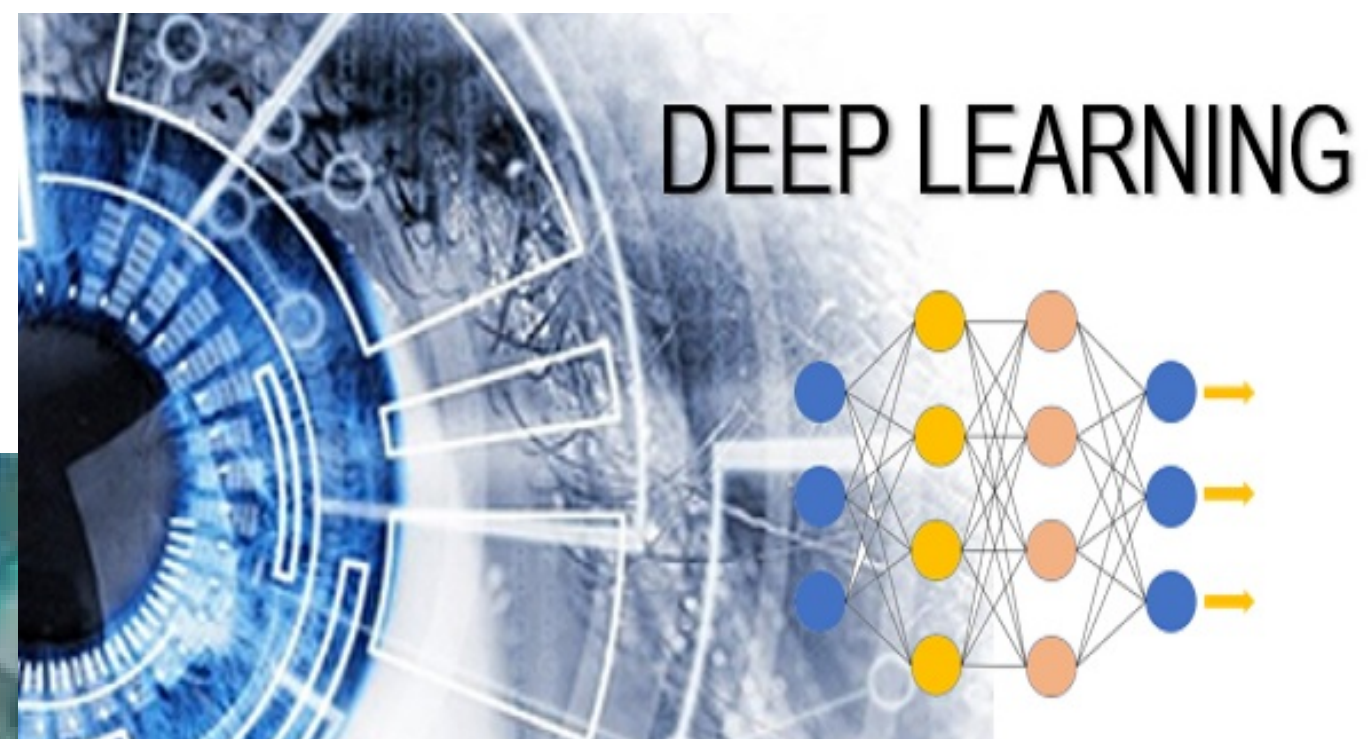




รายวิชา Individual Study 2110291

โมเดล Computer Vision แยกภาพ Xray วัณโรคปอด

PRESENTED BY PISIT WONGSRIPISANT
6331332021



ความเป็นมา / วัตถุประสงค์

ขั้นตอนการทำงาน / สิ่งที่ทำ

1. เชื่อมต่อ Google Colab กับ Google drive ของเรา

```
[ ] from google.colab import drive  
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

2. Install Kaggle library จากนั้น Download dataset tuberculosis-chest-xrays-shenzhen จาก Kaggle ลงใน Google Drive แล้วทำการ unzip ไฟล์ออกมา

```
[ ] from google.colab import files
files.upload() #this will prompt you to update the json

!pip install -q kaggle
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!ls ~/.kaggle
!chmod 600 /root/.kaggle/kaggle.json # set permission
```

No file chosen

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving kaggle.json to kaggle.json
kaggle.json

```
[ ] !kaggle datasets download -d raddar/tuberculosis-chest-xrays-shenzhen -p /content/gdrive/MyDrive/Colab Notebooks
```

```
[ ] import os
os.chdir('/content/gdrive/MyDrive/Colab Notebooks') #change dir
!unzip -q tuberculosis-chest-xrays-shenzhen.zip
```

3. ใช้ Library pandas ในการเปิดไฟล์ CSV ที่ได้จาก Dataset เพื่อข้อมูลรูปภาพ

