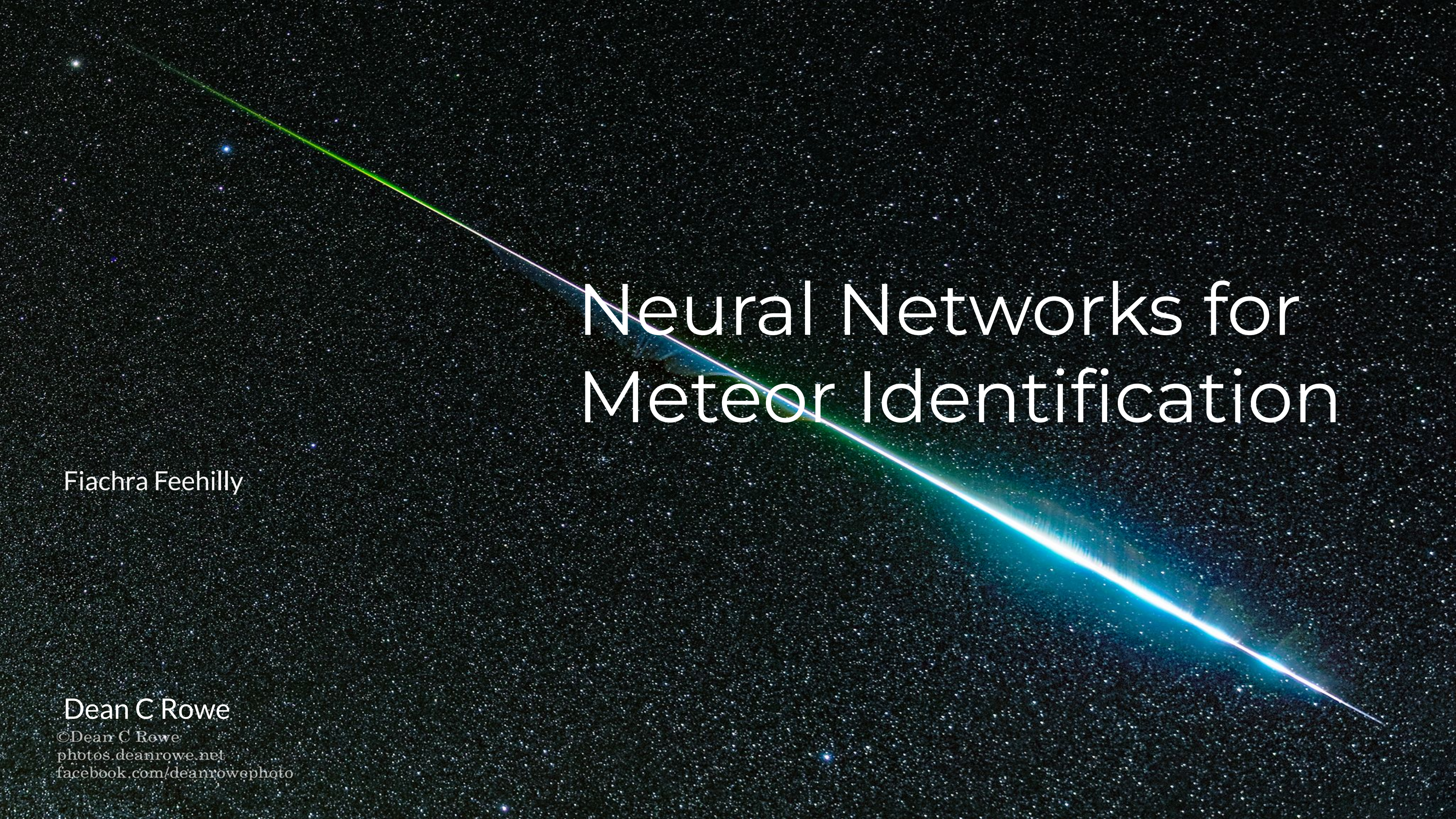


Neural Networks for Meteor Identification

Fiachra Feehilly

Dean C Rowe

Meteors

What are they?

Large rocky objects that burn up in the Earth’s atmosphere

Come from meteoroids, asteroids and comets that come near Earth

Reasons for studying meteors:

- Remained largely unchanged since the start of the solar system
- Give insight how the solar system has changed
- Can impact with spacecraft and satellites

Global Meteor Network(GMN)

Network of camera stations around the world

Record the night sky and picks out any meteor detections

Meteor detections are triangulated using footage from multiple camera stations

Meteor can then be tracked in space or potentially found if it lands

https://globalmeteornetwork.org/

Purpose of my Project

Current GMN detection algorithm has a number of false detections from:

- Planes
- Insects
- Clouds
- Etc.

Current method to remove false positives is human decision on each detection

Deep Learning/Machine Learning algorithm would reduce these false detections and human input required

Previous work

Peter S. Gural

Shallow learning neural network for CAMS network

Had 100,000 Confirmed meteors and 100,000 false alarms

>99% accuracy

Applied on GMN data

Achieved ~97% accuracy

Efforts also made by Esther Gavin

Dataset of total 3210 images

Achieved ~86% accuracy

For comparison

Dataset I used:
Total 84,675 images
58,690 confirmed meteors
25,985 false alarms

Deep Learning

Essentially hoping that by using many different operations you can find some useful combination

