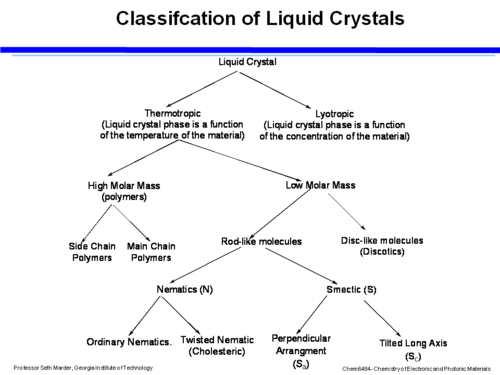
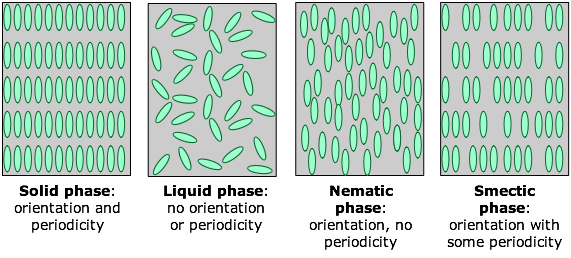
**Can liquid crystal phases be identified via machine learning?**

12/10/2020

The aim of this project is to test the usefulness of modern machine learning, in particular deep learning, algorithms in automating the identification of liquid crystal phases from their textures. A diagram of the different types and phases of liquid crystals is given below.





Liquid crystal texture image data is captured experimentally using polarized optical microscopy (POM). In the isotropic liquid phase with no molecular alignment (no director), light passes through without any change to its polarization axis. As the sample cools, it transitions into the liquid crystal phase at some critical temperature in which the molecules align along a director, leading to birefringent polarization of light passing through the sample. Therefore, when the sample is placed between crossed polarizers, incident light will only pass through when it is in a liquid crystal phase.

Since the data is in image format, a convolution neural network (CNN) will most likely be used in all cases of classification or regression in this project. Some potential applications:

* Phase classification of isotropic/nematic/cholesteric/smectic liquid crystals
* Cholesteric liquid crystal pitch regression
* Nematic liquid crystal temperature regression

**Data sources**

**Experimental**

Vance Williams

<https://www.instagram.com/vance.williams/>

<https://www.youtube.com/channel/UCB8qnCxJbdsuXpQ5RbLNy3Q>

<http://www.sfu.ca/chemistry/people/profiles/vancew/>

Media VR Lab YouTube channel

<https://www.youtube.com/channel/UCqb11FvjkkpD0V5h6p9c2Cg>

Nematic texture video

<https://www.youtube.com/watch?v=c4FuNSUHAPU>

**Simulations**

Nematic textures could potentially be simulated via Monte Carlo methods.

16/10/2020

**First model, Sequential 4 Phases**

**Collecting and processing the data**

We will first attempt to classify images from the following general phases, with an example image provided for each:

* Isotropic (random dark noise due to no birefringence)

A picture containing dark, front, standing, water

Description automatically generated

* Nematic

A picture containing building

Description automatically generated

* Smectic

A picture containing clothing, fabric, rug

Description automatically generated

* Cholesteric

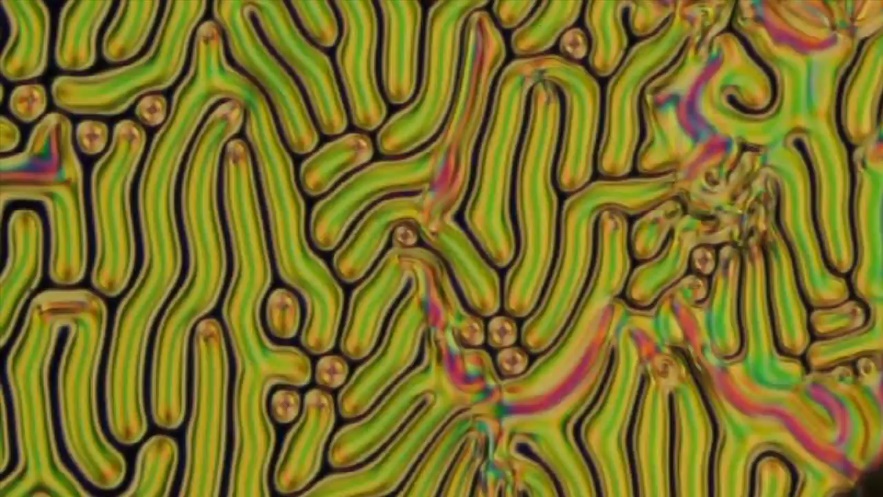
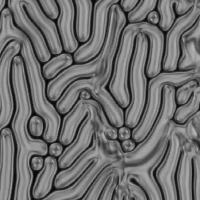
A picture containing honeycomb