```
import pandas as pd
import os
os.chdir('/content/gdrive/MyDrive/Colab Notebooks') #change dir
df=pd.read_csv('./shenzhen_metadata.csv')
df
```

| | study_id | sex | age | findings |
|-----|-------------------|--------|-----|----------------------------------------------------------------------------------------------------------------|
| 0 | CHNCXR_0001_0.png | Male | 45 | normal |
| 1 | CHNCXR_0002_0.png | Male | 63 | normal |
| 2 | CHNCXR_0003_0.png | Female | 48 | normal |
| 3 | CHNCXR_0004_0.png | Male | 58 | normal |
| 4 | CHNCXR_0005_0.png | Male | 28 | normal |
| ... | ... | ... | ... | ... |
| 657 | CHNCXR_0658_1.png | Male | 41 | bilateral secondary PTB with right pneumothorax |
| 658 | CHNCXR_0659_1.png | Male | 33 | secondary PTB in the left upper field |
| 659 | CHNCXR_0660_1.png | Male | 50 | 1.bilateral secondary PTB with right upper atelectasis;2.right pleural adhesions;3.left compensatory emphysema |
| 660 | CHNCXR_0661_1.png | Male | 26 | bilateral secondary PTB with right pleural thickening |
| 661 | CHNCXR_0662_1.png | Male | 32 | secondary PTB in the right upper field |

662 rows × 4 columns

4. Download Library fastai , สร้าง path ไปยังโฟลเดอร์ที่เก็บรูปภาพ X-ray

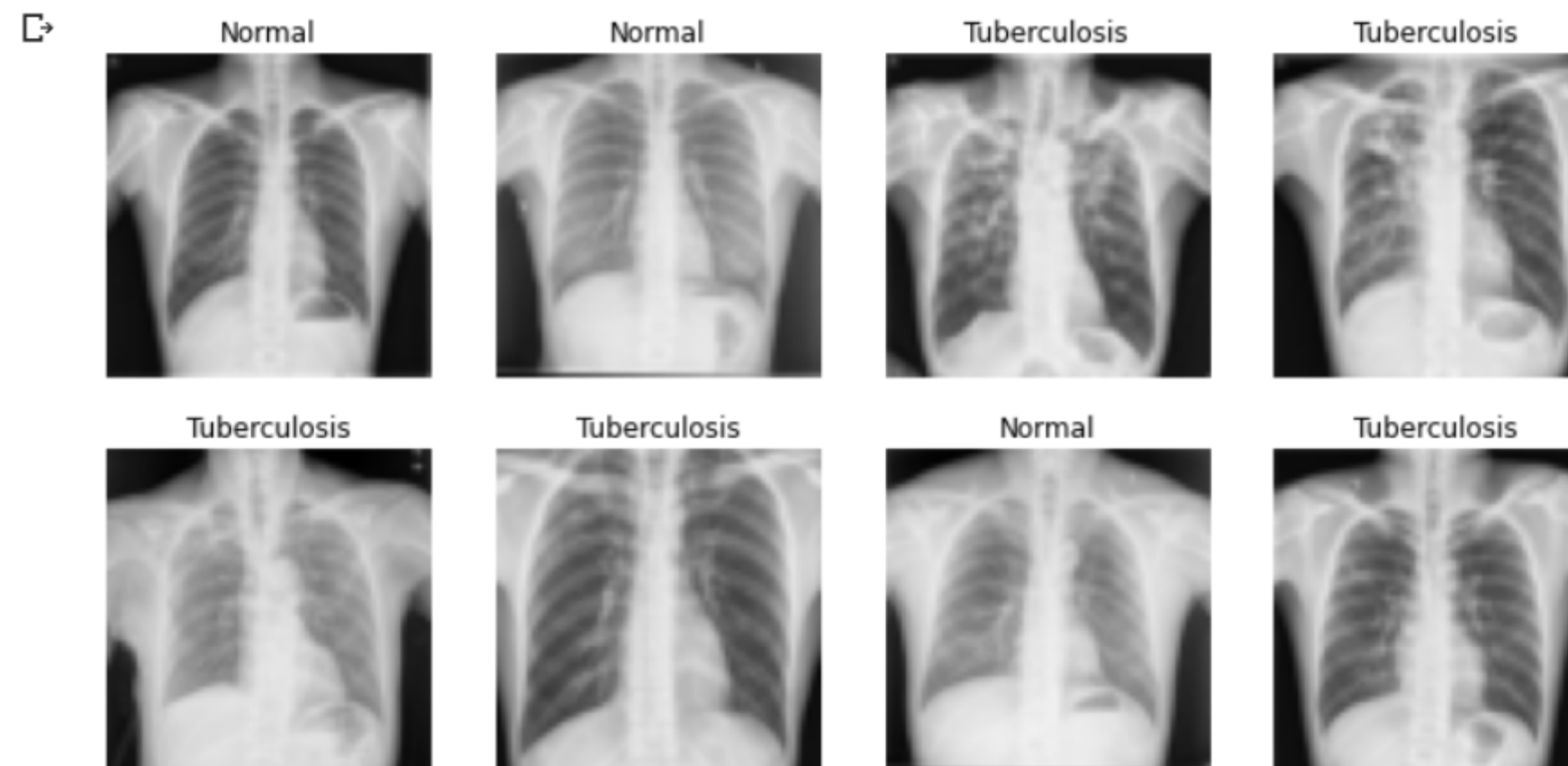
```
[ ] !pip install fastai --upgrade
    from fastai.vision.all import *
    path = Path('/content/gdrive/MyDrive/Colab Notebooks/images/images')
```

5. สร้าง Data Loader ซึ่งรูปภาพทั้งหมด 662 รูป แบ่งเป็น ปอดปกติ 326 รูป และ วัณโรคปอด 336 รูป และใช้วิธีแบ่ง data โดยแบ่ง test set ออกมาก่อน 10 % จากนั้น รูปที่เหลือจะแบ่งเป็น training set 80 % และ validation set 20 %

```
xray = DataBlock(  
    blocks=(ImageBlock, CategoryBlock),  
    get_items=get_image_files,  
    splitter=RandomSplitter(valid_pct=0.2, seed=42),  
    get_y=parent_label,  
    item_tfms=Resize(64))
```