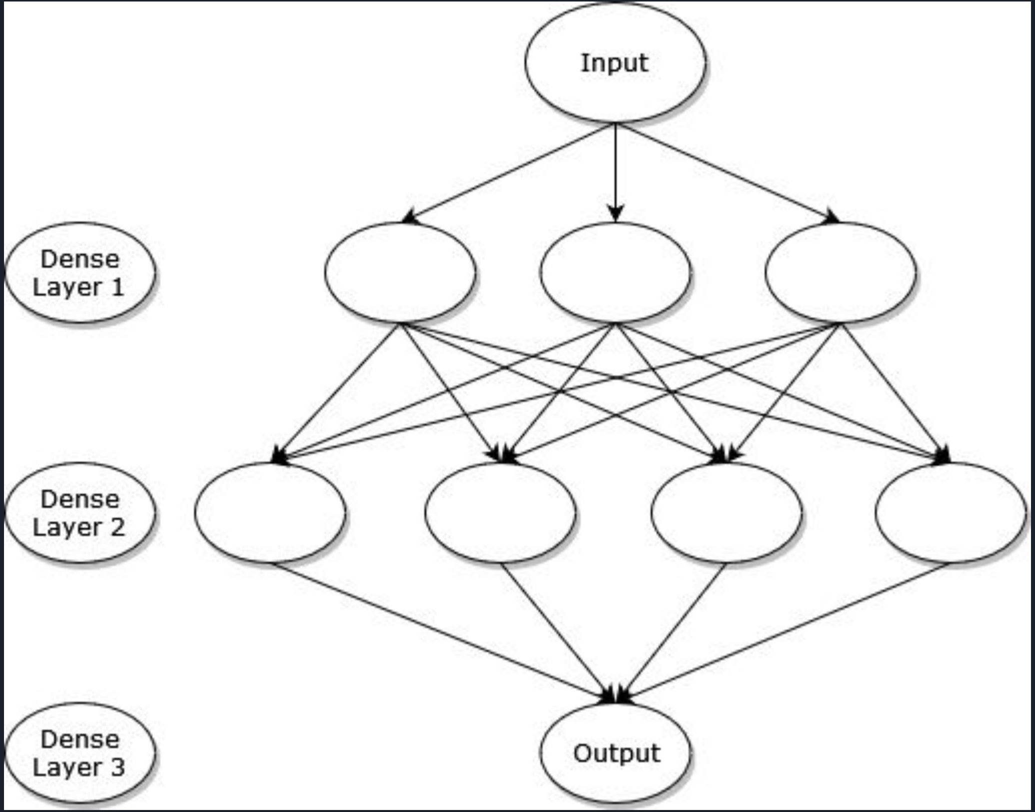
Interconnected neurons, organised in layers

Each neuron has an operation similar on form to

relu(dot(W, input) + b)

Weights randomly initialised

Weights adjusted to reduce loss function/increase accuracy



Training

Neural Network is fed labelled inputs

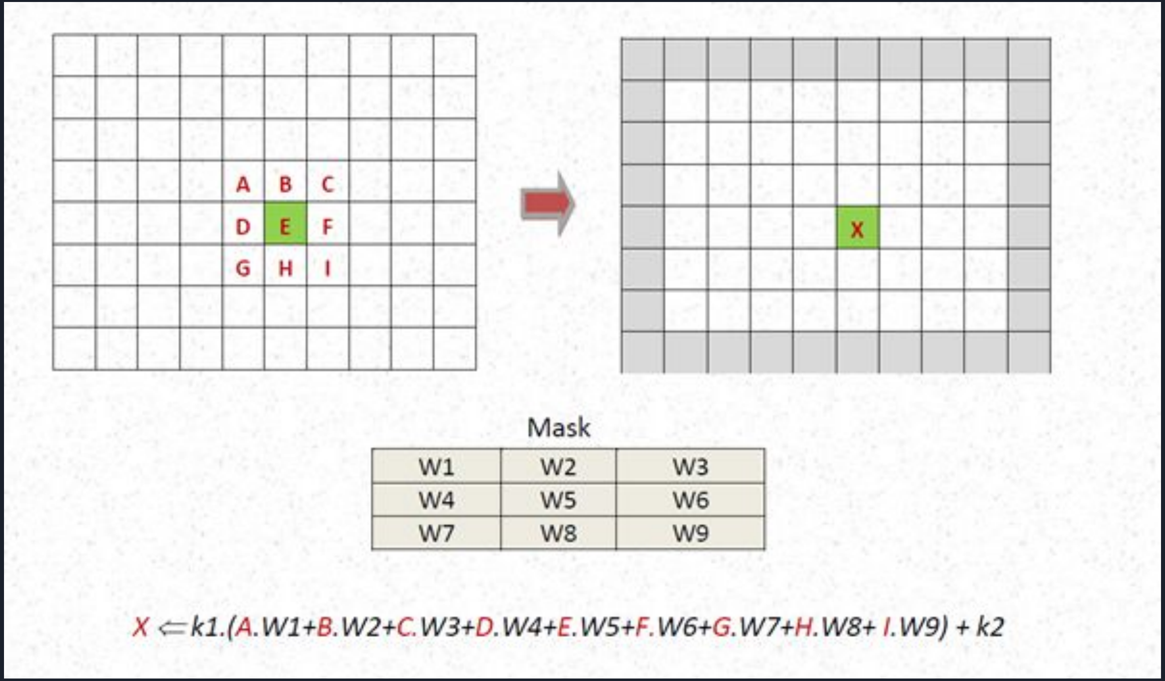
Neural Network uses a loss function to try and improve its performance and correctly identify inputs

Convolution Layer & Maxpool Layer

Dense Layers only apply to 1D tensors

2D convolution layers used to analyse images/spatial data

Maxpool Layer downsamples images so convolution layer can look at “larger” area



$X \Leftarrow k1.(A.W1+B.W2+C.W3+D.W4+E.W5+F.W6+G.W7+H.W8+I.W9) + k2$

Peter S Gural CNN(Convolutional Neural Network) Structure

Flatten Layer needed as Dense layers only work on 1D tensors

Sigmoid activation function in last layer gives probability value between 0 and 1
1 is a meteor
0 is a non meteor

