Experimenting with the method Framework

May 12, 2022

1 Continuous ensemble training

Now that the data, the network frameworks and the training loops are set up, we can investigate fruther.

2 Setting everything up

[11]:
```
# Reload module in case of changes
importlib.reload(utils)
```

[11]:
```
<module 'lib.utils' from
'/net/projects/scratch/winter/valid_until_31_July_2022/fheitzer/BAThesis-
code/notebooks/../lib/utils.py'>
```

[ ]:
```
import tensorflow as tf
import numpy as np
import pandas as pd

import sys; sys.path.insert(0, '..')
import importlib

from lib import data, networks, training, utils
```

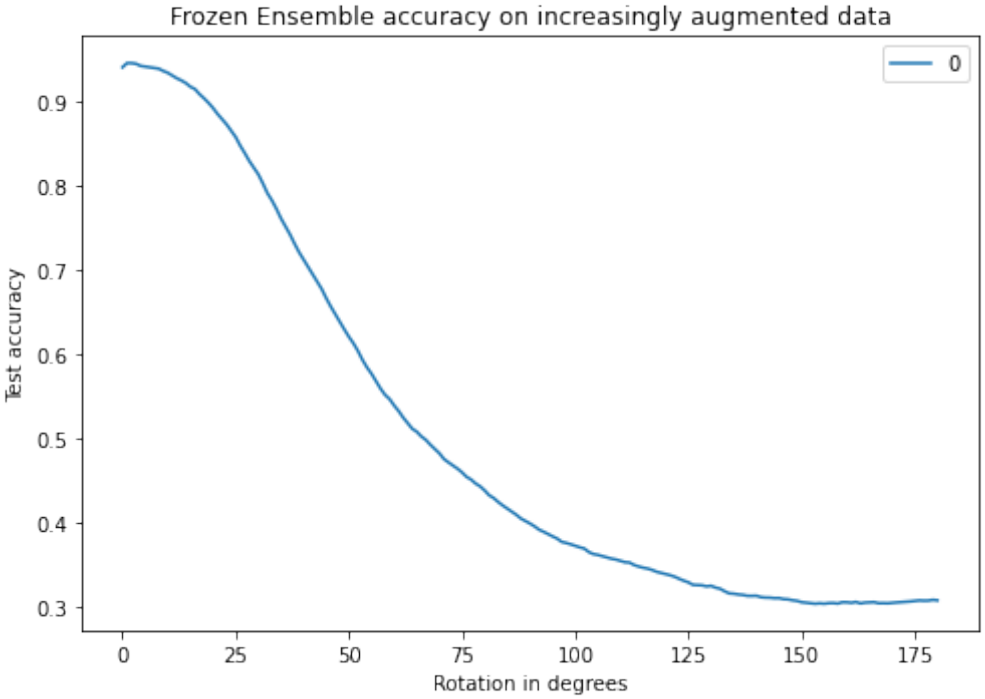
[ ]:
```
# load 10 class data
train_ds_pre, train_ds_post, test_ds, train_generator, test_generator = data.load_data(rotation=30)
dataset_shape = (tf.TensorSpec(shape=(28,28,1), dtype=tf.float64),
                 tf.TensorSpec(shape=(10,), dtype=tf.float32),
```

