Week 16

0. Table of Contents

1. Introduction

Now that we have set up the machine learning system and begun familiarising ourselves with its architecture, the aim of the next few weeks (this one included) is to begin experimenting on different training and validation data, as well as tuning the architecture.

2. Checklist

Our evolving checklist of things to explore regarding CNNs can be found here (Sheet 1). The highlight of what has been currently done and what is currently aimed at has been transcribed below.

- ☒ Data augmentation (rolled): noticeable improvement
- ☒ More samples: noticeable improvement
- ☒ Different alphas (training): noticeable improvement
- ☐ Gaps in alphas (training)
- ☐ Different densities
- ☐ Data augmentation (rotate)
- ☐ Tune learning rate/optimiser
- ☐ Tune batch size

3. Model List and Storage

In order to keep track of our many models, involving different layers and different data training sets, we opted to create a standardised table detailing the different parameters going into the system, such as training data snapshots, ratio of training to prediction data, etc. Below is a quick (preliminary) snapshot of the first few lines of the table, for reference. Note that a lot of columns have been skipped, which detail the utilised densities, tumbling rates and system sizes used for training and validation.

	RUN ID	# SNAPSHOTS	AUGMENTATION	OPTIMIZER	LEARNING RAT	BATCH SIZ	EPOCH	MAE	STD MI	STD AVG	STD MAX	OVERLAP R	R
2	headgear2718	20000 (0.2)	none	adam	0.001	64	10	0.02	0.001	0.05	0.02	0.80	0.98
3	atreus1273	70000 (0.29)	rolling (12)	adam				0.01					0.98
4	michigan2241	10000 (0.2)	none	adam				0.03	0.000	0.03	0.01		
5	kangaroo2519							0.07	0.038	0.07	0.05	0.90	0.78
6	sandpaper7571							0.20	0.000	0.06	0.01	0.20	0.89
7	corned1441						20	0.03	0.000	0.06	0.01	0.80	0.98
8	nearness8197						10	0.02	0.001	0.05	0.02	1.00	0.99
9	affected1468						20	0.04	0.001	0.05	0.02	0.80	0.97
10	implement6908						10	0.06	0.001	0.02	0.01	0.10	0.97
11	culture7803						20	0.02	0.001	0.06	0.02	0.70	0.98
12	subwoofer0187							0.02	0.001	0.06	0.02	0.70	0.98
13	gone0635						15	0.02	0.001	0.05	0.01	0.80	0.99
14	posh3633					32		0.02	0.002	0.05	0.02	0.70	0.97
15	multitask7949					64	30	0.04	0.001	0.06	0.02	0.40	0.96
16	awry4366	40000 (0.2)					10	0.02	0.001	0.05	0.02	0.80	0.98
17	lying1339						20	0.02	0.002	0.06	0.02	0.90	0.99
18	recite9661	6400 (0.2)	rolling (6)				70	0.00	5.93E-05	0.00	0.00	1.00	1.00
19	entrench0085	12800 (0.2)			0.0005		30	0.01	0.006	0.01	0.01	1.00	0.39
20	pruning0896				0.001			0.00	0.004	0.00	0.00	1.00	0.50
21	shamrock2212			sgd	0.0003 (def 0.01)		180	0.01	0.007	0.01	0.01	1.00	0.10
22	astronaut1046				0.005			0.01	0.011	0.02	0.01	1.00	0.98

The growing model list can be found here, on the same link as the checklist above (Sheet 2). Below are transcribed our current sought metrics (coloured in green within the table above).

- Maximum STD: 0.02
  - This is the maximum standard deviation a particular prediction spread can have. For example, given a model, the maximum STD determines the biggest spread in the output predictions (which are generated for some specific tumbling rates). Ideally the spreads would be similar across, but we can consider the learning to have failed if any individual prediction has too big a spread.
- Average STD: 0.01
  - This is the average standard deviation a particular prediction spread can have. Taking the same example as above, this averages over all predicted tumbling rates to obtain an average spread deviation. This naturally has to be small as well in order for the model to succeed.
- Mean Average Error: 0.01
- Pearson's Coefficient: 0.975
- Overlap Ratio: 1
  - This ratio indicates the amount of tumbling rate prediction spreads which overlap at all with the expected tumbling rates.

