	00:42
13	0.027500	0.204848	0.945428	00:42
14	0.018792	0.179265	0.949853	00:42
15	0.015271	0.187761	0.945428	00:42
16	0.014127	0.175276	0.949853	00:42
17	0.011033	0.172132	0.952802	00:42
18	0.011604	0.168345	0.952802	00:42
19	0.008204	0.175305	0.949853	00:42

```
CPU times: user 6min 46s, sys: 5min 22s, total: 12min 8s
Wall time: 14min 53s
```

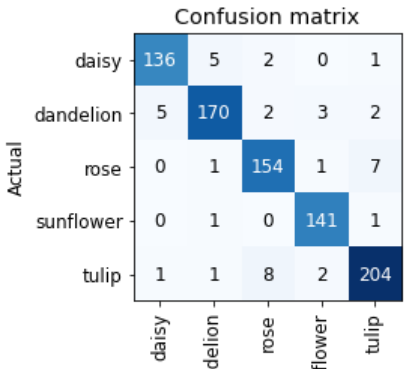
> Contrary to the Literature, normalization did not improve the outcome.

6️⃣ Fifth Training Cycle: Delete bad Images manually

```
# LOAD LAST SUCCESSFUL MODEL
learn2.load(PATH_TO_MODELS + '/flower_1d')
```

```
<fastai.learner.Learner at 0x7fbea2296710>
```

```
interp = ClassificationInterpretation.from_learner(learn2)
interp.plot_confusion_matrix()
```



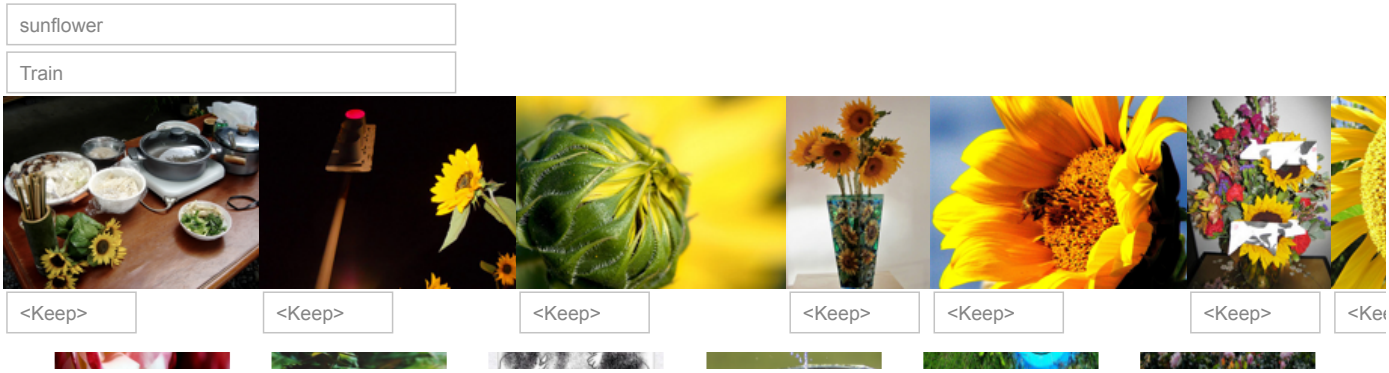
43 Images are wrongly classified! 👆

```
1 interp.plot_top_losses(43, nrows=8)
```

Prediction/Actual/Loss/Probability

It looks like most wrong classified images contain more than one flower, further more a few images does not contain any flowers at all.

```
1 cleaner = ImageClassifierCleaner(learn2)
2 cleaner
```




```
1 #this deletes (unlinks) the chosen images from the DataBlock
2 #rund this code for every flower category!!!
3 n=0
4 for idx in cleaner.delete():
5   n+=1
6   cleaner.fns[idx].unlink()
```

```
1 print("Deleted daisy: ",n)
```

```
Deleted daisy:  8
```

```
1 print("Deleted dandelion: ",n)
```

```
Deleted dandelion:  7
```


```
1 print("Deleted roses: ",n)
```

```
Deleted roses:  7
```

```
1 print("Deleted sunflowers: ",n)
```

```
Deleted sunflowers:  7
```

```
1 print("Deleted tulip: ",n)
```

```
Deleted tulip:  10
```

After deleting these images (actually: marking them - so the model will not use them any more) I received this error, indicating that

```
learn5 = cnn_learner(dls, resnet50, metrics=accuracy)
learn5.fit_one_cycle(20)
```

0.00% [0/20 00:00<00:00]