```
dls = xray.dataloaders(path)
```

▶ `dls.train.show_batch(max_n=8, nrows=2)`



6. Model ที่ 1 ลอง Train model กันที โดยใช้ resnet50 architecture และใช้มาตรวัดเป็น Accuracy จะได้ค่า Accuracy จะอยู่ราว ๆ 78% – 80%

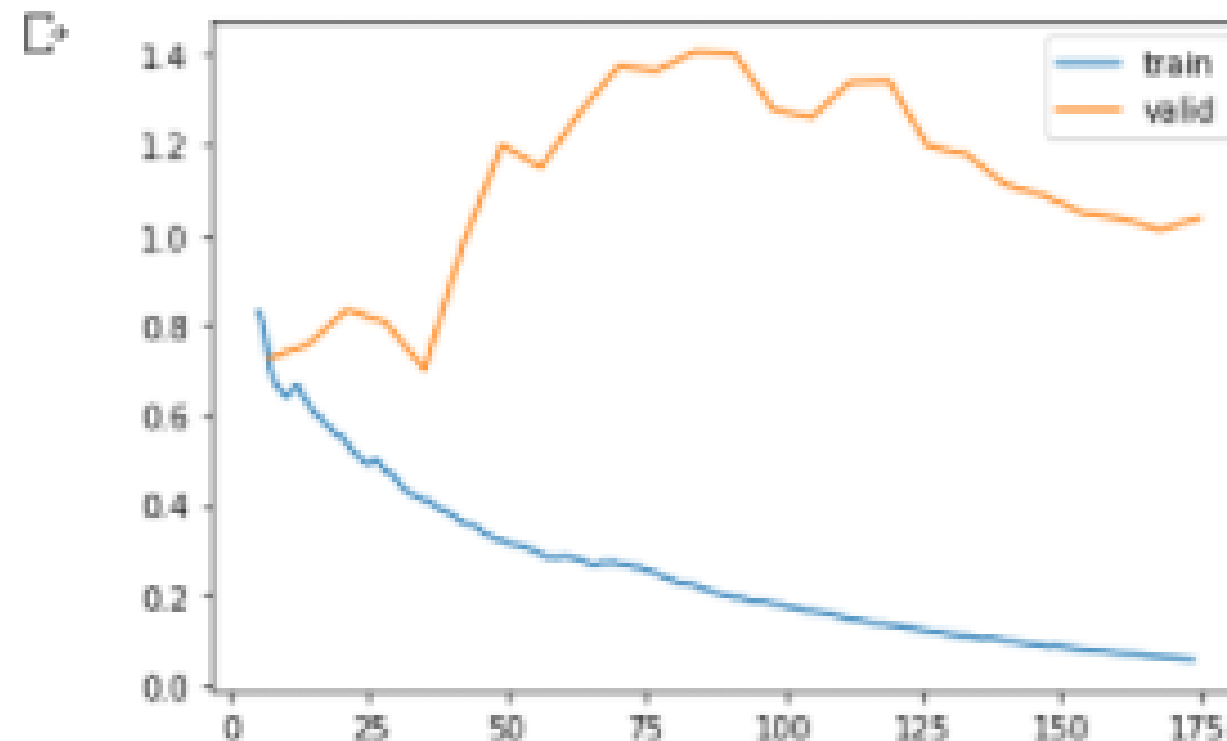
| epoch | train_loss | valid_loss | accuracy | time |
|------------------------------------------------------------------------|------------|------------|----------|-------|
| 0 | 1.104324 | 1.248124 | 0.638655 | 03:24 |
| Better model found at epoch 0 with accuracy value: 0.6386554837226868. | | | | |
| epoch | train_loss | valid_loss | accuracy | time |
| 0 | 0.791878 | 0.725605 | 0.731092 | 01:27 |
| 1 | 0.642833 | 0.756295 | 0.689076 | 01:26 |
| 2 | 0.552567 | 0.833321 | 0.722689 | 01:25 |
| 3 | 0.491387 | 0.803784 | 0.764706 | 01:26 |
| 4 | 0.415786 | 0.700157 | 0.789916 | 01:26 |
| 5 | 0.368538 | 0.978428 | 0.806723 | 01:26 |
| 6 | 0.323524 | 1.198589 | 0.739496 | 01:25 |
| 7 | 0.298844 | 1.149465 | 0.773109 | 01:25 |
| 8 | 0.281754 | 1.269554 | 0.731092 | 01:25 |
| 9 | 0.274202 | 1.372445 | 0.756303 | 01:26 |
| 10 | 0.252768 | 1.363774 | 0.781513 | 01:26 |
| 11 | 0.223226 | 1.406142 | 0.781513 | 01:25 |
| 12 | 0.196911 | 1.402550 | 0.781513 | 01:25 |
| 13 | 0.182634 | 1.278312 | 0.764706 | 01:25 |
| 14 | 0.164521 | 1.259254 | 0.789916 | 01:25 |
| 15 | 0.148690 | 1.338423 | 0.789916 | 01:25 |
| 16 | 0.135079 | 1.340633 | 0.773109 | 01:26 |
| 17 | 0.121161 | 1.196320 | 0.789916 | 01:25 |
| 18 | 0.107454 | 1.178101 | 0.789916 | 01:25 |
| 19 | 0.099809 | 1.110856 | 0.764706 | 01:26 |
| 20 | 0.087699 | 1.087619 | 0.773109 | 01:25 |
| 21 | 0.079668 | 1.048067 | 0.773109 | 01:25 |
| 22 | 0.070403 | 1.035717 | 0.764706 | 01:25 |
| 23 | 0.063098 | 1.010946 | 0.781513 | 01:25 |
| 24 | 0.055643 | 1.036811 | 0.781513 | 01:25 |

```
# Train with resnet50
```

```
learn = cnn_learner(dls, resnet50, metrics=accuracy)  
learn.fine_tune(25, cbs=[SaveModelCallback(monitor='accuracy')])
```