Description automatically generated

We will use data from POM videos from Vance William’s YouTube and Instagram as well as some nematic videos from supervisor Ingo Dierking. Frames are extracted from the videos using the VLC Player software’s scene filter function. The isotropic phase is almost completely dark, so we generate random dark noise images for this class. Using this method, we obtained a total of 1742 texture images, not including isotropic images, and generated 500 isotropic images. We have written a script to process the images that does as follows:

1. Crops any excess pixels from the image in order to make it square, with the side length equal to maximum of width/height. This is to keep the proportions correct when resizing. Pixels are removed equally from either side.
2. Resizes the image to a given square dimension, *d*.
3. Optionally converts the image to grayscale.

Here is an example transformation of a cholesteric texture from 640x360 RGB to 200x200 grayscale:



18/10/2020

**Implementation of the models**

We chose an input image size of 200x200 and split the overall image data into a 7:2:1 ratio of training to validation to test sets. We also chose to convert to grayscale because colour is not important in determining LC phases, and it will reduce model size and training time. The pixel data is rescaled from 8 bit unsigned integer (0-255) to float range 0-1. The finalised distribution of data is as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Isotropic | Nematic | Cholesteric | Smectic | Totals |
| Training | 350 | 475 | 372 | 375 | 1572 |
| Validation | 100 | 135 | 106 | 106 | 447 |
| Test | 50 | 67 | 53 | 53 | 223 |
| Totals | 500 | 677 | 531 | 534 | 2242 |

For the sequential models V1 and V2 we use the following settings.

Training specifications:

* Batch size: 32
* Loss function: categorical cross entropy
* Optimiser: Adam, initial learning rate = 0.001
* All training done using TensorFlow Keras with NVIDIA CUDA on an NVIDIA RTX 2060 GPU.

Keras ImageDataGenerator augmentations to improve regularisation:

* Vertical flip
* Horizontal flip
* 30 rotation range
* 0.1 height shift range
* 0.1 width shift range
* 0.2 zoom range