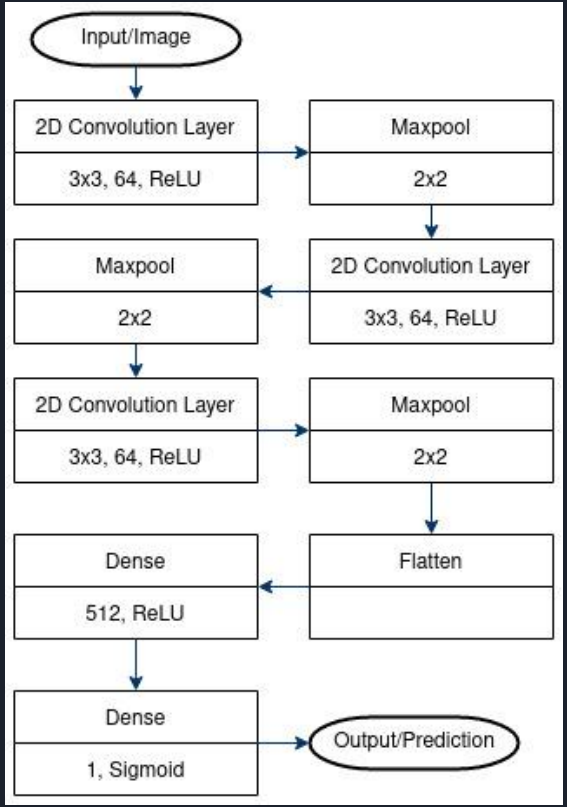


Image inputs

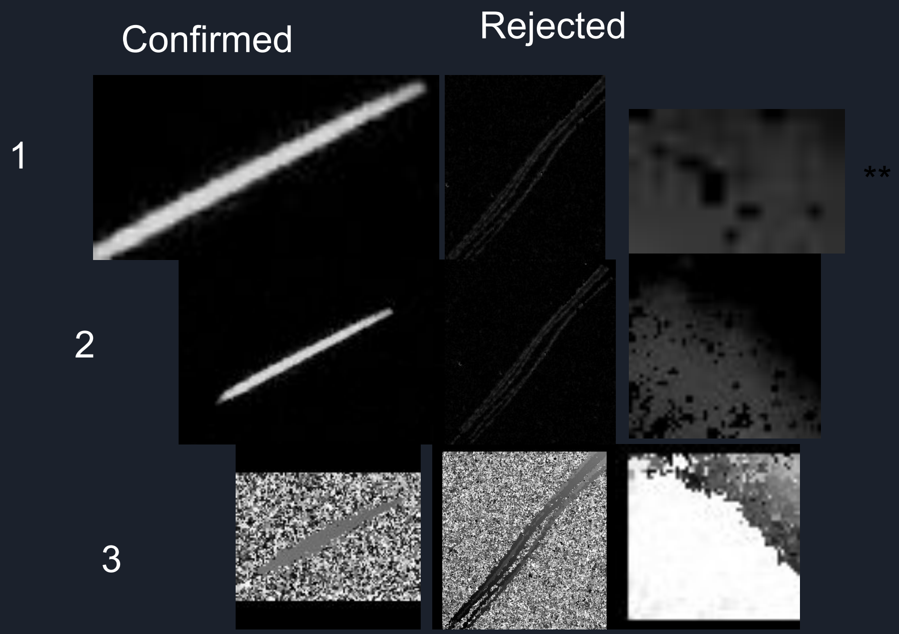
Max pixel image minus Average Pixel image timesliced to just movement frames:

1. Exact dimensions
2. +20 pixels on each side

Max Frame image(time information)

3. +20 pixels each side

*images on right have been stretched/shrunk to be easier to see, especially the ones marked **



Results

	Input Type	Border Size	Accuracy
Large Network 1	Movement	+20	0.9775
Large Network 2	Maxframe	+20	0.9395
Large 2 input Network	Movement + Maxframe	+20	0.9695
Small Network 1	Movement	+20	0.9820
Small Network 2	Movement	+5	0.5300

Small networks are ~950x smaller than Large network

Discussion

Possible cause of poor performance with Maxframe input:

- blackfill added around edges of image, doesn’t match with noise in the rest of the image

Small input images:

- Accuray doesn’t increase as network is trained
- Some images could be small and only contain one pixel value
- Preprocessing step of dividing by standard deviation is causing a divide by 0
- Neural Network operation somewhere causes a divide by 0

Could experiment with even smaller neural networks

Could try analyse operations in a very small neural network to investigate what features the neural network is looking for

Conclusions

Able to reconfirm Peter Gural findings, including about time information not improving accuracy

Managed to reduce size of neural network considerably while still maintaining the same level of accuracy

Discovered that the input image size had a major impact on the performance of the neural network

Maybe Don’t Include

Acknowledgements

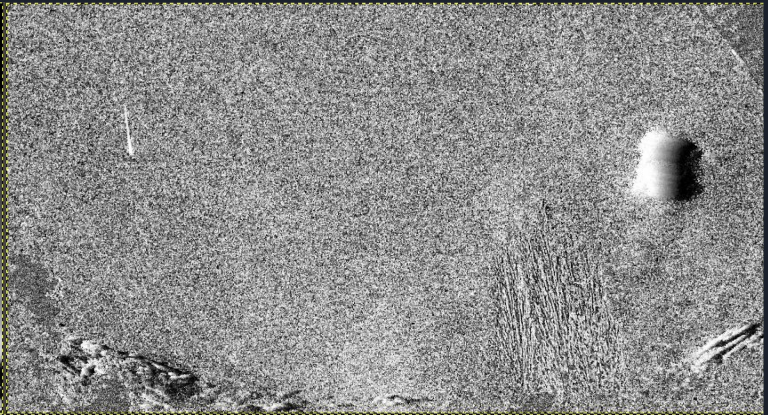
Denis Vida - creator of the Global Meteor Network (GMN) provided the data

Members of GMN - manually classified all the data I used

Jonathan Mackey - Project Supervisor

Max Pixel

Max Frame



Average Pixel

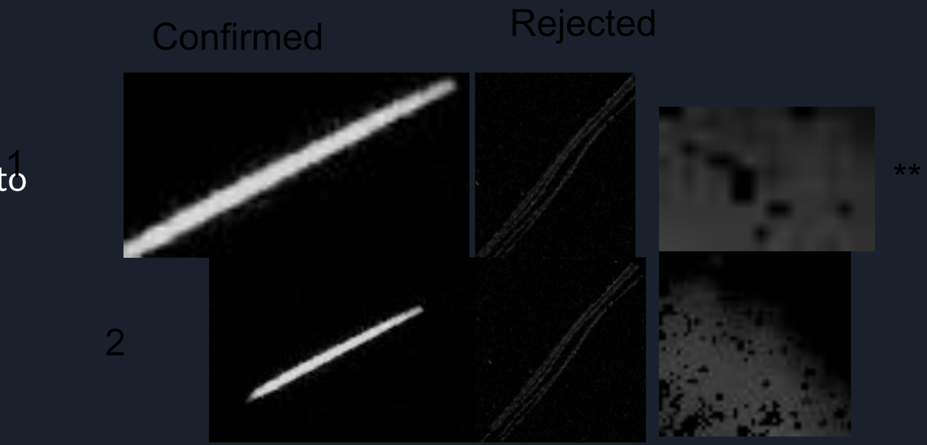
Standard Deviation

Image inputs

Max pixel image minus Average Pixel image timesliced to just detection frames:

1. Exact dimensions
2. +20 pixels on each side

*images on right have been stretched/shrunk to be easier to see, especially the ones marked **