```
Better model found at epoch 0 with accuracy value: 0.7310924530029297.  
Better model found at epoch 3 with accuracy value: 0.7647058963775635.  
Better model found at epoch 4 with accuracy value: 0.7899159789085388.  
Better model found at epoch 5 with accuracy value: 0.8067227005958557.
```

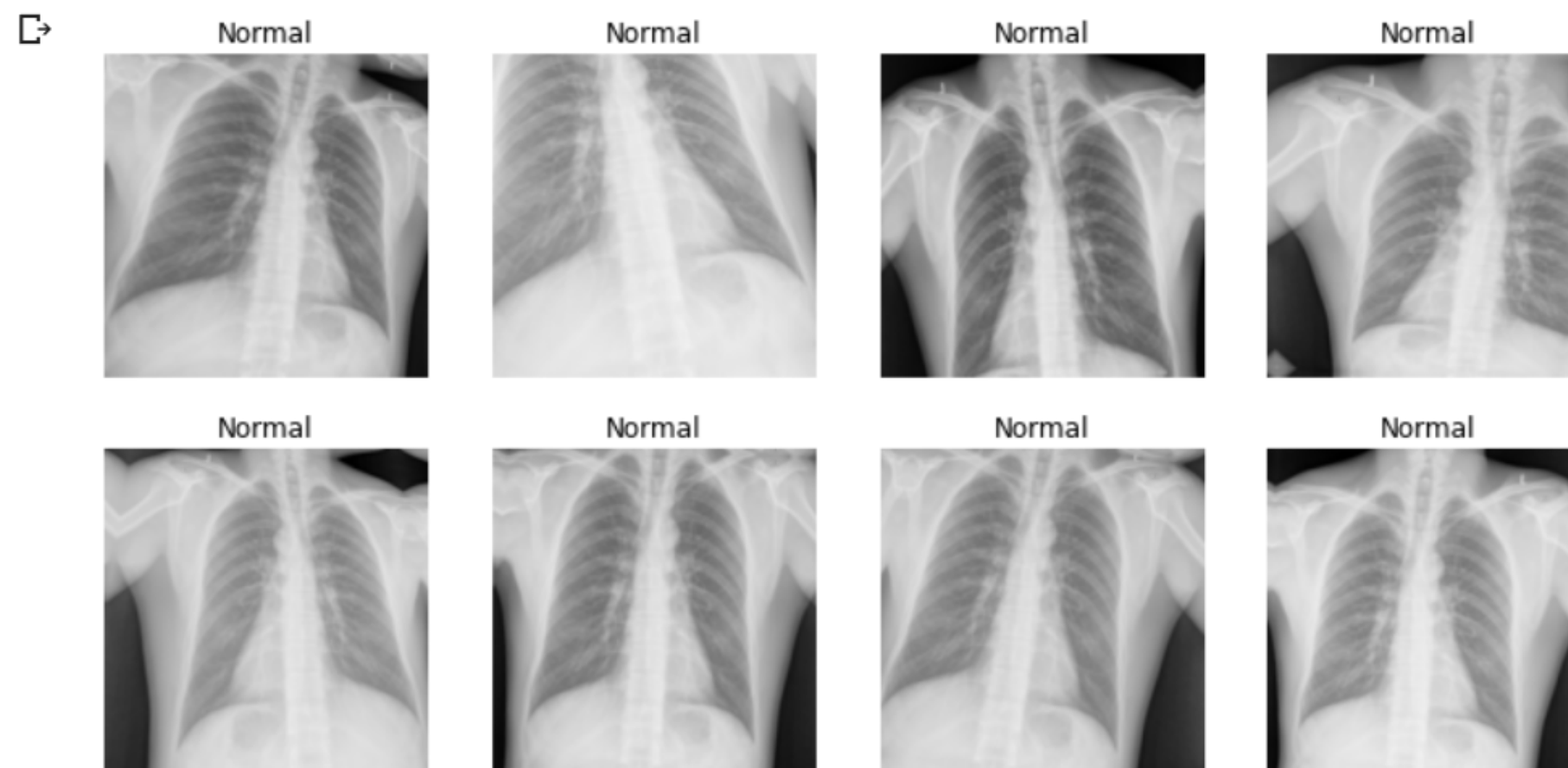
```
learn.recorder.plot_loss()
```



7. Model ที่ 2 train model โดยใช้เทคนิค data augmentation ผ่านฟังก์ชัน `aug_transforms()` ซึ่งจะทำการ flip / rotate / zoom / warp / lighting transforms ภาพ

Try Data Augmentation

