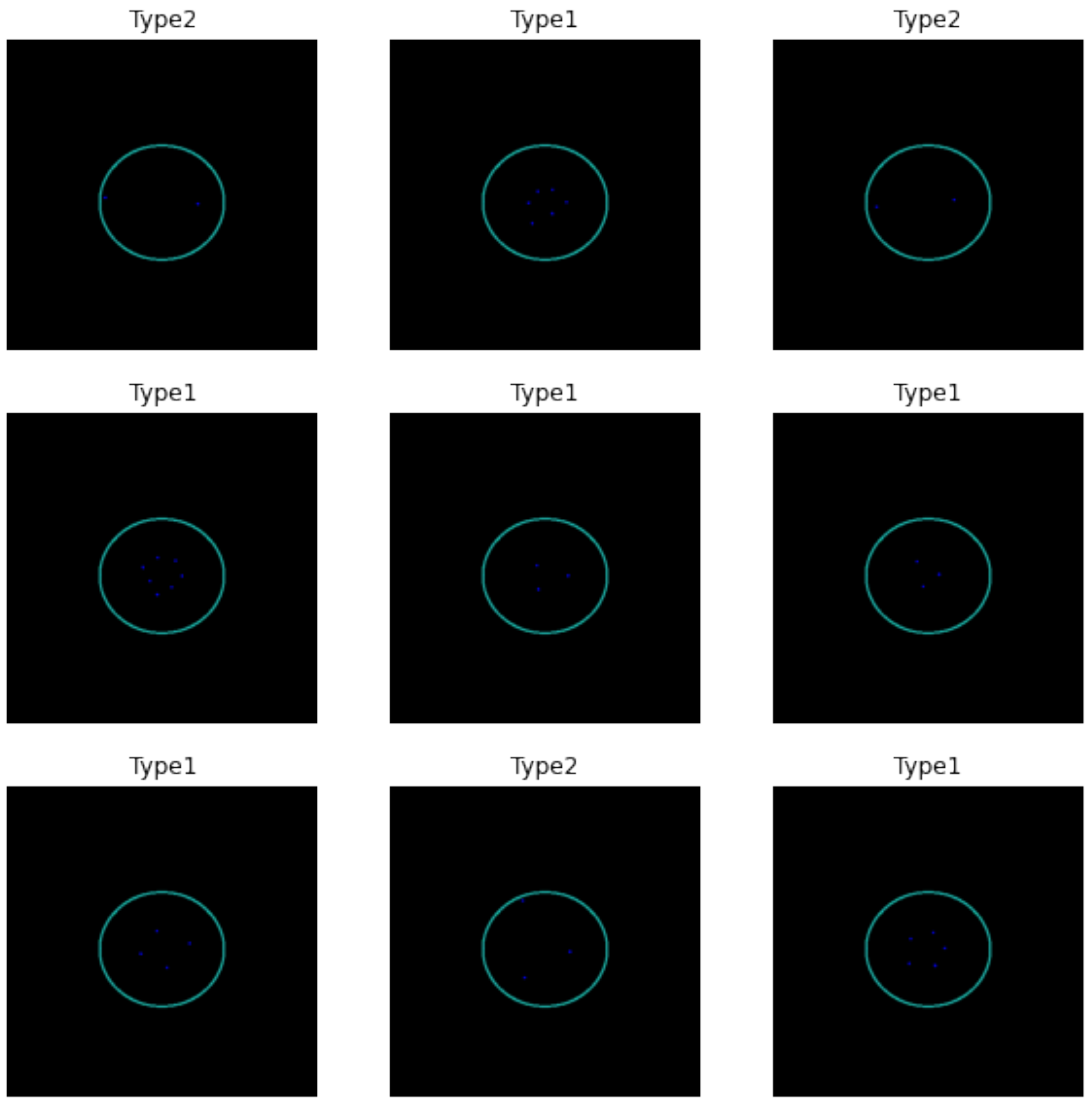
Architecture Design

Image Classification

YASH SARASWAT



Model Choice
CNN from scratch

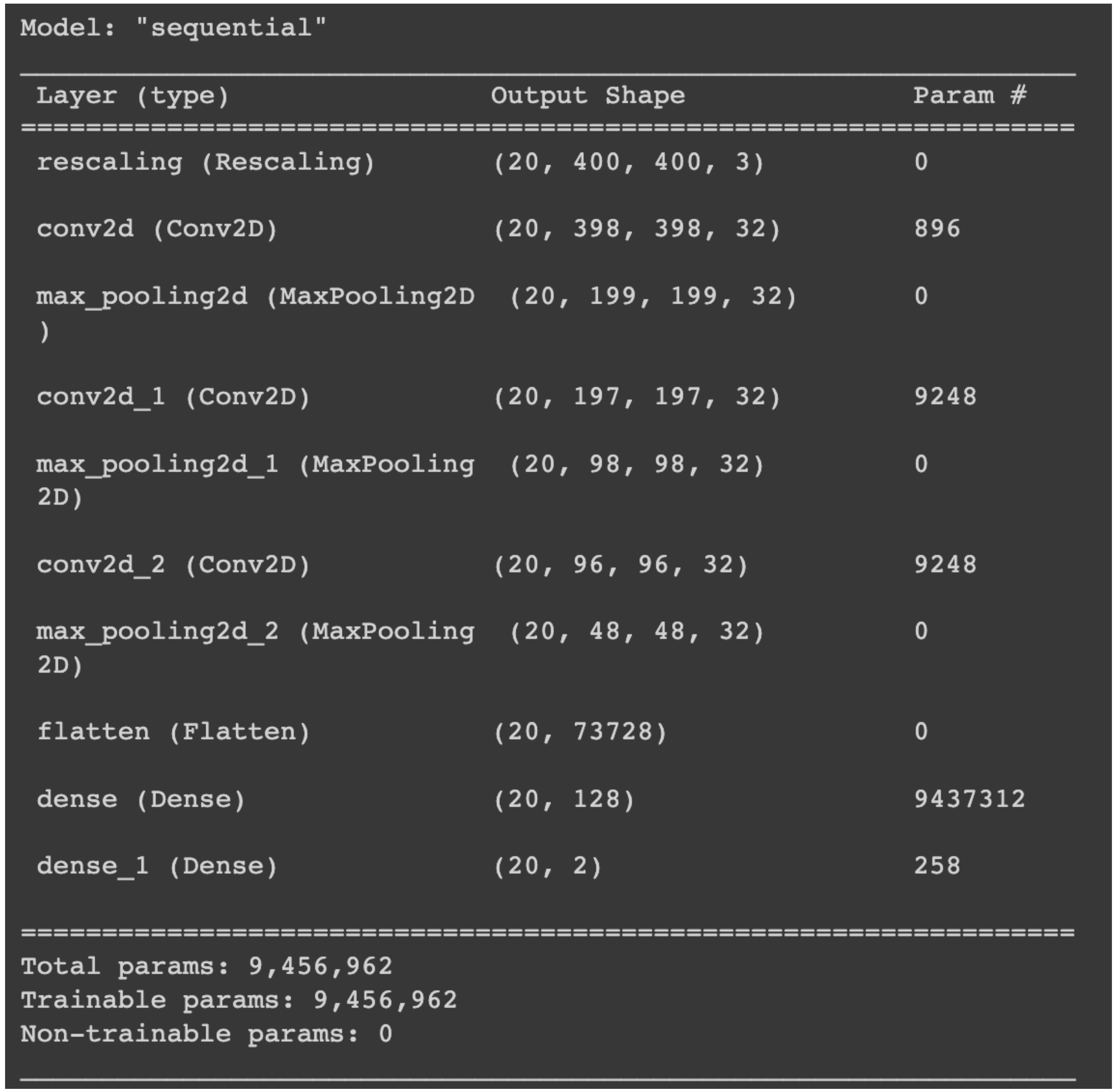
- Given that the problem is one of image classification (binary), going with a CNN architecture is the best way to start.
- I decided to go with writing the CNN architecture from scratch taking inspiration from classification tasks I have done earlier.
- As shown, the images to be classified displayed a relatively simpler image statistic (less number of features at first glance).