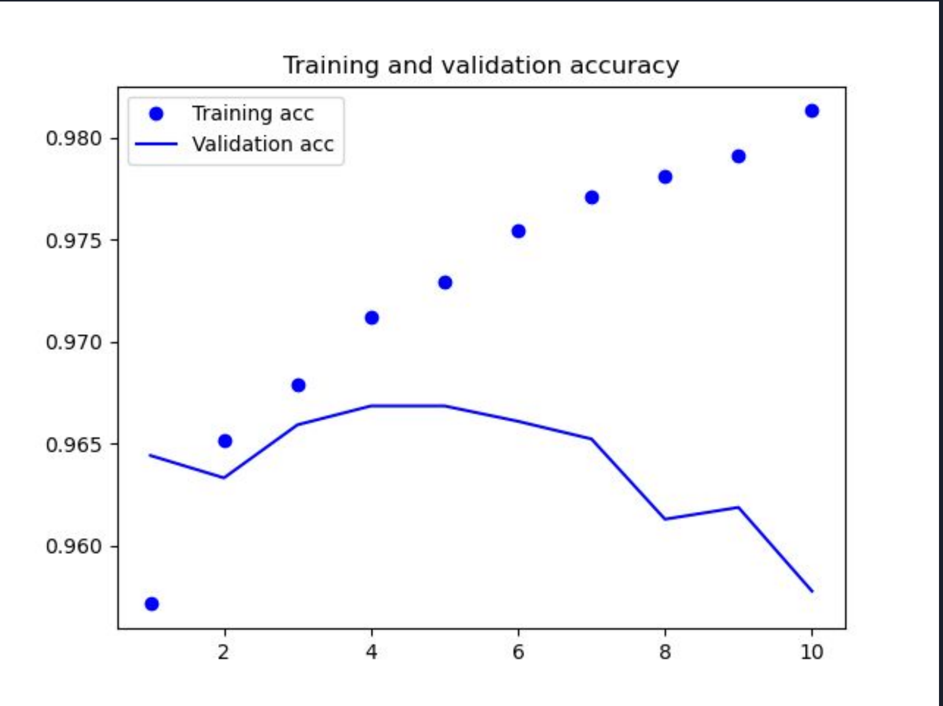
Performance border vs no border

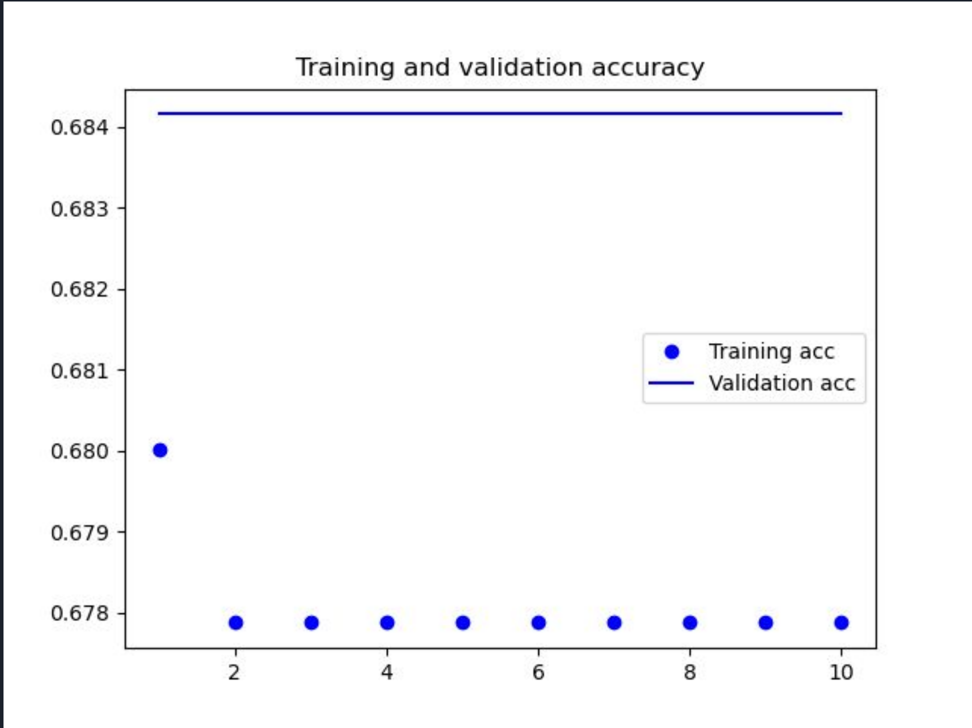
+20 pixels on each side

Made square

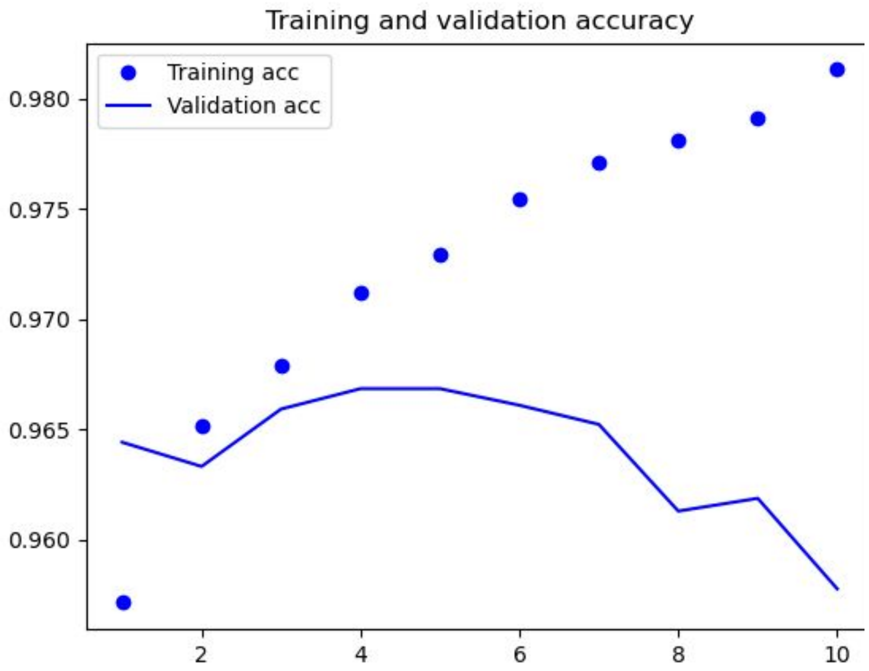
No +20 pixels all on all sides

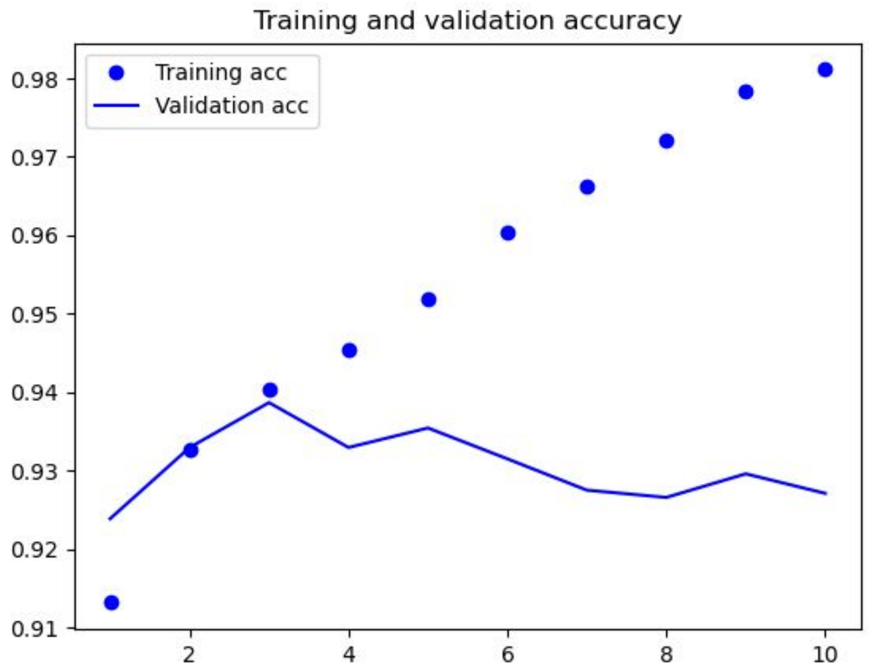