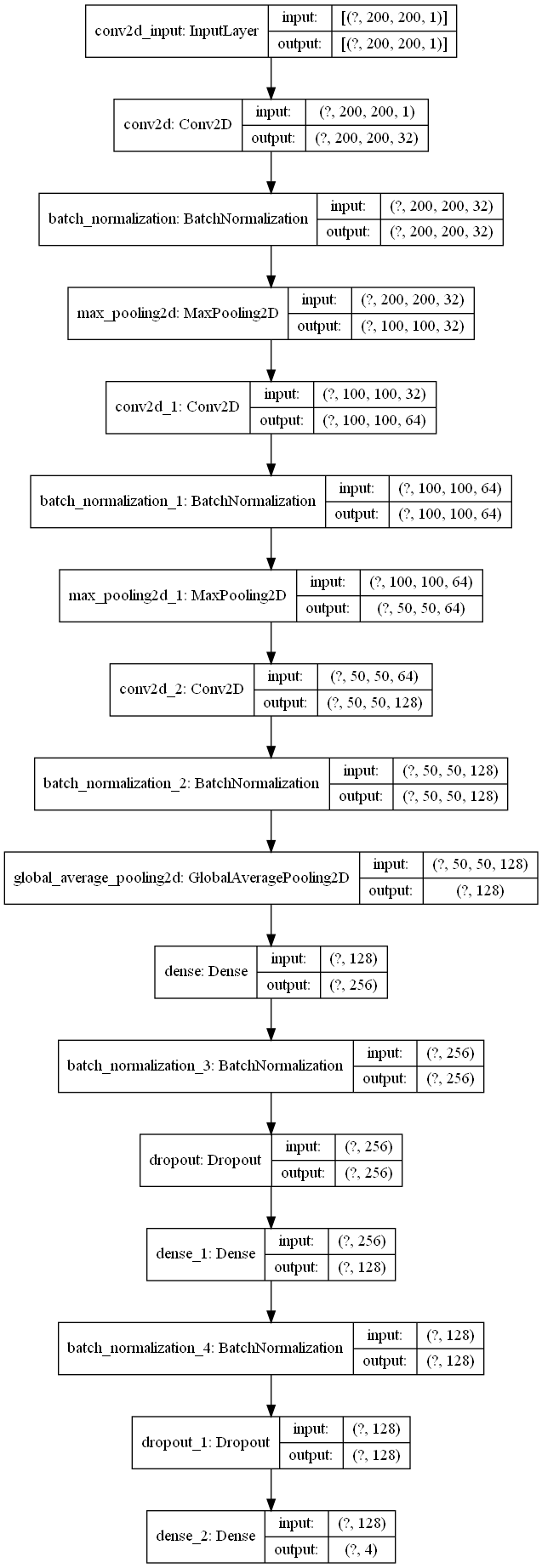
We apply early stopping based on validation set loss with patience of 30 epochs for improved regularisation.

We also reduce the learning rate during training if the validation loss does not improve within 10 epochs, with a reduction factor of 0.5 and minimum learning rate of 1e-5.

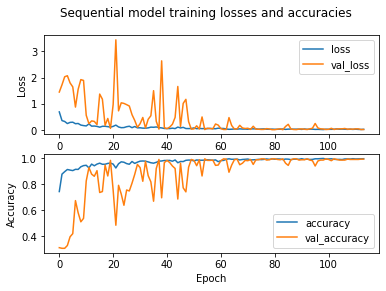
19/10/2020

**Sequential V1**

The first model, Sequential V1, has 3 convolution layers all with ReLU activations and batch normalisation, with max pooling used after the first 2 layers and global average pooling for the last. Each convolutional layer has a stride of 1, kernel size 3x3 and “same” padding. This is followed with 2 dense layers with ReLU activations and dropout for regularisation, and a final 4 unit dense layer for the output. The entire architecture with channel and unit numbers is displayed below.