```
▶ xray = xray.new(item_tfms=RandomResizedCrop(224, min_scale=0.5),  
    batch_tfms=aug_transforms())  
dls = xray.data loaders(path)  
dls.train.show_batch(max_n=8, n_rows=2, unique=True)
```



จากนั้น train โดยใช้ resnet50 architecture และใช้มาตรวัดเป็น Accuracy ซึ่งจะได้ Accuracy อยู่ราว ๆ 87%

| epoch | train_loss | valid_loss | accuracy | time |
|-------|------------|------------|----------|------|
|-------|------------|------------|----------|------|

| | | | | |
|---|----------|----------|----------|-------|
| 0 | 1.027776 | 2.048182 | 0.546219 | 01:26 |
|---|----------|----------|----------|-------|

Better model found at epoch 0 with accuracy value: 0.5462185144424438.

| epoch | train_loss | valid_loss | accuracy | time |
|-------|------------|------------|----------|------|
|-------|------------|------------|----------|------|

| | | | | |
|---|----------|----------|----------|-------|
| 0 | 0.896086 | 0.993757 | 0.630252 | 01:24 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 1 | 0.871606 | 0.812162 | 0.672269 | 01:27 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 2 | 0.856909 | 0.734859 | 0.672269 | 01:26 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 3 | 0.787629 | 0.972290 | 0.630252 | 01:28 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 4 | 0.762171 | 0.974826 | 0.697479 | 01:24 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 5 | 0.738371 | 1.733675 | 0.663866 | 01:13 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 6 | 0.728688 | 1.312600 | 0.630252 | 01:13 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 7 | 0.705005 | 0.940557 | 0.747899 | 01:14 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 8 | 0.677024 | 0.658952 | 0.789916 | 01:14 |
|---|----------|----------|----------|-------|

| | | | | |
|---|----------|----------|----------|-------|
| 9 | 0.643308 | 0.713808 | 0.823529 | 01:14 |
|---|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 10 | 0.609806 | 0.568362 | 0.873950 | 01:14 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 11 | 0.574207 | 0.744086 | 0.865546 | 01:14 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 12 | 0.551851 | 0.582402 | 0.840336 | 01:13 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 13 | 0.520737 | 0.428576 | 0.823529 | 01:14 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 14 | 0.481446 | 0.406031 | 0.848740 | 01:14 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 15 | 0.452716 | 0.387631 | 0.831933 | 01:13 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 16 | 0.429192 | 0.392395 | 0.840336 | 01:14 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 17 | 0.408252 | 0.391165 | 0.840336 | 01:13 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 18 | 0.387303 | 0.398885 | 0.831933 | 01:13 |
|----|----------|----------|----------|-------|

| | | | | |
|----|----------|----------|----------|-------|
| 19 | 0.372828 | 0.404448 | 0.848740 | 01:13 |
|----|----------|----------|----------|-------|

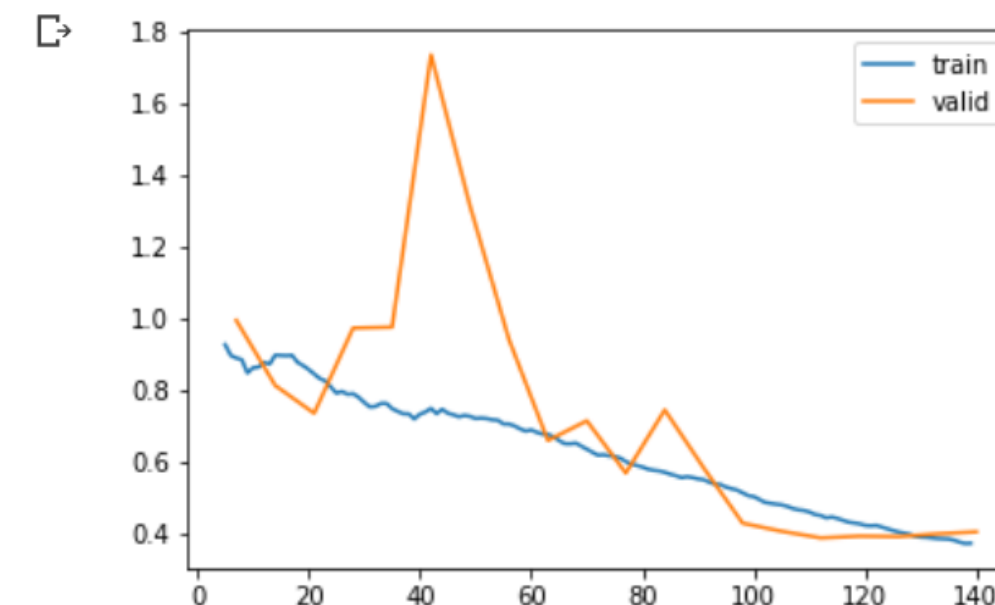
```
# Train with resnet50
```

```
learn = cnn_learner(dls, resnet50, metrics=accuracy)
```

```
learn.fine_tune(20, cbs=[SaveModelCallback(monitor='accuracy')])
```

```
Better model found at epoch 0 with accuracy value: 0.6302521228790283.  
Better model found at epoch 1 with accuracy value: 0.6722689270973206.  
Better model found at epoch 4 with accuracy value: 0.6974790096282959.  
Better model found at epoch 7 with accuracy value: 0.7478991746902466.  
Better model found at epoch 8 with accuracy value: 0.7899159789085388.  
Better model found at epoch 9 with accuracy value: 0.8235294222831726.  
Better model found at epoch 10 with accuracy value: 0.8739495873451233.
```

```
learn.recorder.plot_loss()
```

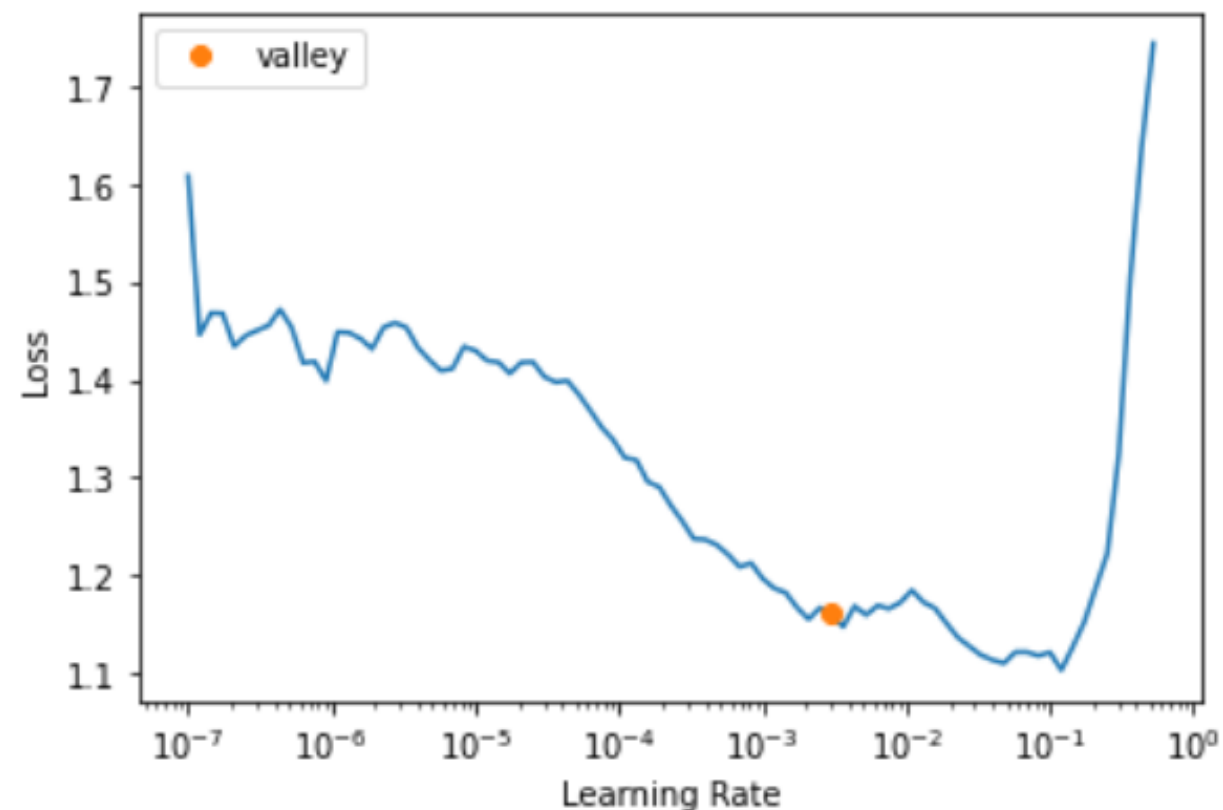


8. Model ที่ 3 train model โดยใช้เทคนิค data augmentation และ discriminative learning rate ได้ Accuracy อยู่ที่ 90%

Learning rate finder