Similar performance if made square or not





Performance



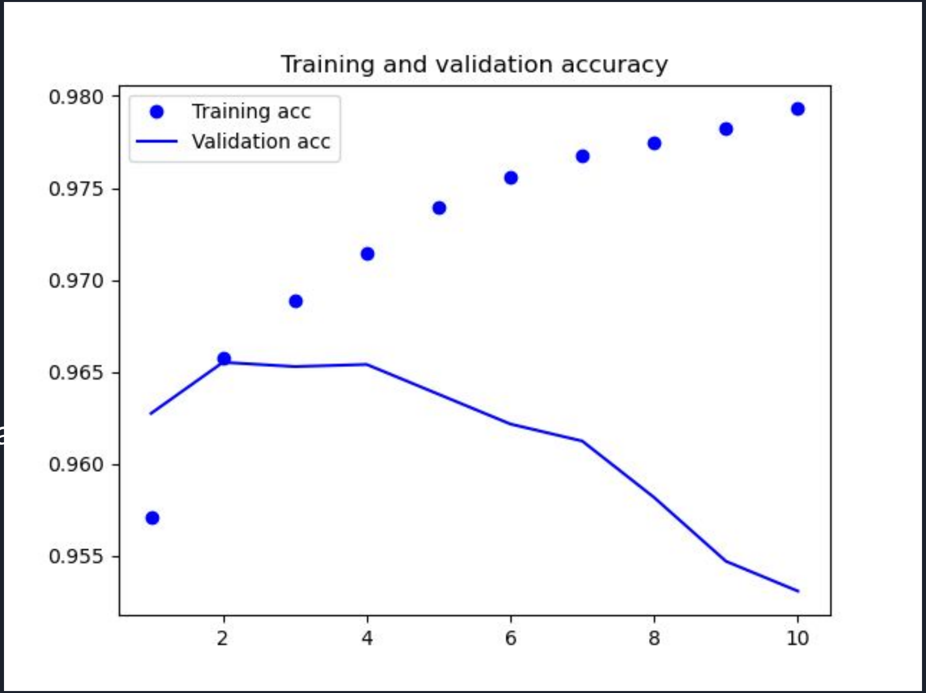


Small Network

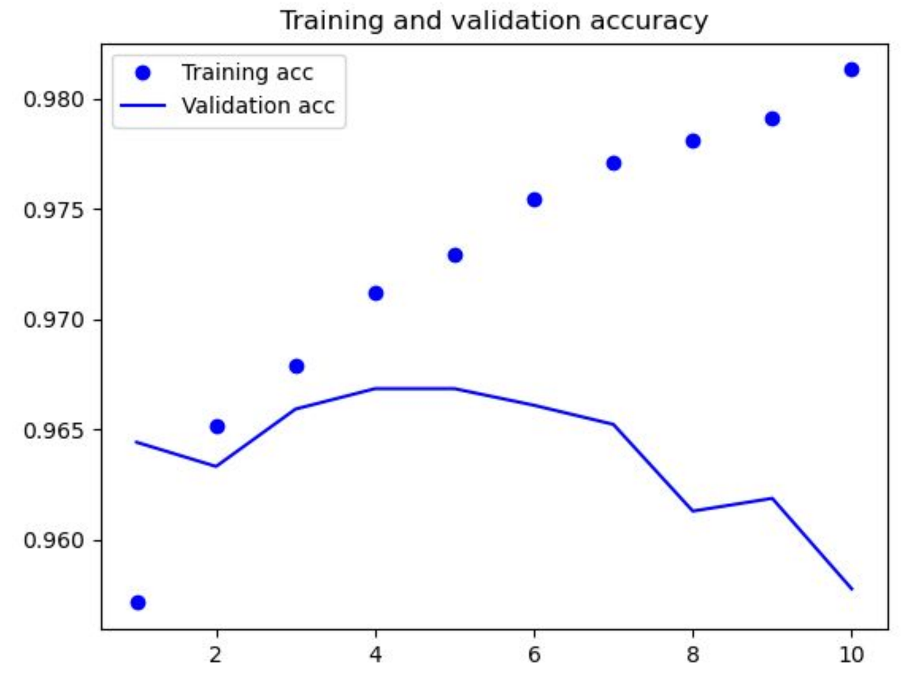
- 1 3x3, 32, Convolution Layer
- 1 3x3, Max Pooling Layer
- 1 64 Dense Layer
- 1 sigmoid Layer