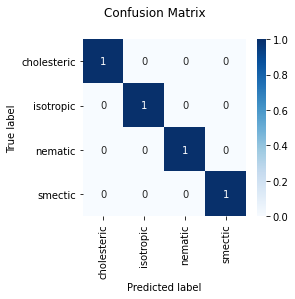


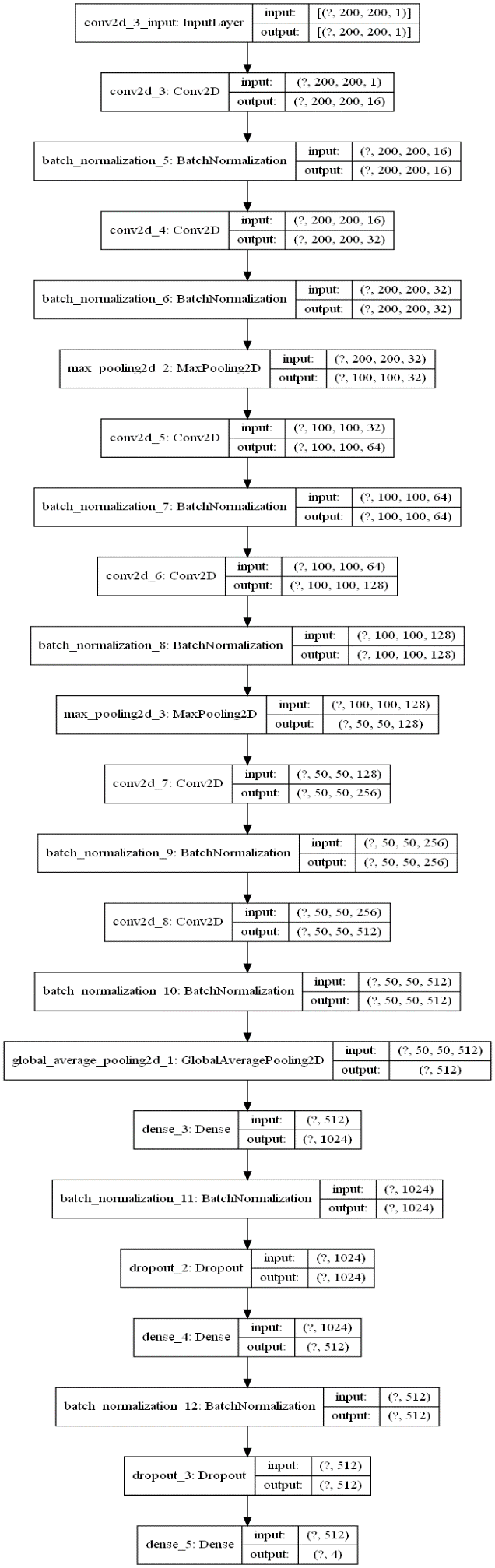
**Sequential V1**

After completion of the training cycle, the final validation set accuracy was 99.04%, and test set accuracy was 100.00%. This clearly demonstrates that CNNs will be extremely viable for the LC phase classification task, as this was a relatively small model. The graphs of loss and accuracy for both training and validation sets over the entire training period are displayed below.

The validation accuracy fluctuates greatly at the beginning before converging to near 100%.

The confusion matrix for the test set predictions is displayed below.





**Sequential V2**

The model has 100% accuracy on the test set leading to a fully diagonal confusion matrix.

20/10/2020

**Sequential V2**

The second sequential model is similar to V1, but it has 3 extra convolutional layers and more filters and dense units, for a much higher total model capacity. The full architecture is displayed to the left.

The model achieved a higher validation set accuracy of 99.76%, and again a test set accuracy of 100.00%. The accuracies for both models are potentially extremely high because image samples from the same videos were included in the training, validation and test sets, which could be a form of data leakage as images from the same video can be very similar to each other. This could mean the models do not actually generalise well. Next, we will try training on a dataset in which training images are from completely different videos to the validation and test set ones, in order to avoid any form of data leakage.

The V2 model loss and accuracy plot is displayed below.

Chart, histogram

Description automatically generated

8/11/2020

**Sequential V3**

For this model we will explicitly experiment with the effect of increasing the number of convolutional layers on model accuracy. We now have an expanded dataset with POM videos from the LC compounds:

* 5CB
* 8CB
* D5
* D6
* D7
* D8
* M5
* M6
* M7
* M9
* M10

We choose to try input image sizes of 256x256 and 128x128 (powers of 2 help with GPU memory). The new videos have large dimensions of 2048x1088 pixels. We will split each one into 6 sub images of size 682x544, before cropping to square and resizing as with the previous set of images. Also, when splitting the data into train, validation and test sets, we now avoid putting images from the same video in more than one set, in order to prevent the data leakage observed in models V1 and V2. The new distribution of data is as follows:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Isotropic | Nematic | Cholesteric | Smectic | Totals |
| Training | 1500 | 1691 | 1549 | 1689 | 6429 |
| Validation | 400 | 471 | 405 | 457 | 1733 |
| Test | 200 | 208 | 178 | 287 | 873 |
| Totals | 2100 | 2370 | 2132 | 2433 | 9035 |