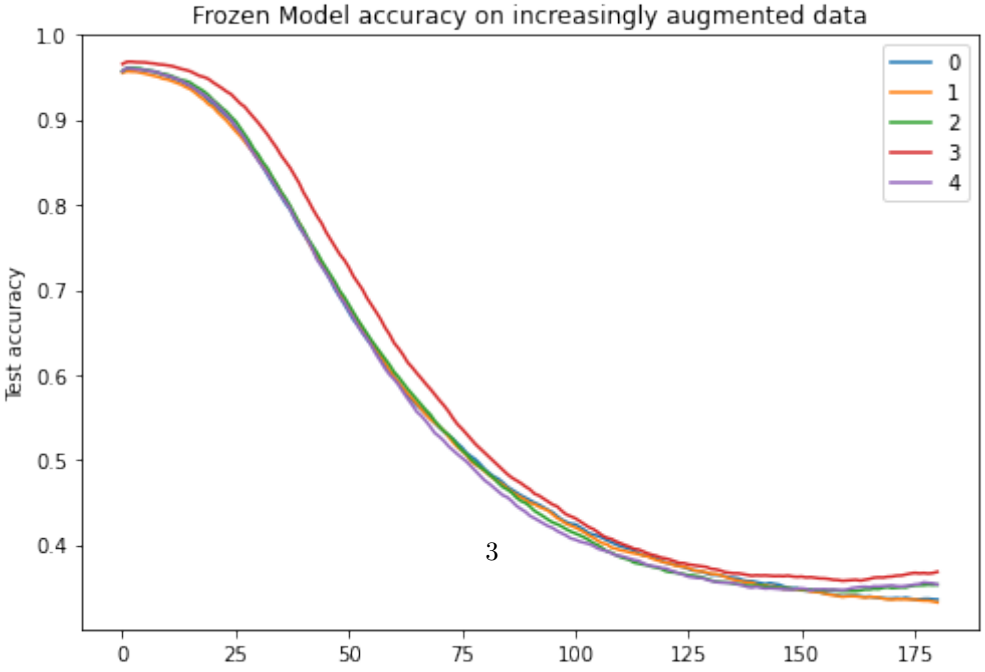
```
                tf.TensorSpec(shape=(), dtype=tf.int32),
                tf.TensorSpec(shape=(10,), dtype=tf.float32))

num_classes = 10
# Small model
model1 = networks.NN([128, 128], num_classes)
# Broad Model
model2 = networks.NN([512], num_classes)
# Mixed Model
#model3a = networks.NN([256, 256], num_classes)

model3b = networks.CNN([(32, 3), (64, 5)])
# cnn
model4 = networks.CNN([(32, 3), (64, 5), (128, 7)], num_classes)
# cnn small
model5 = networks.CNN([(16, 3), (32, 3), (64, 5)], num_classes)
# ensemble
ensemble = networks.Ensemble([model1, model2, model3b, model4, model5])

model1.load_weights('../models/NN128128extra')
model2.load_weights('../models/NN512extra')
model3b.load_weights('../models/CNN3264extra')
model4.load_weights('../models/CNN3264128extra')
model5.load_weights('../models/CNN163264extra')
```

3 Investigation

3.1 What happens if the model has to face increasingly augmented data?

[11]:
```
utils.plot_frozen_model('Frozenmodel_r180_c180')
```


Frozen Ensemble accuracy on increasingly augmented data
0
Test accuracy
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
25
50
75
100
125
150
175
Rotation in degrees


Frozen Model accuracy on increasingly augmented data
0
1
2
3
4
Test accuracy
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
25
50
75
100
125
150
175

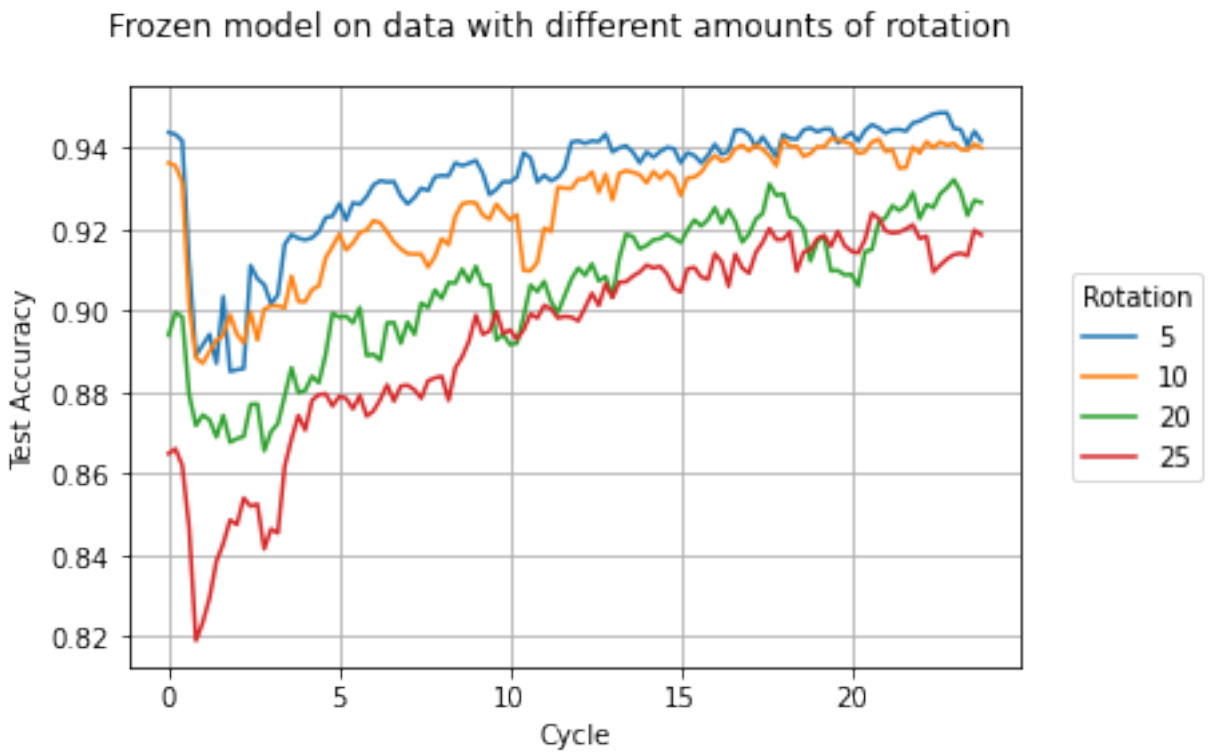
3.2 We should apply continuous ensemble training!