Combining a ubiquitous overlap ratio with a very small average and maximum standard deviation yields results which are both very accurate and very precise.

4. Shorthand Model List

While each model is referred to by the aforementioned standardisation, it is also useful to communicate architecture more quickly. Here's the current list of different architecturess

being used, all indexed by a model number (MN). The latter predictions generated in Weeks 13-15 are all done with MN_1, for example.

MN_1

1. CONV (filters=3,kernel_size=(3,3),padding='same',strides=(3,3),activation="relu", input_shape=shape)
2. BN
3. CONV (filters=3,kernel_size=(3,3),padding='same')
4. BN
5. MAXPOOL (pool_size=(3,3))
6. CONV (filters=6,kernel_size=(3,3),padding='same')
7. BN
8. CONV (filters=6,kernel_size=(3,3),padding='same')
9. BN
10. MAXPOOL (pool_size=(3,3))
11. FC (units=128,activation='relu')
12. DO (0.2) (without layout optimiser)
13. FC (units=10,activation='softmax')
14. FLATTEN
15. FC (units=1,activation='linear')

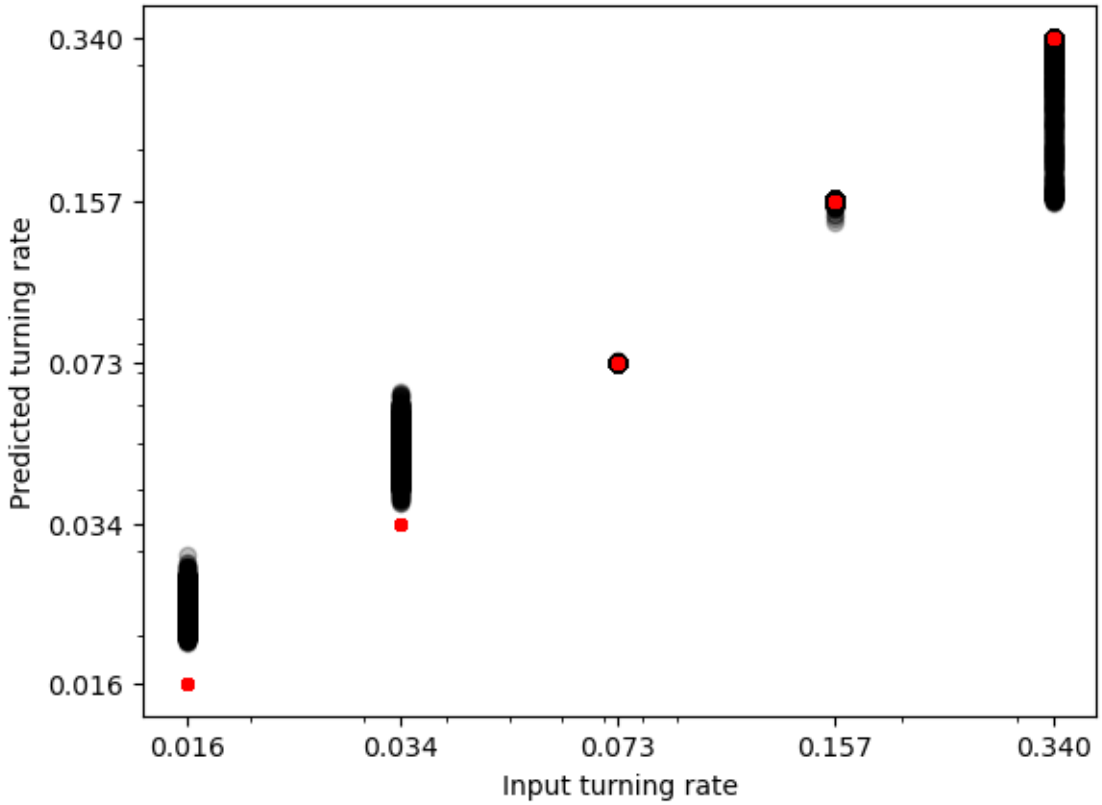
MN_2

1. CONV (filters=3,kernel_size=(3,3),padding='same',input_shape=shape)
2. MAXPOOL (pool_size=(2,2),padding='same')
3. ReLU
4. BN
5. CONV (filters=6,kernel_size=(5,5),padding='same')
6. MAXPOOL (pool_size=(2,2),padding='same')
7. ReLU
8. BN
9. AVGPOOL
10. DO (0.2) (without layout optimiser)
11. FC (units=64,activation='relu')
12. DO (0.2) (without layout optimiser)
13. FC (units=10,activation='softmax')
14. FLATTEN
15. FC (units=1,activation='linear')

As more architectures are employed, the full list will be updated here.