We first train six individual models with convolutional layer number increasing from 1 to 6. For each model, the first layer has 32 filters, with the number of filters doubling with each successive convolutional layer. Each convolutional layer, like the first models, has ReLU activations, “same” padding, stride of 1 and kernel size 3x3. In addition, each convolutional layer is followed by batch normalization and then max pooling with pool size 2x2. The final convolutional layer is followed by global average pooling instead of max pooling. Each model has 2 dense layers, first with 256 units and then 128, both with dropout rate 0.5 for regularisation, as well as batch normalisation, followed by a 4 unit output layer. We use less dense units than V1 and V2 in order to prevent overfitting. We use the same training specifications as for V1 and V2, aside from an increased early stopping patience of 100 epochs. We also train the models a second time with only horizontal and vertical flip augmentations, for input image sizes of

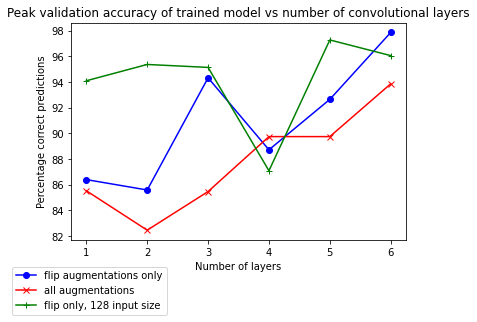
The results of the training are summarised in the tables below:

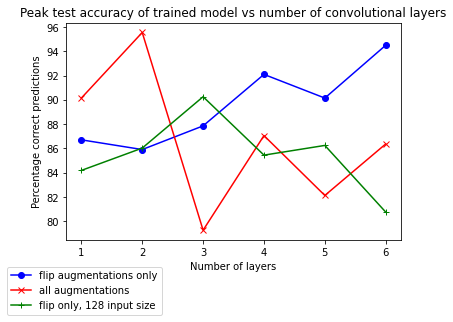
|  |  |  |  |
| --- | --- | --- | --- |
| **All image augmentations (same as models V1 and V2), input size 256x256** | | | |
| Number of convolutional layers | Validation accuracy (%) | Test accuracy (%) | Training time (s) |
| 1 | 85.53 | 90.15 | 6210 |
| 2 | 82.46 | 95.55 | 5520 |
| 3 | 85.47 | 79.27 | 5750 |
| 4 | 89.75 | 87.06 | 5474 |
| 5 | 89.75 | 82.13 | 6486 |
| 6 | 93.86 | 86.37 | 5290 |
| Mean | **87.80** | **86.76** | **5788** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Vertical and horizontal flip augmentations only, input size 256x256** | | | |
| Number of convolutional layers | Validation accuracy (%) | Test accuracy (%) | Training time (s) |
| 1 | 86.40 | 86.71 | 3040 |
| 2 | 85.59 | 85.91 | 2180 |
| 3 | 94.33 | 87.86 | 3040 |
| 4 | 88.72 | 92.10 | 2289 |
| 5 | 92.65 | 90.15 | 3108 |
| 6 | 97.86 | 94.50 | 3990 |
| Mean | **90.92** | **89.54** | **2941** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Vertical and horizontal flip augmentations only, input size 128x128** | | | |
| Number of convolutional layers | Validation accuracy (%) | Test accuracy (%) | Training time (s) |
| 1 | 94.10 | 84.19 | 2736 |
| 2 | 95.37 | 86.03 | 2470 |
| 3 | 95.14 | 90.26 | 3439 |
| 4 | 87.09 | 85.45 | 2261 |
| 5 | 97.28 | 86.25 | 3249 |
| 6 | 96.06 | 80.76 | 2926 |
| Mean | **94.17** | **85.49** | **2847** |

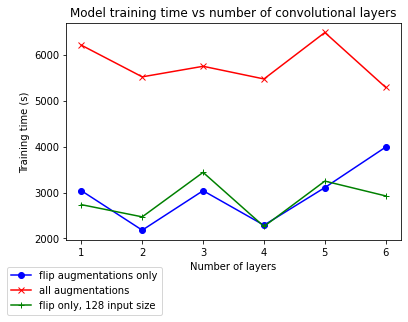
As can be seen from the mean accuracies, the models have a higher accuracy when trained using only flip augmentations. This is potentially due to the vastly expanded dataset requiring less augmentation for good regularisation. The 256x256 input models have similar validation and training accuracies in both cases, suggesting good model regularisation. The models with input size 128x128 have a higher mean validation accuracy than the 256x256 models, but the mean test set accuracy is much lower than the validation accuracy, suggesting that they are not well regularised. Since the training times do not differ by much for both input sizes with only flip augmentations, we will continue with future models using the 256x256 input size only. Plots of model accuracy on both validation and test sets vs number of convolutional layers are displayed below.