Starting off with a cycle size of 15.000

3.2.1 How does the method react to different jumps in augmentation?

[24]:
```
jumpfiles = ["Jump_r5_e1_b1_c24_d15000",
             "Jump_r10_e1_b1_c24_d15000",
             "Jump_r20_e1_b1_c24_d15000",
             "Jump_r25_e1_b1_c24_d15000"]
```

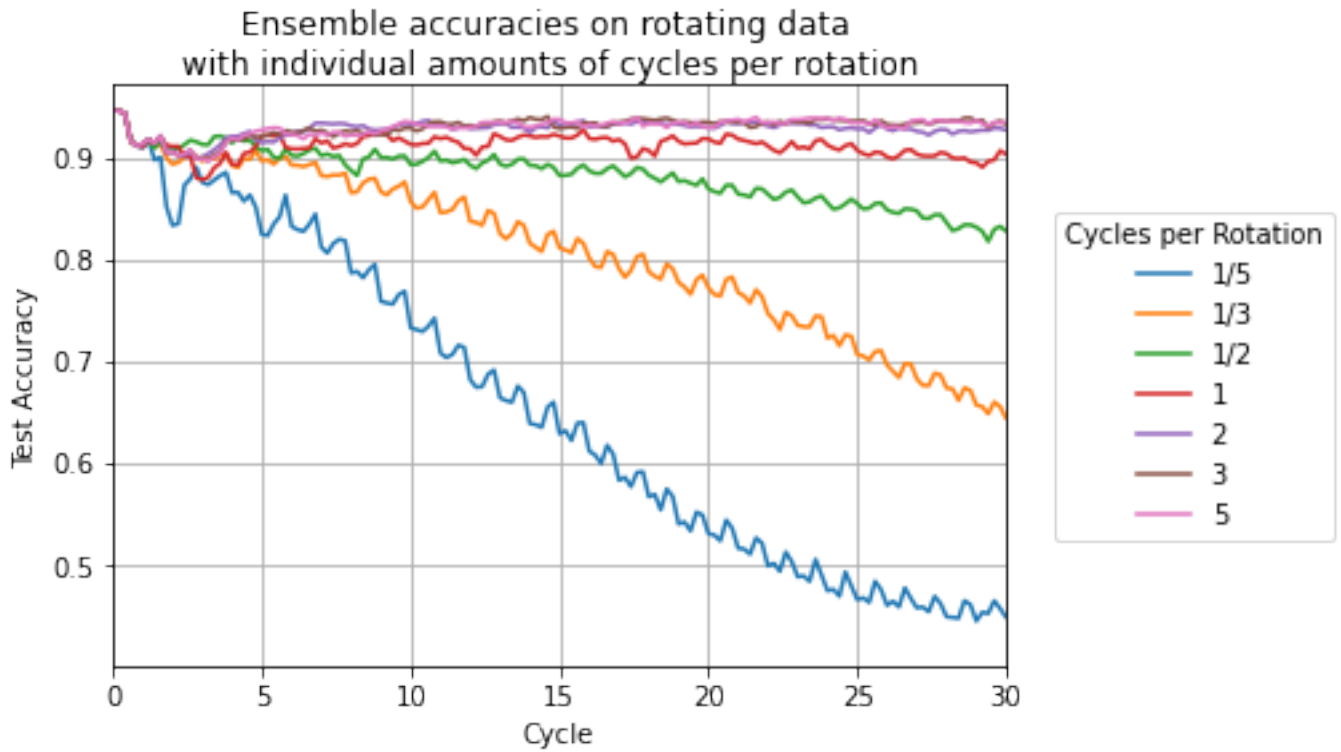
[32]:
```
utils.plot_multiple_ensemble_accuracies(jumpfiles, "Jump")
```