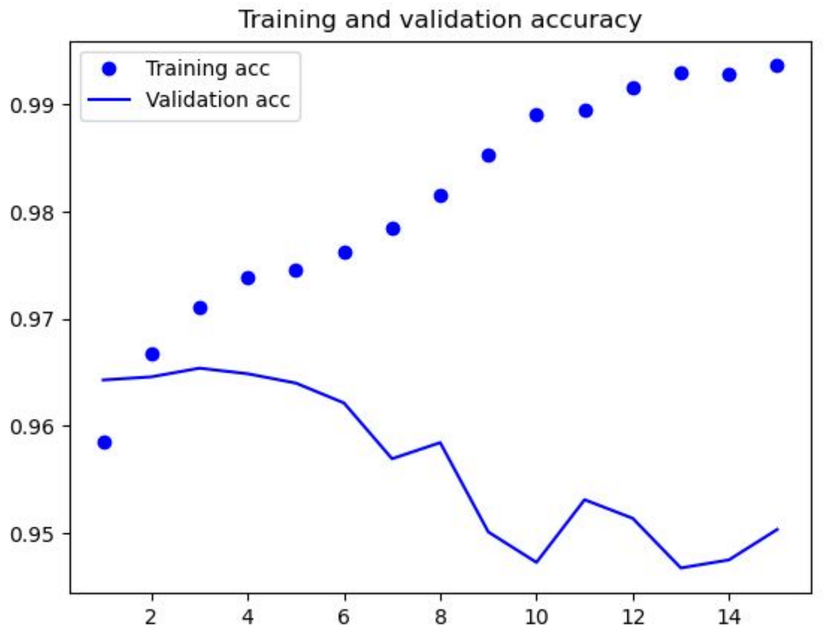
~Half the number the number of variables a
the Peter Gural structure

3,613,121 vs 6,498,049



Performance





Rescaling, Centering, Standardising

- Images were scaled to have values between 0 and 1 by dividing by the max possible intensity value in the image (255)
- Data was then recenter about 0 by subtracting the mean of each image from itself
- Each image was then standardised to give the same distribution by dividing each image by its standard deviation
    - This caused values to then be outside the range -1 and 1 as the images have a bimodal instead of gaussian distribution and so standard deviation is quite small
- Each image was then rescaled again to be between -1 and 1

No significant performance difference between each stage, at most ~2%

Would need to do more testing to see if there is consistent differences

Smaller performance benefits of become more significant as overall accuracy increases and there may be more consistent improvements seen once duplicate images are removed

My Data

Num: COnfirmed Meteors

Num: Rejected Meteors

Same structure 90% accuracy

1 Convolution layer

Different size convolution layers

Different input images (different borders)

Data augmentation

Multi input networks

Effect of rescaling, centering and normalising

Input images

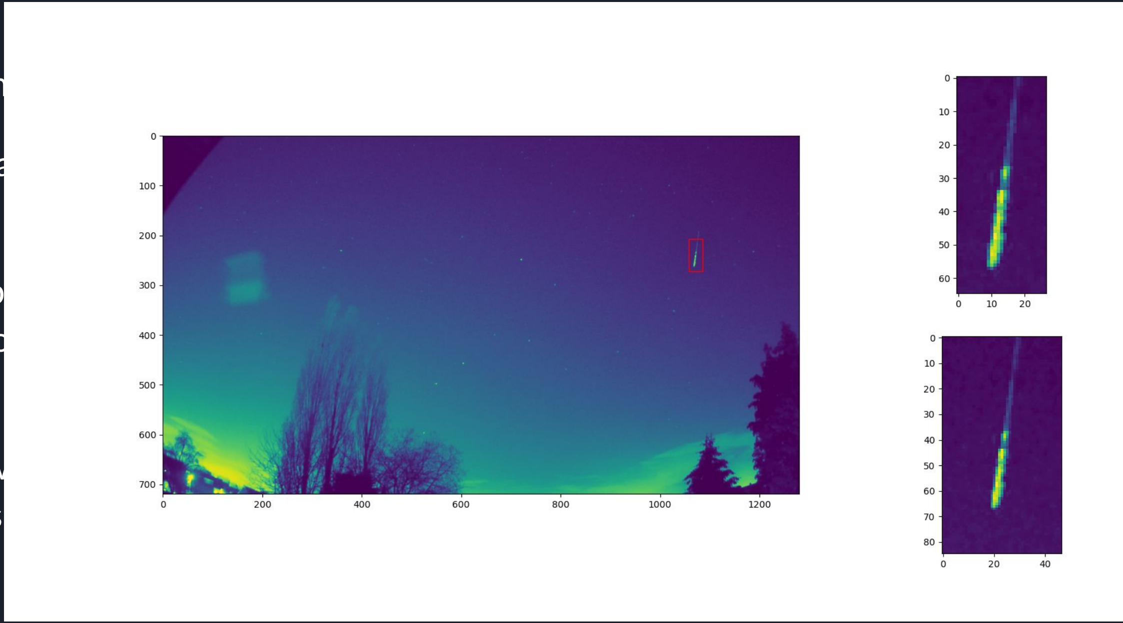
Made dataset of png images

Each png was the Maxpixel frame - Avgpixel fran

Each png included +20 pixels on each side outwa from where the detection in the FTP stops

If detection was near an edge, additional black p value intensity) were added to keep the detectio centered

Each image was also made square by adding row columns of black pixels equally to the required s