3. New Architecture

Using the architecture of MN_2, we can try to examine a similar case to which we applied MN_1. Below is the prediction of model atreus1273, applied to a sample size of 70000 iterations spread equally among 5 turning rates (50000 training, 20000 validation), for density 0.15.

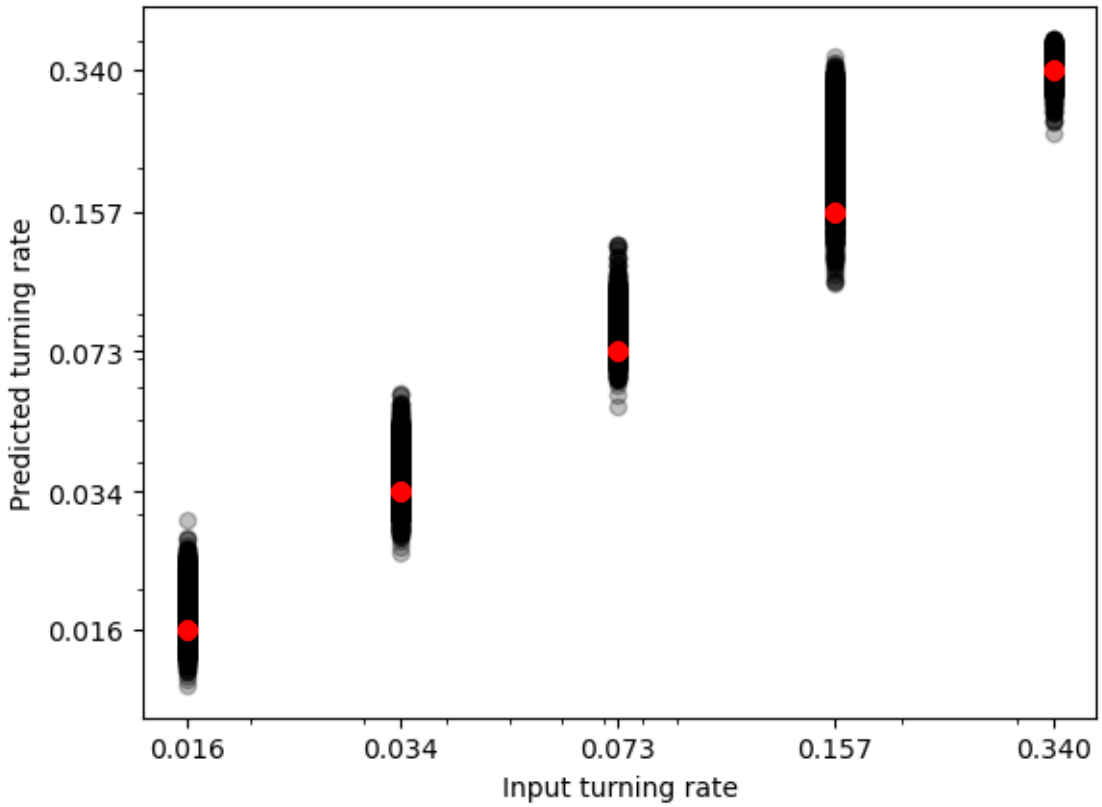


The fitting overall seems to be worse, but the top middle values are almost perfectly fitted. My lab partner has gotten better results with this model, so I will continue exploring it in parallel with MN_1 for now.

4. More Tumbling Rates

There is an obvious extension in increasing the amount of tumbling rates utilised. This may help the CNN pick up on the clustering/dissipation pattern better by providing more scenarios in which it can be examined.