3.2.2 What augmentation speed can the method handle?

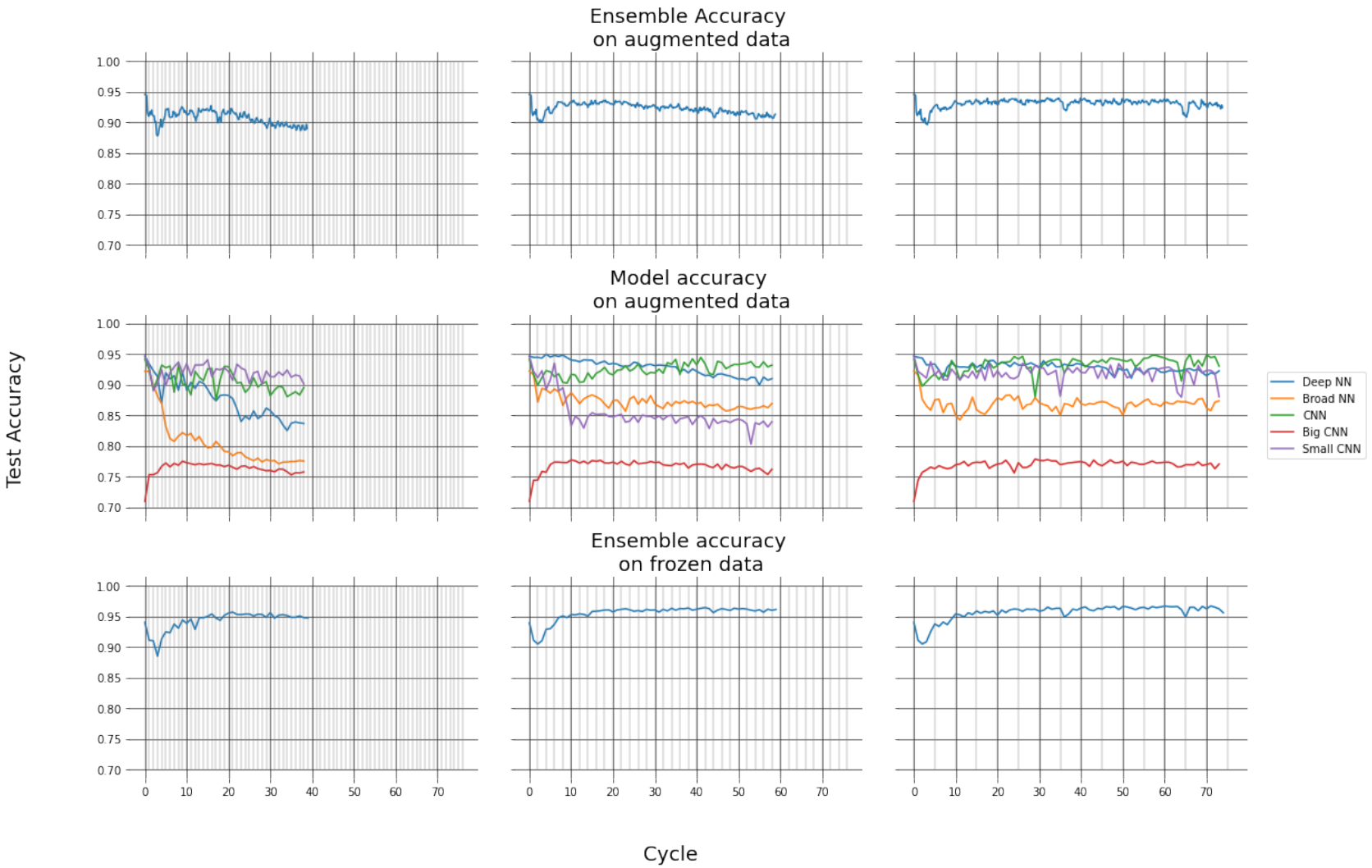
[28]:
```
files = ["Increment_r180_e1_b1_c36_d15000",
         "Increment_r180_e1_b1_c60_d15000",
         "Increment_r180_e1_b1_c90_d15000",
         "Increment_r90_e1_b1_c90_d15000",
         "Increment_r180_e1_b1_c360_d15000",
         "Increment_r30_e1_b1_c90_d15000",
         "Increment_r180_e1_b1_c900_d15000"]
```

[30]:
```
utils.plot_multiple_ensemble_accuracies(files, "Increment", xlim=30)
```



3.2.3 Does the continously trained ensemble still perform well on the original data and how do the individual models cope with the augmentation?

