Summer Internship

A Hunt for Kaons

Supervisor:

Marcin Stolarski

Interns:

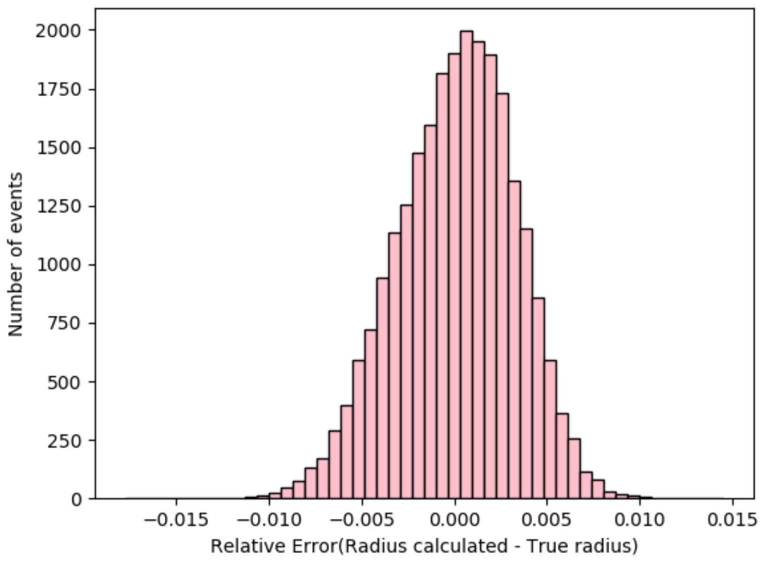
Rui Gonçalves

Filipa Ribeiro

What we want:

This internship's goal was to find the optimal configuration for a neural network that would be able determine the radius of a photon created ring. The samples could contain background noise, a possible shift of the ring and several other negligible rings.

The configurations for the different settings have a loss target of around 1x10^-5. This value was obtained through the mean squared error of the average of all values in each dataset.



Problem: Single Ring, no Background

Best results were obtained using the following network:

```
model.add(Convolution2D(2, (5, 5), activation='elu', input_shape=(36, 36, 1), strides=2))

model.add(AveragePooling2D(pool_size=(2,2)))

model.add(Flatten())

model.add(Dense(4, activation='elu'))
model.add(Dense(2, activation='elu'))
model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Convolutional layers with filters = 4
- Convolutional layer with kernel size of 18x18
- Just one dense layer of size 2 and size 4
- Two dense layers of size 2

Optimizers:
The best results, in this example, were consistently obtained using a combination of two optimizers, Nadam and SGD, in the following configuration:

```
10 epochs using Nadam (learning rate of 0,004) and a Batch Size of 32
5  epochs using SGD(default learning rate) and a Batch Size of 32
10 epochs using Nadam (default learning rate) and a Batch Size of 32
5 epochs using SGD (default learning rate) and a Batch Size of 32
```

Results:
Various tests were done under this configuration, the results are in the following table.

Test	Loss	Validation Loss
1	8.0263e-06	8.0117e-06
2	7.5934e-06	7.6444e-06
3	7.4839e-06	7.6222e-06
4	8.2041e-06	8.3979e-06
5	8.7197e-06	9.5448e-06
6	8.5571e-06	8.8758e-06
7	7.9829e-06	8.2244e-06

Commentary:
One thing of notice with this configuration is how unstable the validation loss is throughout the epochs using the Nadam optimizer, jumping quite a bit between lower and higher values. It still seems to go generally downwards, but using SGD helps to stabilize that value.

Problem: Single Ring with Background

Best results were obtained using the following network:

```
model.add(Convolution2D(10, (5, 5), activation='elu', input_shape=(36, 36, 1)))

model.add(AveragePooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dropout(0.15))
model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(8, activation='elu'))
model.add(Dense(4, activation='elu'))
model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Using one convolutional layer with filters equal to 4, 6 and 16.
- Using only two dense layers but of bigger sizes like 64 and 32.
- Adding no Dropout

Optimizers:
Like in the previous problem, the best results were obtained using a combination of Nadam and SGD, this time using the following configuration:

```
25 epochs using Nadam (learning rate of 0,004) and a Batch Size of 32
5  epochs using SGD(default learning rate) and a Batch Size of 32
```

Results:

Validation Loss: 8.0223e-06

Loss: 5.2644e-06

Commentary:
In this problem we started with a network similar to the one that had worked on the previous one. The results were unsatisfactory with validation loss being as high as 6,2e-05.
What we observed through various experimentations is that a bigger network was needed in order to ignore the background noise and a little dropout aided in avoiding overfitting.
We also verified that, as previously, having a few epochs using SGD helped in stabilizing the validation loss results (this ends up being a common thread in all problems analyzed).

Problem: Single Ring with Background and Shift

Best results were obtained using the following network:

```
model.add(Convolution2D(25, (8, 8), activation='elu', input_shape=(36, 36, 1)))
model.add(AveragePooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dropout(0.35))

model.add(Dense(64, activation='elu'))
model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(8, activation='elu'))
model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Convolutional layer with more(128, 64) and less(10) filters
- Convolutional layer with kernel size of 5
- Convolutional layer with strides(2,2)
- Smaller Dropout
- Same number of dense layers but smaller in size
- More dense layers but with less units each
- Adding a bigger dense layer(128) after the convolutional layer
- Using the shift coordinates to help the network finding better results

Optimizers:
The best results were obtained using a combination of Nadam and SGD, using the following configuration:

```
25 epochs using Nadam (default learning rate) and a Batch Size of 32
5  epochs using SGD(default learning rate) and a Batch Size of 32
```

Results:

Validation Loss: 2.4919e-05

Loss: 1.3385e-05

Commentary:
Surprisingly the network had a hard time figuring out the shift, as in general we ended up with worse results. This training, for the most part, didn’t take into account any of the shifting data only training using the information regarding the radius. The validation loss value still remained fairly unstable through the iterations using Nadam, only to start decreasing at a very slow pace during the epochs that used SGD.
We tried applying strides(2,2) in the convolutional layer but that meant a significant reduction in the parameters used by the network, which in the end, would arrive at worse results.

Problem: Multiple Rings with no Background

Best results were obtained using the following network:

```
model.add(Convolution2D(25, (5, 5), activation='elu', input_shape=(36, 36, 1)))
model.add(AveragePooling2D(pool_size=(2,2)))

model.add(Flatten())
model.add(Dropout(0.50))

model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(8, activation='elu'))
model.add(Dense(8, activation='elu'))
model.add(Dense(4, activation='elu'))
model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Convolutional layer with more(64, 40, 35, 30) and less(10, 20) filters
- Convolutional layer with kernel size of 8
- Convolutional layer with strides(2,2)
- Smaller Dropout (0.25, 0.45)
- Less dense layers with similar parameters
- More dense layers with similar parameters
- Less dense layers but bigger each (128, 64)

Optimizers:
The best results were obtained using a combination of Nadam and SGD, using the following configuration:

```
25 epochs using Nadam (default learning rate) and a Batch Size of 32
5  epochs using SGD(default learning rate) and a Batch Size of 32
```

Results:

Validation Loss: 1.0813e-05

Loss: 8.9620e-06

Commentary:
Using almost the same structure as before the network seemed to have an easier time reaching results closer to to the desired values. Again, like before, strides(2,2) were tested in the convolutional layer, as expected the network was faster but the validation loss was higher.
The validation loss reached low values (2.3 e-05) fairly fast but had difficulty going lower than those. As per usual it was fairly unstable around the 2.5e-05 mark with a few spikes to higher values. The SGD used in the last 5 epochs seemed to help stabilize the value closer to 1.0e-05.

Problem: Multiple Rings with Background

Best results were obtained using the following network:

```
model.add(Convolution2D(25, (5, 5), activation='elu', input_shape=(36, 36, 1),
strides=(2,2)))

model.add(Flatten())
model.add(Dropout(0.50))

model.add(Dense(32, activation='elu'))
model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(4, activation='elu'))
model.add(Dense(4, activation='elu'))

model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Convolutional layer with more(64, 40, 35, 30) and less(10, 20) filters
- Bigger Dropout (0.65)
- Less dense layers with similar parameters
- More dense layers with similar parameters
- One big dense layer(128) right after the convolutional layer.

Optimizers:
The best results were obtained using a combination of Nadam and SGD, this time using the following configuration:

```
30 epochs using Nadam (default learning rate) and a Batch Size of 16
10  epochs using SGD(default learning rate) and a Batch Size of 16
```

Results:

Validation Loss: 1.0105e-05

Loss: 6.2973e-06

Commentary:
This Network used a very similar structure to the one on the previous problem.
There's a few key differences noticed though. The network seemed to get better results when not using any kind of pooling layer, but applying strides(2,2) to the convolutional layer. This not only allowed the network to get closer to the desired values but also to be significantly faster, taking around 20 seconds less per epoch using the same machine.
Also of note the usage of a smaller batch size (16) seemed to improve the overall learning of the network.
The optimizers were used in the same structure as before, only this the network did a few more epochs.

Problem: Multiple Rings with Background and Shift

Best results were obtained using the following network:

```
model.add(Convolution2D(25, (5, 5), activation='elu', input_shape=(36, 36, 1),
strides=(2,2)))

model.add(Flatten())
model.add(Dropout(0.50))

model.add(Dense(128, activation='elu'))
model.add(Dense(64, activation='elu'))
model.add(Dense(32, activation='elu'))
model.add(Dense(16, activation='elu'))
model.add(Dense(4, activation='elu'))

model.add(Dense(1,activation='sigmoid'))
```

What we tried:

- Convolutional layer with more(64, 32) filters
- Convolutional layer with strides (3,3)
- Bigger Dropout (0.60)
- Less and smaller dense layers.
- More dense layers with similar parameters
- One bigger dense layer (256) right after the convolotional layer

Optimizers:
The best results were obtained using a combination of Nadam and SGD, this time using the following configuration:

```
30 epochs using Nadam (default learning rate) and a Batch Size of 16
5 epochs using SGD(default learning rate) and a Batch Size of 16
```

Results:

Tests were done under this configuration using different sample sizes, the results can be found in the following table:

Sample Size	Loss	Validation Loss
25 000	1.7817e-05	4.5955e-05
50 000	2.9662e-05	2.5499e-05

Commentary:

As per before the neural network has a hard time dealing with the shift. Considering the resources available no tests were done without using strides (2,2) on the convolutional layer, although previous results hint that it could result in an excess of parameters for the network and the scope of the problem.
After various tests a ceiling of what could be achieved with the amount of data present in 25 000 samples seemed to be reached. As so, the network was trained for 35 epochs with 50 000 samples and the results obtained show the validation loss slowly descending with no clear signs of overfitting.

General Remarks:

We can conclude that in this type of experiments:

- Neural networks have a difficult time dealing with shifted circles, when only using data regarding the radius.
- One convolutional layer followed by a few dense layers of decreasing sizes seems to be the most effective configuration for this kind of problem.
- Using the Nadam optimizer can be effective to get low values of validation loss in a few epochs, but it’s also fairly unstable, with frequent spikes throughout the iterations. As so it can be useful to do a few epochs utilizing SGD, since it seems to stabilize the values.
- More complex problems, like multiple rings with shift, need a bigger pool of data to get results closer to the desired.