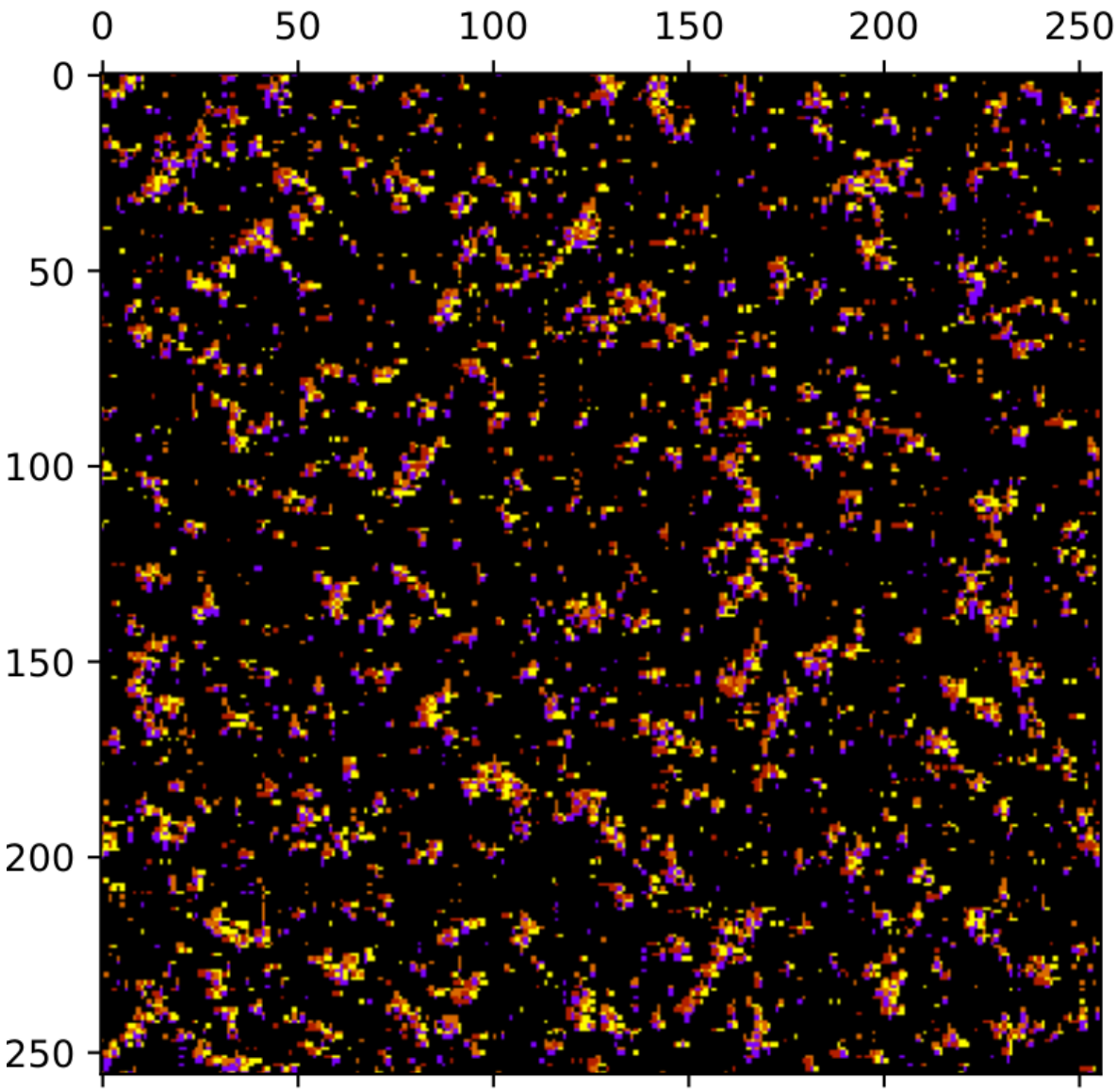
Below is an example reprouction of MN_1 trained on the same values as the last example in Weeks 13-15, model median4431. It has now been standardised to be easliy referred to in our table.