[57]:
```
utils.plot_cycle_accuracies_grid(files[3:-2] + [files[-1]])
```

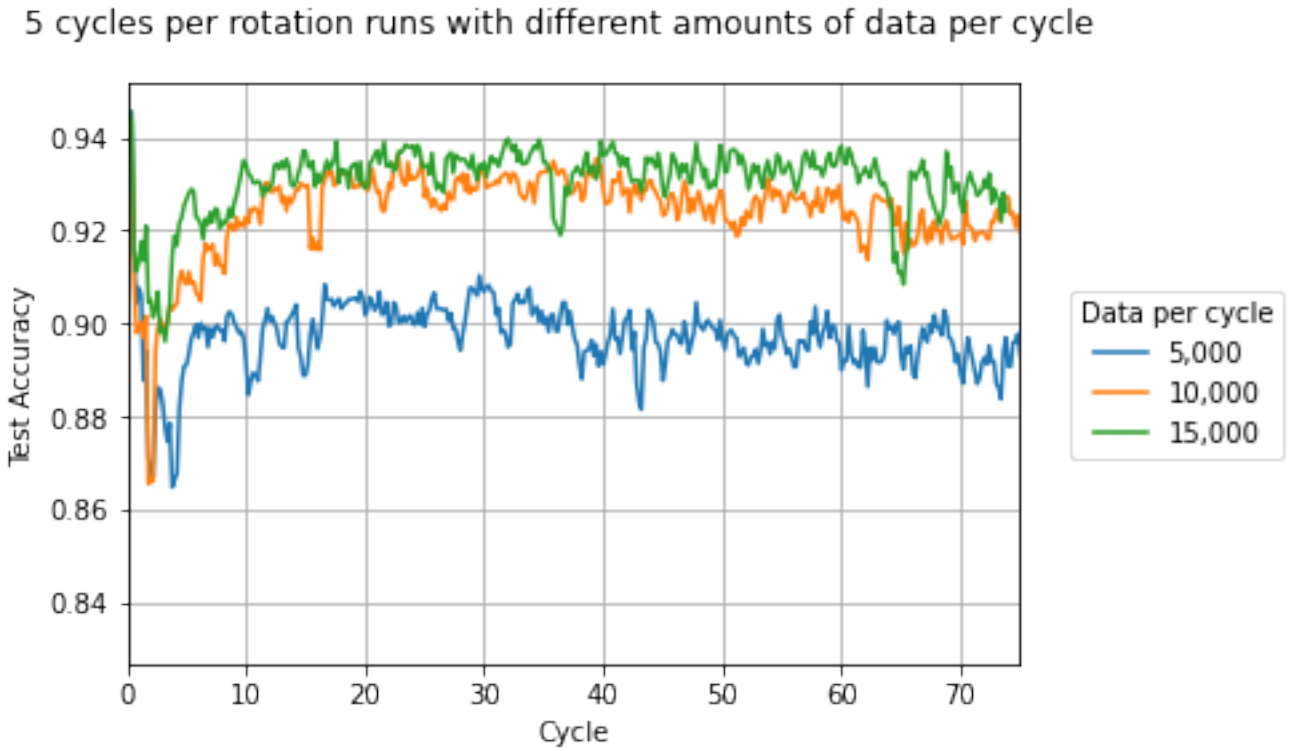

Each column shows a separate simulation with 1, 2, and 5 cycles per rotation as indicated by the background grid
Ensemble Accuracy
on augmented data
Model accuracy
on augmented data
Ensemble accuracy
on frozen data
Test Accuracy
Cycle
Deep NN
Broad NN
CNN
Big CNN
Small CNN

Continuous ensemble training allows for stable performance in a changing environment whithout a decrease in performance on the original data.

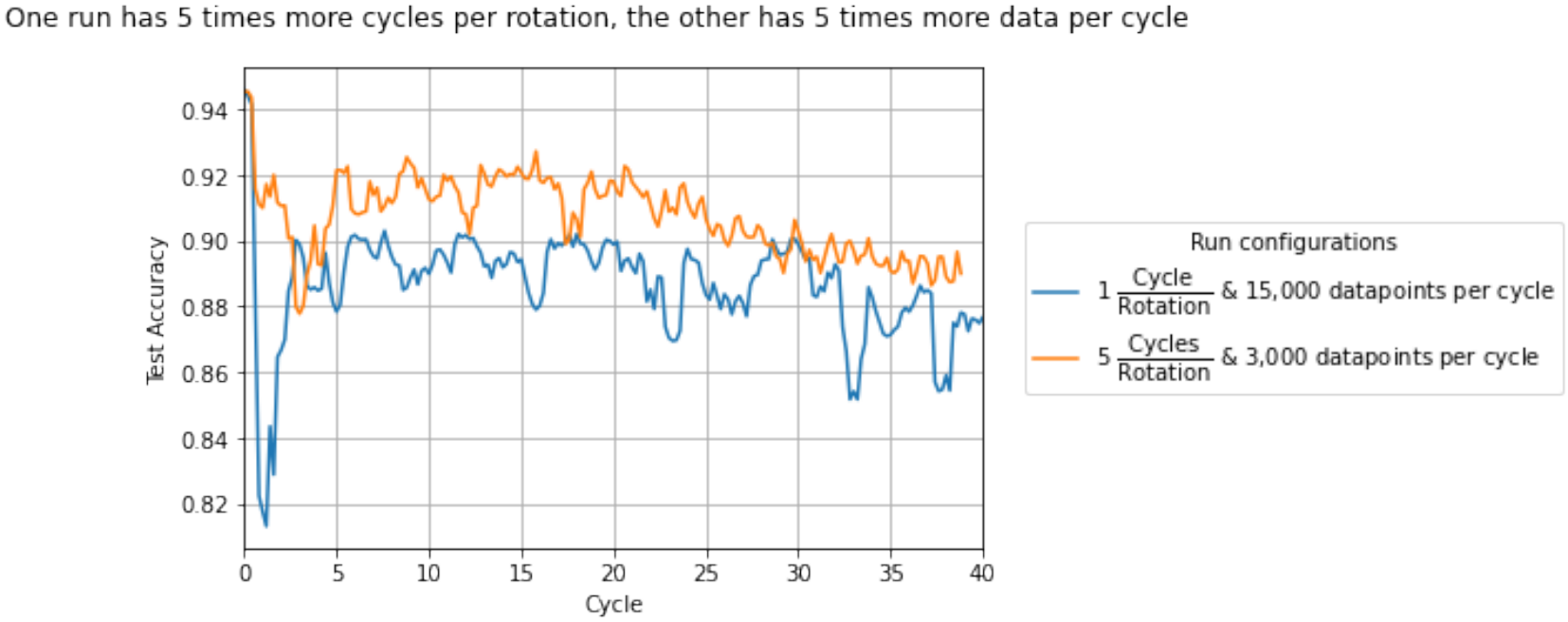
3.3 What if there is less or more data per cycle?