```
# create learner  
learn = cnn_learner(dls, resnet50, metrics=accuracy)  
  
learn.lr_find()
```

➞ SuggestedLRs(valley=0.0030199517495930195)



- เริ่มจากการทำ Learning rate finder เราจะเลือก learning rate ที่ค่า loss ลดลงด้วยความชันมากที่สุด

```
learn = cnn_learner(dls, resnet50, metrics=accuracy)
learn.fit_one_cycle(3, 0.0030199517495930195)
```

| epoch | train_loss | valid_loss | accuracy | time |
|-------|------------|------------|----------|-------|
| 0 | 1.193958 | 2.919961 | 0.529412 | 01:11 |
| 1 | 1.274510 | 1.226995 | 0.680672 | 01:11 |
| 2 | 1.153265 | 1.076876 | 0.663866 | 01:11 |

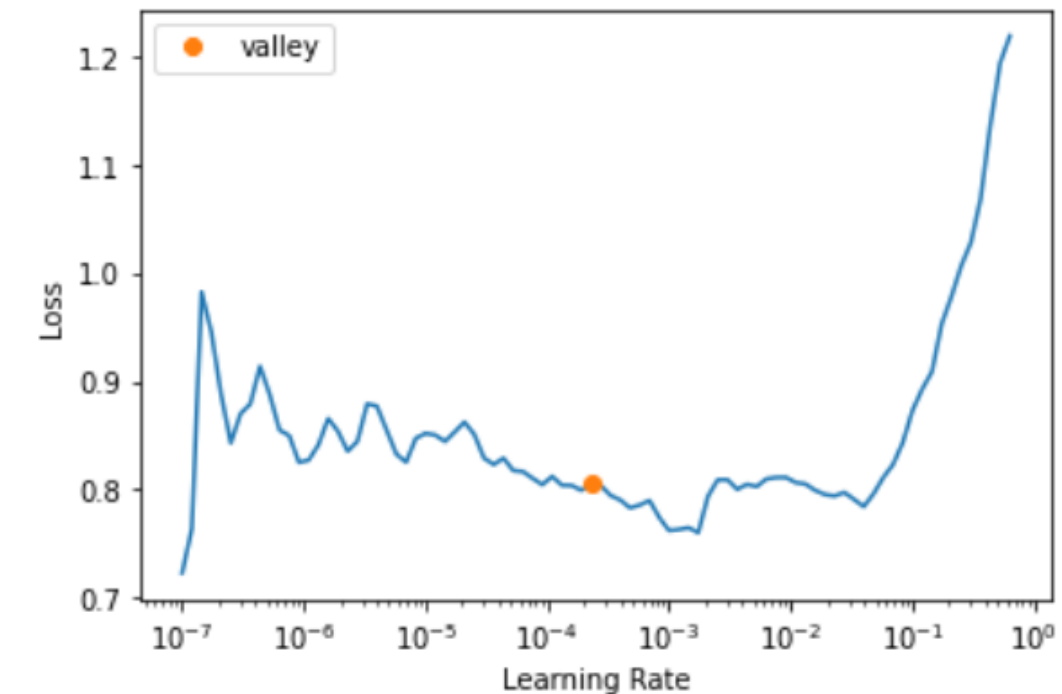
- จากนั้น train model ด้วย learning rate ที่เหมาะสมเฉพาะ layer ของ classifier และ freeze weight ของ layer ก่อนหน้าไว้ทั้งหมด , train ด้วย จำนวน epoch น้อยๆ ประมาณ 1-3 epoch

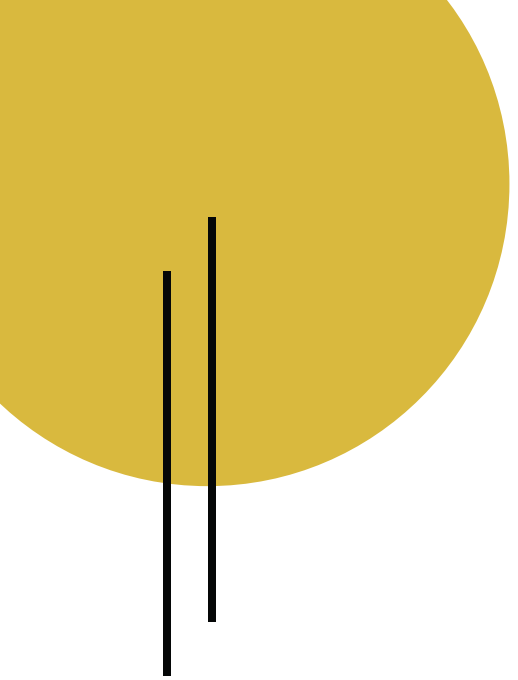
- Unfreeze weight ใน layer ก่อนหน้า classifier จากนั้นทำ learning rate finder อีกรอบ

Unfreeze and find learning rate again