Processing and Training Hyperparameters

- **Batch size (20)** is usually chosen as 32 (for smaller datasets), 64, 128, etc. Batch size of 20 was chosen as it was close to the conventional size and it perfectly completed the dataset (n = 2000).
- Choice of **Optimizer (Adam)** and **Loss (Sparse crossentropy)** is again trivial as they are shown to have the best performance for classification tasks and are generally chosen for the first experiment.
- **Epoch size (10)** was chosen after experimenting with higher epochs. It was seen that after epoch ~6-8, the accuracy of the model stagnated. This meant higher epochs contributed nothing but higher compute time.

CNN Architecture

- **Architecture** of **Conv-Pool-Conv-Pool-Conv-Pool (AlexNet type trend of 32-64-128)** was used which is widely shown to succeed for simpler datasets and binary classification. The motivation behind trying out this architecture written from scratch was purely experimental.
  - **Alternates:** Other alternates to this configuration are **VGG-16 (Conv-Conv-Pool type)** or **ResNet (skip connections)**. We usually to move to these architectures if the performance on the simpler networks is not sufficient or if we have a huge number of classes.
- Finally a **Flatten()** layer followed by a **Dense(128 features)** was used. Multiple **Dense(128 + 64)** layers were experimented, with but they were dropped to avoid overfitting the data as the training size is relatively small. The final classification layer was a **Dense(num_classes)** layer with a softmax activation.
- For all **Conv2D()** and **MaxPooling2D()** layers standard kernel sizes (3x3 and 2x2 respectively) were used. Higher kernel sizes used would increase the chances of the dots (the only feature of the image that is used for classification) being undermined in the features as the kernel would pick up more of the dark background. Higher kernel sizes are preferred for shrinking larger image datasets, which was not required here.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling (Rescaling)       (20, 400, 400, 3)         0

 conv2d (Conv2D)             (20, 398, 398, 32)        896

 max_pooling2d (MaxPooling2D  (20, 199, 199, 32)       0
 )

 conv2d_1 (Conv2D)           (20, 197, 197, 32)        9248

 max_pooling2d_1 (MaxPooling  (20, 98, 98, 32)         0
 2D)

 conv2d_2 (Conv2D)           (20, 96, 96, 32)          9248

 max_pooling2d_2 (MaxPooling  (20, 48, 48, 32)         0
 2D)

 flatten (Flatten)           (20, 73728)               0

 dense (Dense)               (20, 128)                 9437312

 dense_1 (Dense)             (20, 2)                   258

=================================================================
Total params: 9,456,962
Trainable params: 9,456,962
Non-trainable params: 0
_________________________________________________________________
```

CNN Architecture (continued)

- **Padding** was not used as the images are well padded and uniform from the even background. **Augmentation** was not used for the same reason: the model would just deteriorate from the dataset if images are augmented, as all the images are the same and uniform except for dot placement.
- Model summary is shown on the left.

Footnote.

- One of the obstacles faced while experimenting was the lack of test true labels. The final test accuracy is not known for any model we have tried hindering the final evaluation and inferences that can be drawn from the various models tested.
- Given that the dataset was just a bunch of points that are to be classified based on their spatial scattering, this problem can also be treated as a spatial classification problem where the coordinates can be fed into a Neural Net and a classification boundary can be drawn. This will reduce the compute expenditure we spent on the convolutions, but will require image decomposition and preprocessing.