[27]:
```
paths = ["datapercyclecomparison_r20_e1_b1_c100_d3000",
         "Increment_r90_e1_b1_c90_d15000",
         "lessdata_r50_e1_b1_c250_d5000",
         "lessdata_r50_e1_b1_c250_d10000",
         "Increment_r180_e1_b1_c900_d15000"]
```

[13]:
```
utils.plot_multiple_ensemble_accuracies(paths[2:], "5cr_comparison", xlim=75)
```



[32]:
```
utils.plot_multiple_ensemble_accuracies(paths[:2], "1cr15k_5cr3k", xlim=40)
```



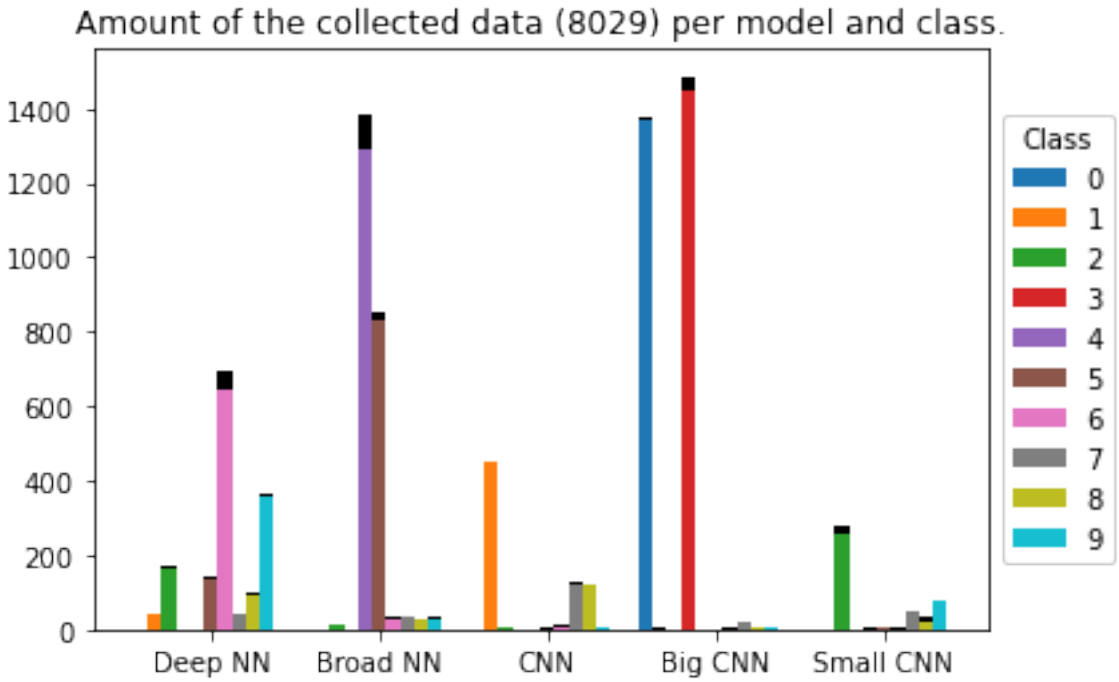
3.4 What does the collected data show?