epoch	train_loss	valid_loss	accuracy	time

2.56% [1/39 00:01<00:57]

```
---------------------------------------------------------------------------
UnidentifiedImageError                    Traceback (most recent call last)
<ipython-input-78-33175da7cb15> in <module>()
      1 learn5 = cnn_learner(dls, resnet50, metrics=accuracy)
----> 2 learn5.fit_one_cycle(20)

13 frames
/usr/local/lib/python3.7/dist-packages/torch/_utils.py in reraise(self)
    427             # have message field
    428             raise self.exc_type(message=msg)
--> 429         raise self.exc_type(msg)
    430
    431

UnidentifiedImageError: Caught UnidentifiedImageError in DataLoader worker process 1.
Original Traceback (most recent call last):
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/_utils/worker.py", line 202, in _worker_loop
    data = fetcher.fetch(index)
  File "/usr/local/lib/python3.7/dist-packages/torch/utils/data/_utils/fetch.py", line 34, in fetch
    data = next(self.dataset_iter)
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/load.py", line 118, in create_batches
    yield from map(self.do_batch, self.chunkify(res))
  File "/usr/local/lib/python3.7/dist-packages/fastcore/basics.py", line 216, in chunked
    res = list(itertools.islice(it, chunk_sz))
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/load.py", line 133, in do_item
    try: return self.after_item(self.create_item(s))
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/load.py", line 140, in create_item
    if self.indexed: return self.dataset[s or 0]
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/core.py", line 333, in __getitem__
    res = tuple([tl[it] for tl in self.tls])
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/core.py", line 333, in <listcomp>
    res = tuple([tl[it] for tl in self.tls])
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/core.py", line 299, in __getitem__
    return self._after_item(res) if is_indexer(idx) else res.map(self._after_item)
  File "/usr/local/lib/python3.7/dist-packages/fastai/data/core.py", line 261, in _after_item
    def _after_item(self, o): return self.tfms(o)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/transform.py", line 200, in __call__
    def __call__(self, o): return compose_tfms(o, tfms=self.fs, split_idx=self.split_idx)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/transform.py", line 150, in compose_tfms
    x = f(x, **kwargs)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/transform.py", line 73, in __call__
    def __call__(self, x, **kwargs): return self._call('encodes', x, **kwargs)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/transform.py", line 83, in _call
    return self._do_call(getattr(self, fn), x, **kwargs)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/transform.py", line 89, in _do_call
    return retain_type(f(x, **kwargs), x, ret)
  File "/usr/local/lib/python3.7/dist-packages/fastcore/dispatch.py", line 118, in __call__
    return f(*args, **kwargs)
  File "/usr/local/lib/python3.7/dist-packages/fastai/vision/core.py", line 110, in create
    return cls(load_image(fn, **merge(cls._open_args, kwargs)))
  File "/usr/local/lib/python3.7/dist-packages/fastai/vision/core.py", line 85, in load_image
    im = Image.open(fn)
  File "/usr/local/lib/python3.7/dist-packages/PIL/Image.py", line 2896, in open
    "cannot identify image file %r" % (filename if filename else fp)
PIL.UnidentifiedImageError: cannot identify image file '/content/gdrive/My
Drive/fastai/unpackai/flowers/training/sunflower/20183028616_beb937e75c_m.jpg'
```

SEARCH STACK OVERFLOW

I tried to delete these images from `cleaner.fns`, but without final success and without knowing what caused this error:

```
# Delete files from cleaner which do not exist in folder
import os.path
print("Length before: "+len(cleaner.fns))
```

```
print("Length before: ",len(cleaner.fns))
for img in cleaner.fns:
  if not os.path.exists(img):
    cleaner.fns.remove(img)
print("Length after: ",len(cleaner.fns))
```

```
Length before:  30
Length after:  25
```

```
FileNotFoundError: Caught FileNotFoundError in DataLoader worker process 0.
Original Traceback (most recent call last):
...
[Errno 2] No such file or directory: '/content/gdrive/My Drive/fastai/unpackai/flowers/training/tulip/5717949231_23e2d4c297_n
```

```
! ls "/content/gdrive/My Drive/fastai/unpackai/flowers/training/tulip/5717949231_23e2d4c297_n.jpg"
#20183028616_beb937e75c_m.jpg
```

```
  File "<ipython-input-77-20ad59a03ed8>", line 1
    ````
    ^
SyntaxError: invalid syntax
```

SEARCH STACK OVERFLOW

## 7️⃣ New Training Cycle: change image size again 🤷‍♂️

Running this learning epochs - for whatever reason - gave me the following strange message:

```
%%time

# LOAD PREVIOUS MODEL (UNCOMMENT DESIRED)
learn2.load(PATH_TO_MODELS + '/flower_1b')

# DataBlock
flowers = DataBlock(
    blocks=(ImageBlock, CategoryBlock),
    get_items=get_image_files,
    splitter=RandomSplitter(valid_pct=0.25, seed=42),
    get_y=parent_label,
    item_tfms=Resize(400)) # 👈 changed!
# DataLoader
dls = flowers.dataloaders(PATH_TO_TRAIN)

learn2 = cnn_learner(dls=dls, arch=resnet50, metrics=accuracy)
learn2.fit_one_cycle(20)
```
```

0.00% [0/20 00:00<00:00]

epoch	train_loss	valid_loss	accuracy	time

0.00% [0/39 00:00<00:00]

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
<ipython-input-80-0d55d434ec1c> in <module>()
----> 1 get_ipython().run_cell_magic('time', '', "\n# LOAD PREVIOUS MODEL (UNCOMMENT
DESIRED)\nlearn2.load(PATH_TO_MODELS + '/flower_1b')\n\n# DataBlock\nflowers = DataBlock(\n    blocks=
(ImageBlock, CategoryBlock), \n    get_items=get_image_files, \n    splitter=RandomSplitter(valid_pct=0.25,
seed=42),\n    get_y=parent_label,\n    item_tfms=Resize(350)) # 👈 changed!\n# DataLoader\ndls =
flowers.dataloaders(PATH_TO_TRAIN)\n\n\nlearn2 = cnn_learner(dls=dls, arch=resnet50,
metrics=accuracy)\nlearn2.fit_one_cycle(20)")
```

29 frames

```
<decorator-gen-53> in time(self, line, cell, local_ns)
```

5️⃣ Final Test with Test Data

```
weight, bias, training, momentum, eps)
```

🌵🐫🐫...

```
torch.backends.cudnn.enabled
```

Finally, I was running out of GPU resources. And as I don't want to upgrade to `Colab Pro` at the moment, I leave for now ...