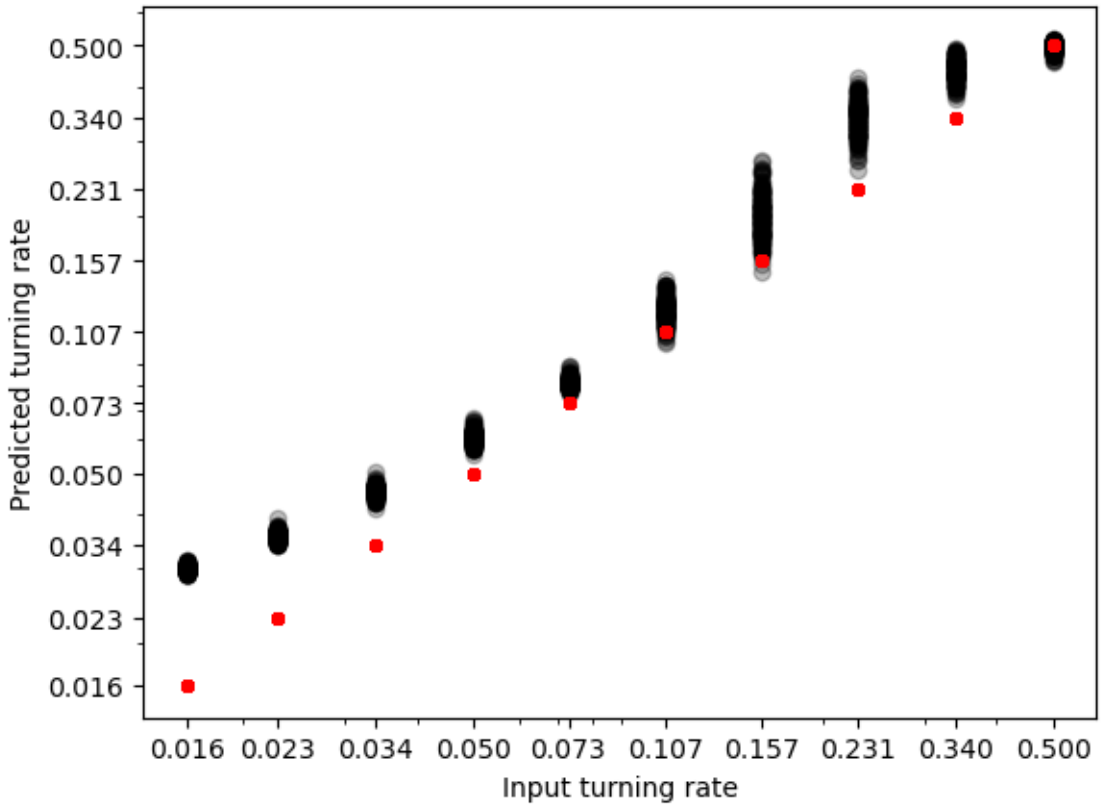
Note that this is using rolled data, so it is bound to be more accurate than the unrolled variants we will consider below.

4. Bigger N Values

Here is a brief attempt to train the CNN on broader N values. This will be investigated more in depth at the end of the project. Model michigan2241 is trained on $N_x = N_y = 256$; an example screenshot can be found below.



Running MN_2 architecture, the training yields the following predictions:



We can see the fitting does not match

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 3)       30

 max_pooling2d (MaxPooling2  (None, 128, 128, 3)       0
 D)

 re_lu (ReLU)                (None, 128, 128, 3)       0

 batch_normalization (Batch  (None, 128, 128, 3)       12
 Normalization)

 conv2d_1 (Conv2D)           (None, 128, 128, 6)       456

 max_pooling2d_1 (MaxPoolin  (None, 64, 64, 6)         0
```

```
g2D)

re_lu_1 (ReLU)              (None, 64, 64, 6)         0

batch_normalization_1 (Bat  (None, 64, 64, 6)         24
chNormalization)

global_average_pooling2d (  (None, 6)                 0
GlobalAveragePooling2D)

dropout (Dropout)           (None, 6)                 0

dense (Dense)               (None, 64)                448

dropout_1 (Dropout)         (None, 64)                0

dense_1 (Dense)             (None, 10)                650

flatten (Flatten)           (None, 10)                0

dense_2 (Dense)             (None, 1)                 11

=================================================================
Total params: 1631 (6.37 KB)
Trainable params: 1613 (6.30 KB)
Non-trainable params: 18 (72.00 Byte)
_________________________________________________________________
```

5. Model Summaries