3.4.1 $1\frac{\text{Cycle}}{\text{Degree of rotation}}$

(90 degrees in 39/90 cycles)

[20]:
```
utils.plot_cycles_oneline(ensemble,
                          "Increment_r90_e1_b1_c90_d15000", only_some=[-2,-1])
```

```
Cycle:  0
7701.0 collected datapoints labeled correct
328.0 collected datapoints were labeled wrong
544 datapoints were not classified.
```

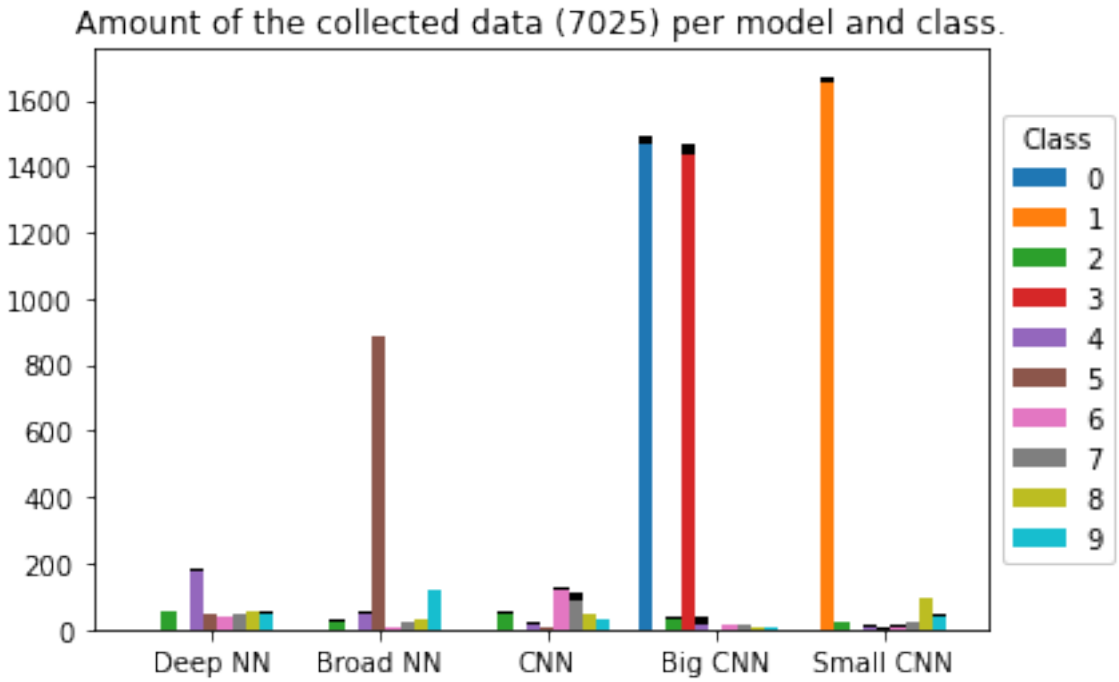


3.4.2 $2\frac{\text{Cycles}}{\text{Degree of rotation}}$

(180 degrees in 59/360 cycles)

[19]:
```
utils.plot_cycles_oneline(ensemble,
                          "Increment_r180_e1_b1_c360_d15000", only_some=[-2,-1])
```

```
Cycle:  0
6787.0 collected datapoints labeled correct
238.0 collected datapoints were labeled wrong
271 datapoints were not classified.
```