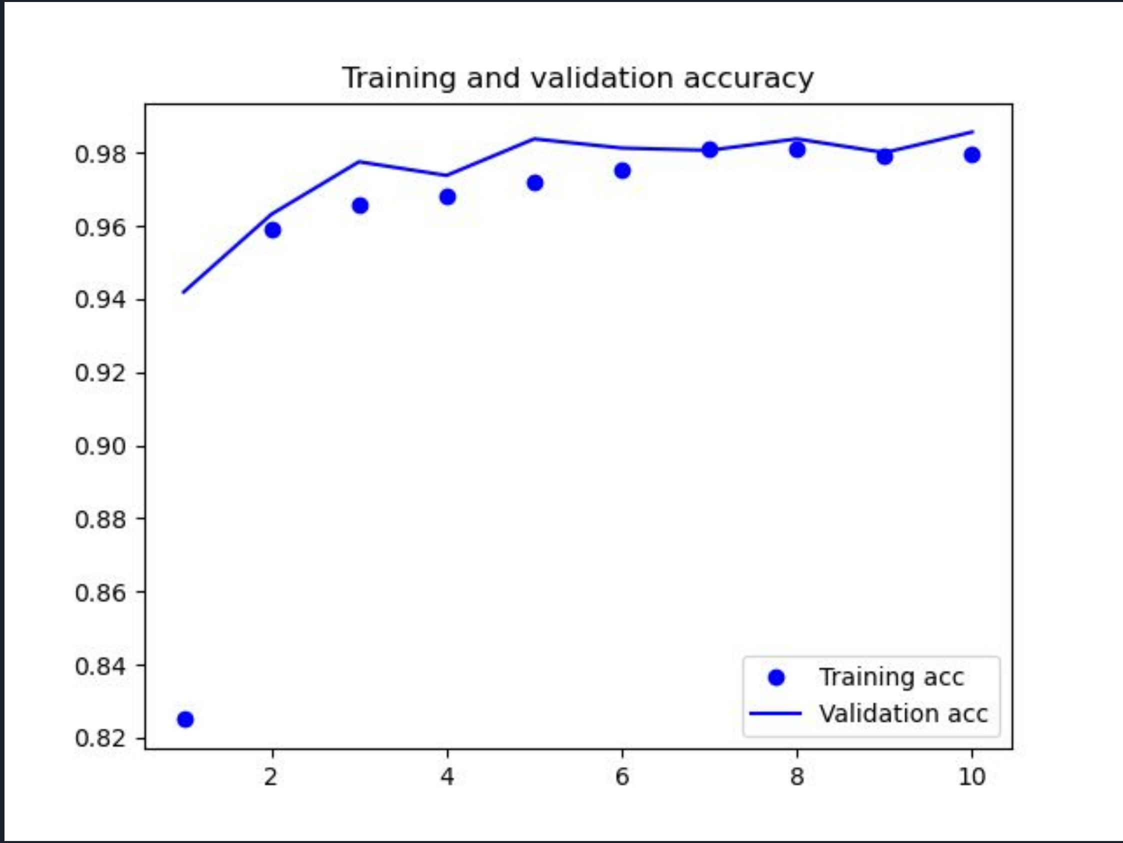
Performance on smaller Dataset

Had:

~56,000 Confirmed detections

~16,000 Rejected detections

Heavy bias towards Confirmed detection



Performance on augmented smaller Dataset

Had:

~57,000 Confirmed detections

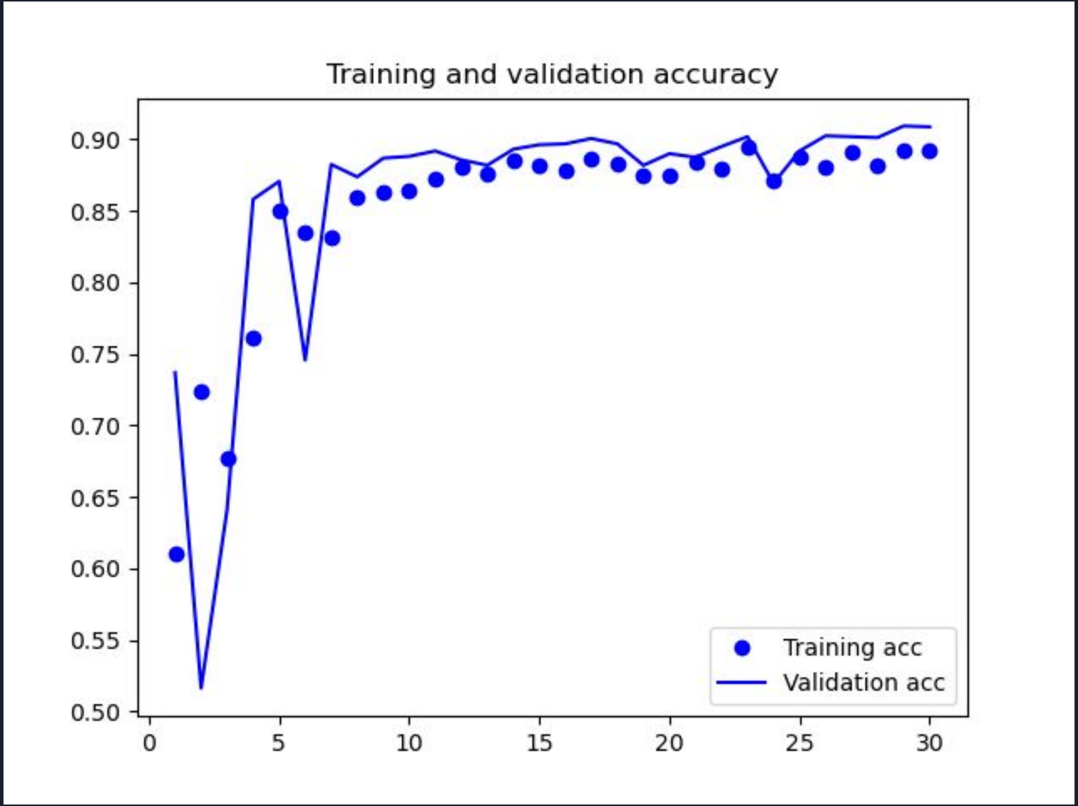
~33,000 Rejected detections

Augmented through horizontal and vertical flips to give

~114,000 Confirmed Detections

~132,000 Rejected Files

Realised later that it contained ~6415 duplicate files that were labelled both Confirmed and Rejected