```
[ ] learn.unfreeze()
learn.lr_find()
```

SuggestedLRs (valley=0.0002290867705596611)

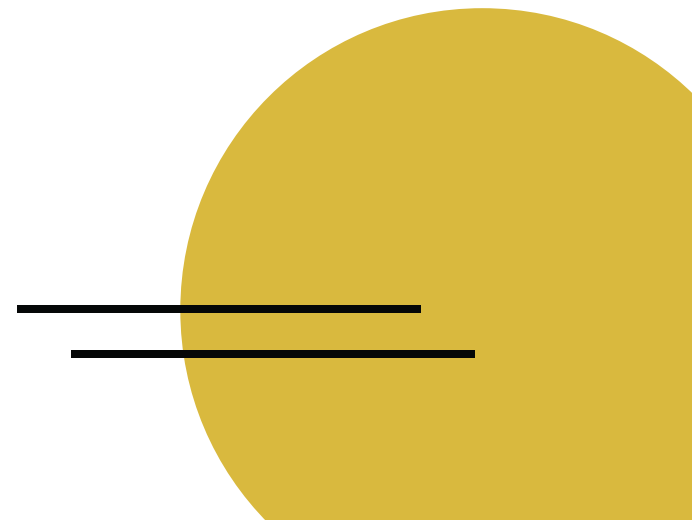




- ใช้เทคนิค discriminative learning rate โดยกำหนดให้ learning rate ใน layer ตื้นๆน้อยกว่า learning rate ใน layer ท้ายๆ ผ่าน function slice เริ่มจาก learning rate ที่ค่อยๆเพิ่มสูงขึ้น แล้วจึงค่อยๆลดลงต่ำกว่า learning rate เริ่มต้นในตอนแรก เพื่อเป็นการ warm up ช่วงแรก แล้วจึงค่อยๆเข้าสู่จุดที่ weight ของ model เริ่มเสถียรด้วยการลด learning rate ลง

Discriminative learning rate

```
[ ] learn = cnn_learner(dls, resnet50, metrics=accuracy)
learn.fit_one_cycle(3, 0.0014454397605732083)
learn.unfreeze()
learn.fit_one_cycle(25, lr_max=slice(0.0002290867705596611,6e-4), cbs=[SaveModelCallback(monitor='accuracy')])
```

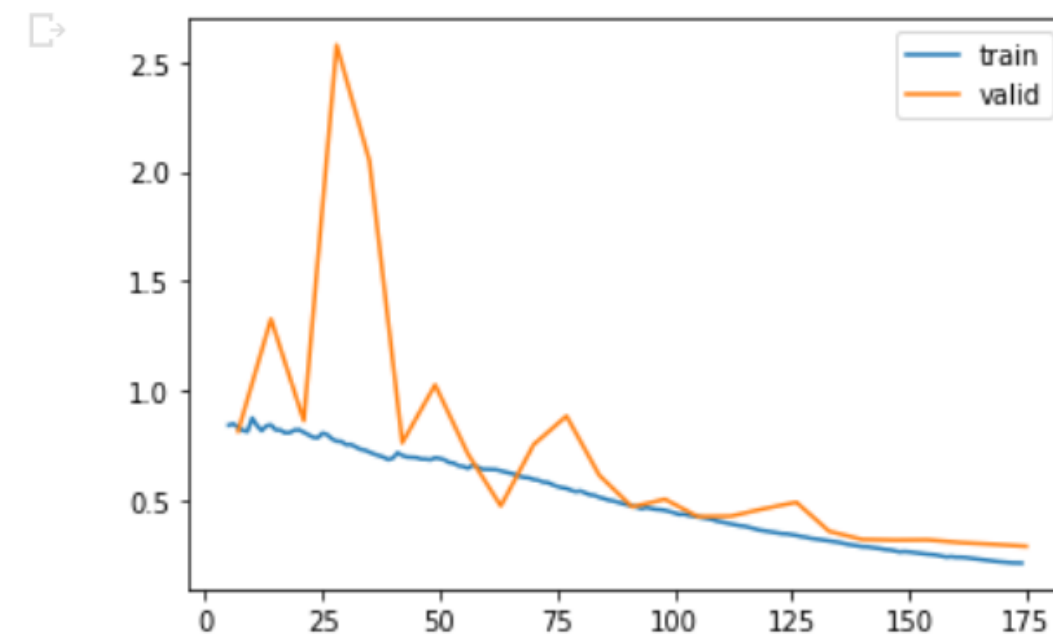


| epoch | train_loss | valid_loss | accuracy | time |
|-------|------------|------------|----------|-------|
| 0 | 1.058444 | 2.964665 | 0.512605 | 01:10 |
| 1 | 1.069575 | 0.818169 | 0.722689 | 01:09 |
| 2 | 0.997598 | 0.645952 | 0.747899 | 01:09 |
| epoch | train_loss | valid_loss | accuracy | time |
| 0 | 0.849305 | 0.815335 | 0.722689 | 01:12 |
| 1 | 0.839825 | 1.327891 | 0.683866 | 01:11 |
| 2 | 0.822035 | 0.863419 | 0.722689 | 01:11 |
| 3 | 0.780488 | 2.577646 | 0.621849 | 01:11 |
| 4 | 0.729393 | 2.048433 | 0.638655 | 01:12 |
| 5 | 0.716908 | 0.763276 | 0.789916 | 01:11 |
| 6 | 0.685510 | 1.027247 | 0.840336 | 01:10 |
| 7 | 0.654988 | 0.711527 | 0.739496 | 01:10 |
| 8 | 0.640297 | 0.474686 | 0.815126 | 01:10 |
| 9 | 0.603115 | 0.754731 | 0.773109 | 01:11 |
| 10 | 0.558611 | 0.884837 | 0.747899 | 01:10 |
| 11 | 0.522655 | 0.614549 | 0.815126 | 01:10 |
| 12 | 0.477915 | 0.470637 | 0.823529 | 01:10 |
| 13 | 0.457742 | 0.505300 | 0.798319 | 01:11 |
| 14 | 0.427973 | 0.426215 | 0.831933 | 01:10 |
| 15 | 0.398806 | 0.427860 | 0.873950 | 01:10 |
| 16 | 0.365211 | 0.461819 | 0.823529 | 01:10 |
| 17 | 0.343987 | 0.491747 | 0.806723 | 01:11 |
| 18 | 0.317866 | 0.359029 | 0.882353 | 01:10 |
| 19 | 0.292821 | 0.322032 | 0.907563 | 01:11 |
| 20 | 0.274094 | 0.319279 | 0.882353 | 01:11 |
| 21 | 0.256620 | 0.320080 | 0.865546 | 01:11 |
| 22 | 0.240841 | 0.307547 | 0.873950 | 01:11 |
| 23 | 0.225372 | 0.299753 | 0.873950 | 01:12 |
| 24 | 0.214332 | 0.290667 | 0.873950 | 01:12 |

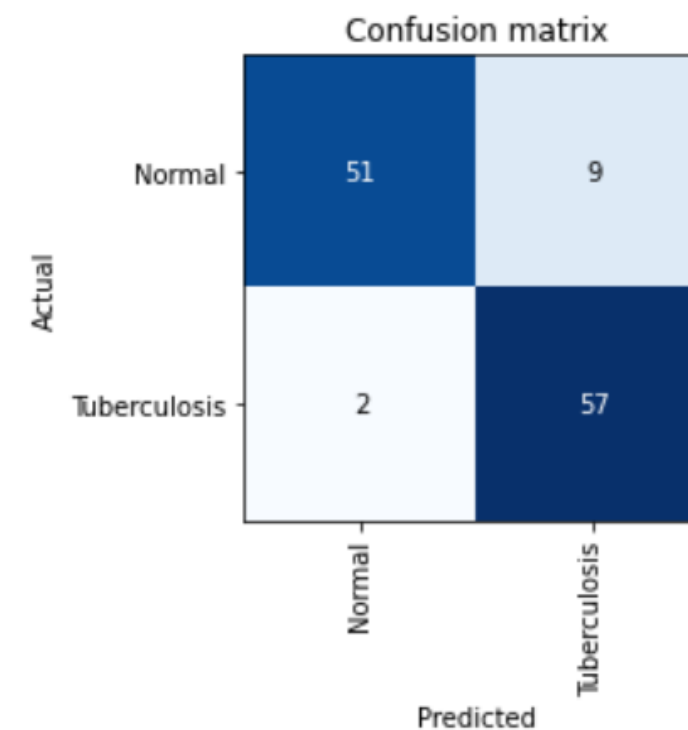
- ผลลัพธ์การ Train model ที่ 3 บน
validation set ได้ Accuracy ประมาณ 90 %

Better model found at epoch 0 with accuracy value: 0.7226890921592712.
 Better model found at epoch 5 with accuracy value: 0.7899159789085388.
 Better model found at epoch 6 with accuracy value: 0.8403361439704895.
 Better model found at epoch 15 with accuracy value: 0.8739495873451233.
 Better model found at epoch 18 with accuracy value: 0.8823529481887817.
 Better model found at epoch 19 with accuracy value: 0.9075630307197571.