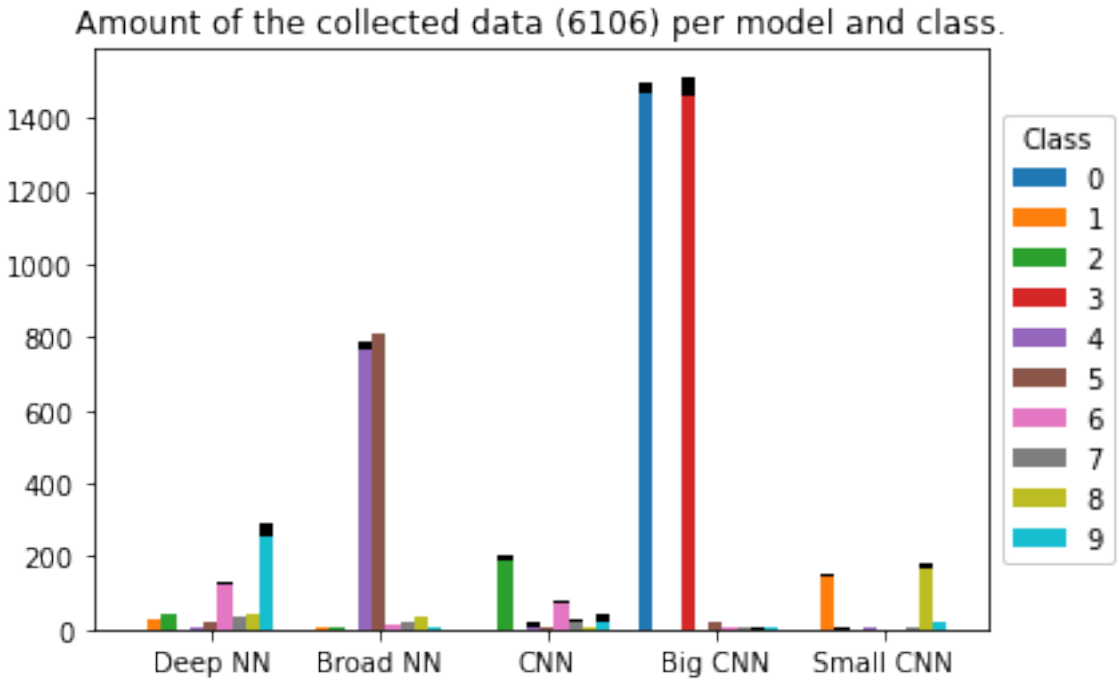


3.4.3 $3\frac{\text{Cycles}}{\text{Degree of rotation}}$

(30 degrees in 48/90 cycles)

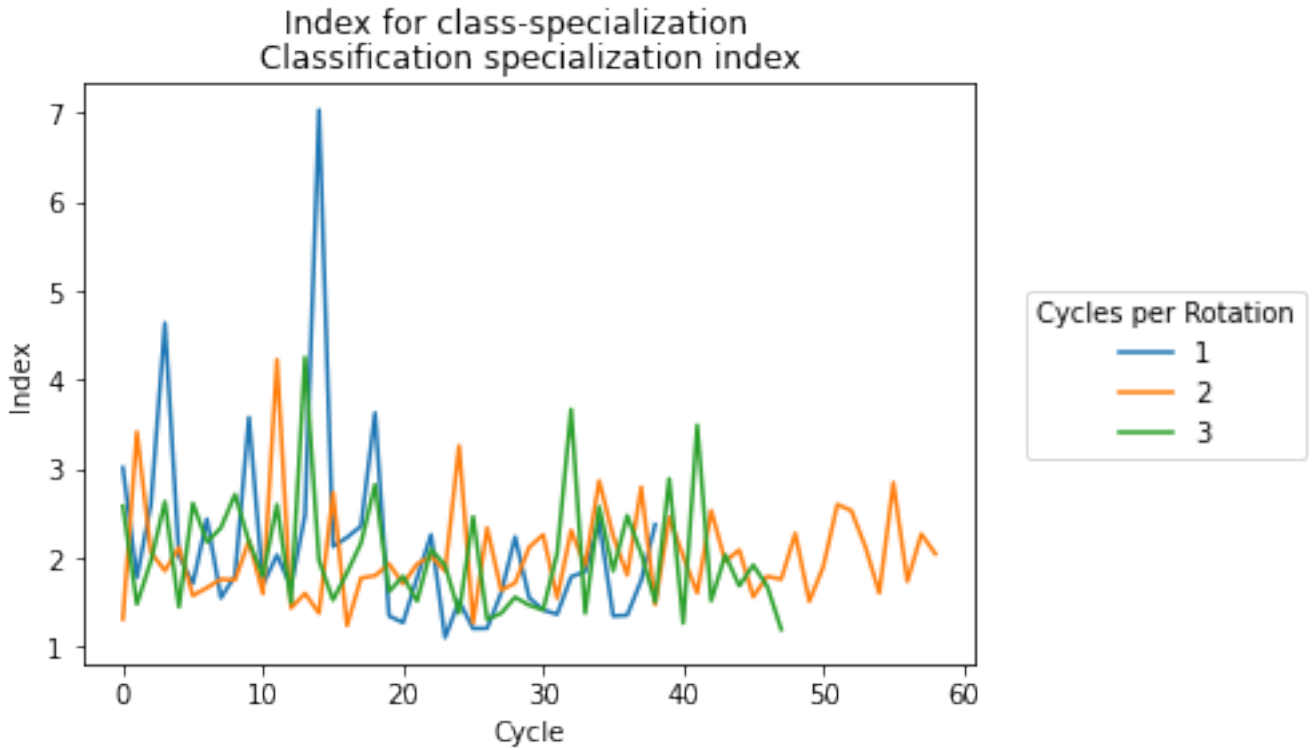
[18]:
```
utils.plot_cycles_oneline(ensemble,
                          "Increment_r30_e1_b1_c90_d15000", only_some=[-2,-1])
```

```
Cycle:  0
5872.0 collected datapoints labeled correct
234.0 collected datapoints were labeled wrong
187 datapoints were not classified.
```



3.5 Can we evaluate the specialization?

[55]:
```
utils.classification_specialization_mean(ensemble,
                                         ["Increment_r90_e1_b1_c90_d15000",
                                          "Increment_r180_e1_b1_c360_d15000",
                                          "Increment_r30_e1_b1_c90_d15000"],
                                          legend=["1","2","3"])
```


Index for class-specialization
Classification specialization index
Index
Cycle
0
10
20
30
40
50
60
1
2
3
4
5
6
7
Cycles per Rotation
1
2
3