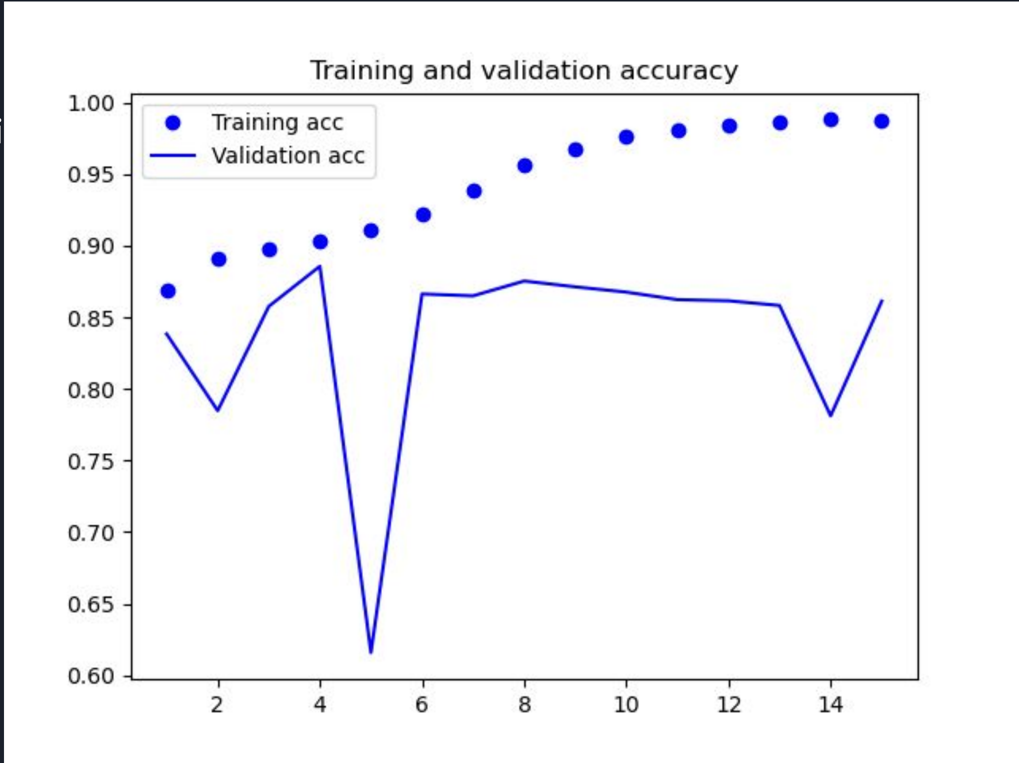


Performance maxframe

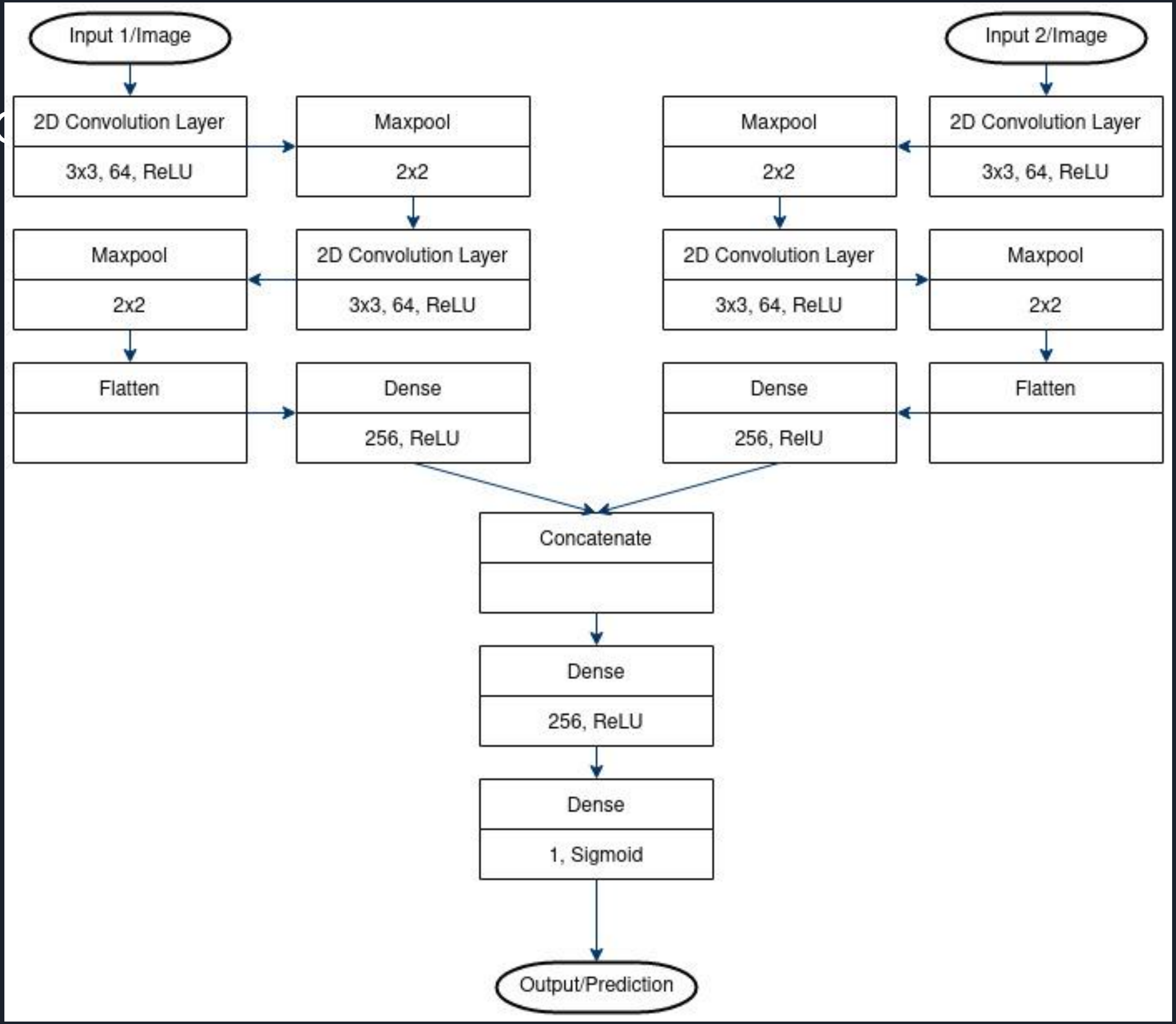
+20 pixels each side

Similar performance regardless if time sli

Images had black around edges to make square/center detection in image



Combo


Input 1/Image
Input 2/Image
2D Convolution Layer
3x3, 64, ReLU
Maxpool
2x2
Maxpool
2x2
2D Convolution Layer
3x3, 64, ReLU
Maxpool
2x2
2D Convolution Layer
3x3, 64, ReLU
2D Convolution Layer
3x3, 64, ReLU
Maxpool
2x2
Flatten
Dense
256, ReLU
Dense
256, RelU
Flatten
Concatenate
Dense
256, ReLU
Dense
1, Sigmoid
Output/Prediction

Training vs Validation

Convolution/Layer Explanations

Suggestions

Different amount of border around detection

Mirroring parts of image if on the edge of frame

Using a different form of fill around the border of maxframe images(noise like- same mean and standard deviation as original image, solid fill of mean/mode, extrapolate border and fill outwards according to pixel on edge

Any way to analyse how full the cropped image is?

Perhaps analysis on mean, mode, median or a combination

Analysis on shape of image? More square means cloud? Rectangular means meteor?

Perhaps a different normalisation function (instead of just dividing by stdev as images are not Gaussian distributed)

Different compile step, loss function, learning rate etc