```
[ ] learn.recorder.plot_loss()
```



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

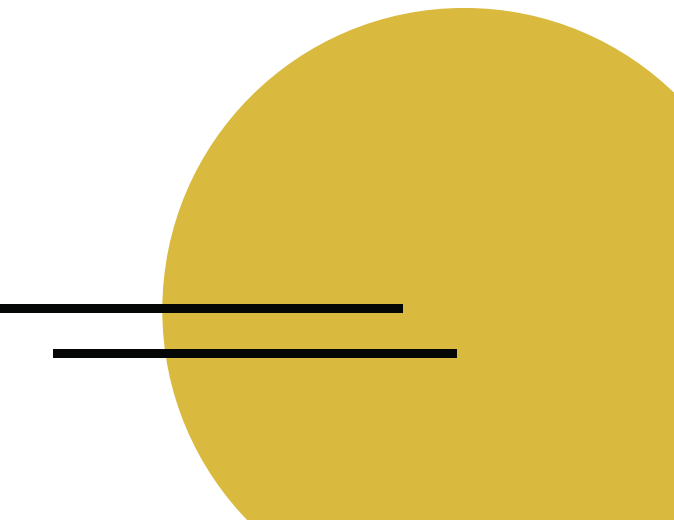




ผลลัพธ์ของงาน

จากการที่ได้ลอง Train Model ทั้ง 3 รูปแบบ ได้ผลลัพธ์ว่า model ที่ 3 ได้ Accuracy ที่ทดสอบใน validation set สูงที่สุด อยู่ที่ 90% ซึ่งเป็น model ที่ใช้เทคนิค data augmentation และ discriminative learning rate

เราจึงเลือก model นี้มาใช้ทดสอบกับข้อมูล test set ของเราต่อ ซึ่งผลลัพธ์ของการ Predict รูปแบบ test set ได้ Accuracy อยู่ที่ 90.769%



ผลลัพธ์ของงาน

Use third model to predict the test set

```
[ ] learner = load_learner("/content/gdrive/MyDrive/Colab Notebooks/models/thirdModel.pkl")
```

```
▶ path = Path("/content/gdrive/MyDrive/Colab Notebooks/test set")
  all_images = get_image_files(path)

  correct_pred = 0
  wrong_pred = 0

  for im in all_images:
    result = learner.predict(im)[0][:]
    if(result == "Normal" and im.name[-5] == "0"):
      correct_pred += 1
    elif(result == "Tuberculosis" and im.name[-5] == "1"):
      correct_pred += 1
    else:
      wrong_pred += 1

  print("correct_predict = ", correct_pred)
  print("wrong_predict = ", wrong_pred)
  print("accuracy = " + str( (correct_pred/(correct_pred+wrong_pred))*100 ) + " %" )
```

```
↳ correct_predict = 59
   wrong_predict = 6
   accuracy = 90.76923076923077 %
```



Thank you!