The general trend appears to be an increase in accuracy with the number of convolutional layers for the validation set, with not obvious trend for the test accuracies.

The mean training time using all augmentations is approximately double that with only flip augmentations, demonstrating that augmentation processing is a major computational expense. A plot of training time vs number of convolutional layers is displayed below.



There does not appear to be any correlation between training time and number of layers from this plot.

The architecture diagrams, training loss/accuracy graphs and test set prediction confusion matrices for each model with input size 256x256 are listed below.

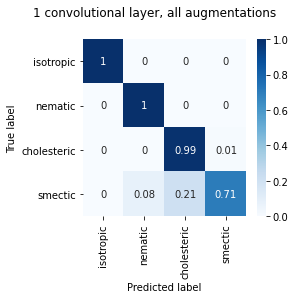
A picture containing graphical user interface

Description automatically generated**1 convolutional layer:**

All augmentations:

**A picture containing chart

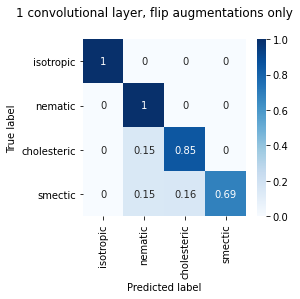
Description automatically generated**



Flip augmentations only:

Chart, histogram

Description automatically generated

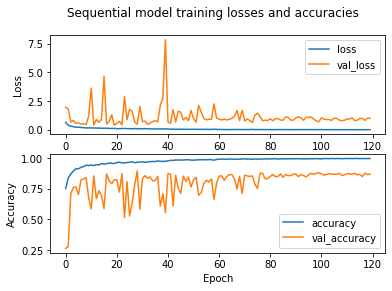


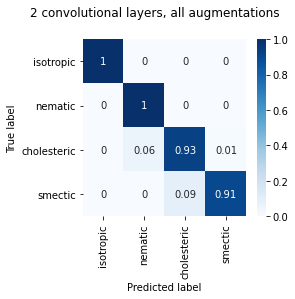
**2 convolutional layers:**

**A picture containing icon

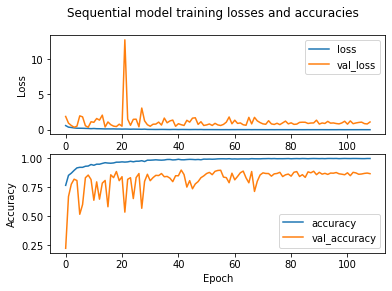
Description automatically generated**

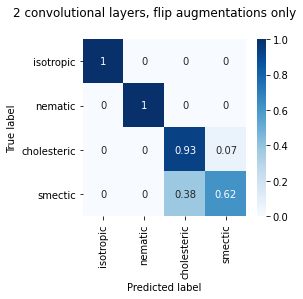
All augmentations:





Flip augmentations only:



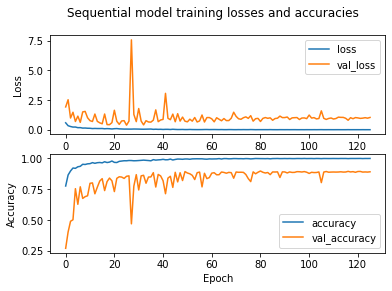


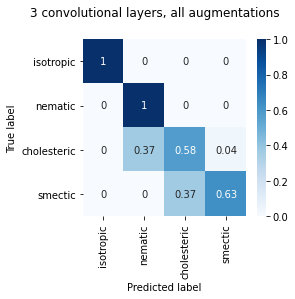
**3 convolutional layers:**

**A picture containing diagram

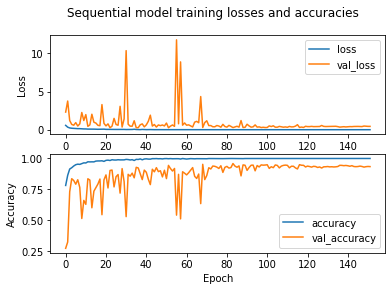
Description automatically generated**

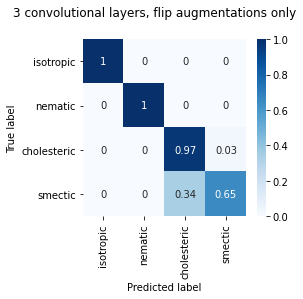
All augmentations:





Flip augmentations only:



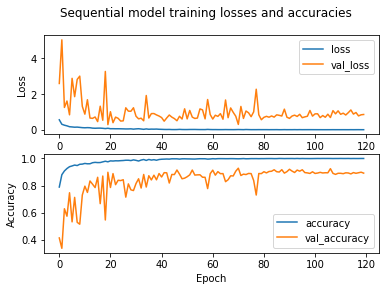


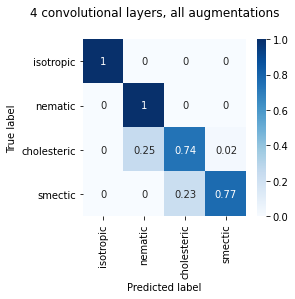
**4 convolutional layers:**

**A picture containing icon

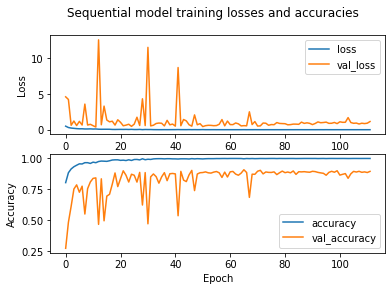
Description automatically generated**

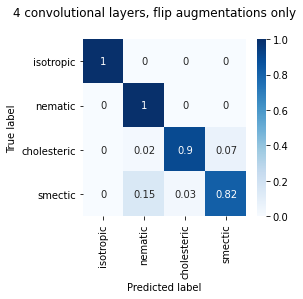
All augmentations:





Flip augmentations only:



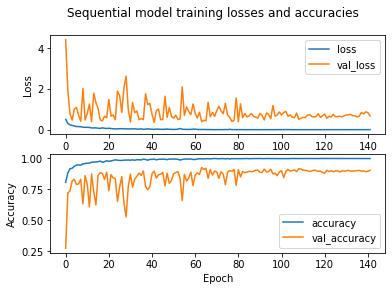


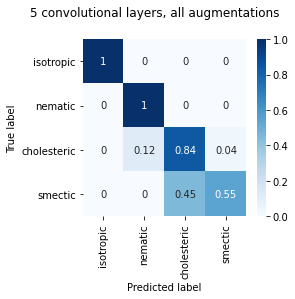
**5 convolutional layers:**

**Diagram

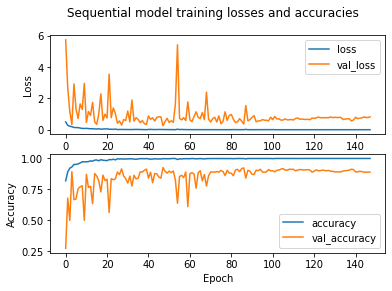
Description automatically generated**

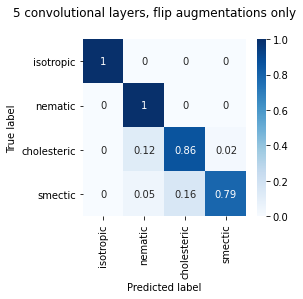
All augmentations:





Flip augmentations only:



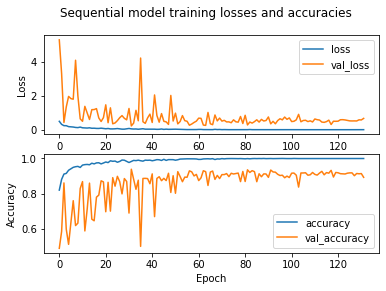


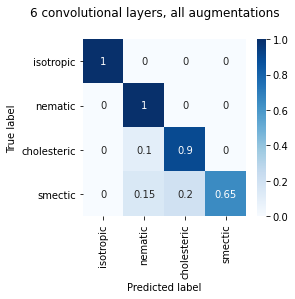
**6 convolutional layers:**

**A close up of a sign

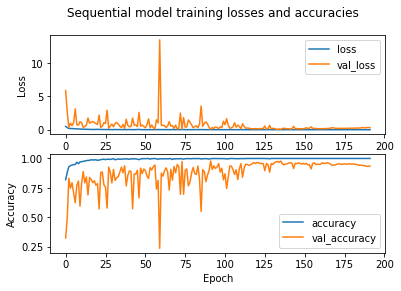
Description automatically generated**

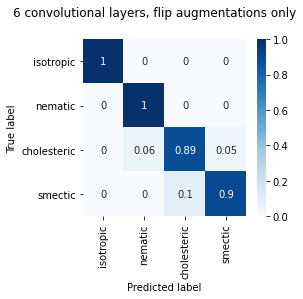
All augmentations:





Flip augmentations only:





It is clear from the confusion matrices that all models are highly accurate in identifying the isotropic and nematic phases whilst often confusing the cholesteric and smectic phases. This is potentially due to the similarities between certain cholesteric and smectic texture images.

17/11/2020

**Repeating the training**

We will now repeat all V3 training 2 more times and measure the accuracies in order to calculate a mean accuracy and uncertainty for each model, this time including the case with all augmentations and 128x128 image size. Uncertainty on the mean is taken as half the range of the values, due to the small sample size of 3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **All augmentations, 256x256 input size, validation set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 85.53 | 82.46 | 85.47 | 89.75 | 89.75 | 93.86 |
| 2nd run | 88.14 | 87.44 | 89.53 | 88.14 | 97.51 | 92.07 |
| 3rd run | 85.07 | 86.46 | 87.44 | 86.23 | 91.78 | 96.76 |
| Means | 86.25 | 85.45 | 87.48 | 88.04 | 93.01 | 94.23 |
| Uncertainty | 1.54 | 2.49 | 2.03 | 1.76 | 3.88 | 2.35 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **All augmentations, 256x256 input size, test set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 90.15 | 95.55 | 79.27 | 87.06 | 82.13 | 86.37 |
| 2nd run | 91.78 | 81.25 | 78.36 | 83.22 | 81.83 | 91.20 |
| 3rd run | 90.28 | 93.98 | 78.47 | 84.72 | 82.41 | 87.27 |
| Means | 90.74 | 90.26 | 78.70 | 85.00 | 82.12 | 88.28 |
| Uncertainty | 0.81 | 7.15 | 0.45 | 1.92 | 0.29 | 2.42 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Flip augmentations, 256x256 input size, validation set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 86.40 | 85.59 | 94.33 | 88.72 | 92.65 | 97.86 |
| 2nd run | 88.48 | 83.56 | 92.65 | 86.57 | 93.34 | 91.96 |
| 3rd run | 85.13 | 81.13 | 91.44 | 86.92 | 94.85 | 86.34 |
| Means | 86.67 | 83.43 | 92.81 | 87.40 | 93.61 | 92.05 |
| Uncertainty | 1.68 | 2.23 | 1.45 | 1.08 | 1.10 | 5.76 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Flip augmentations, 256x256 input size, test set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 86.71 | 85.91 | 87.86 | 92.10 | 90.15 | 94.50 |
| 2nd run | 89.00 | 94.33 | 88.19 | 86.46 | 86.69 | 84.49 |
| 3rd run | 97.57 | 94.21 | 80.90 | 81.37 | 95.83 | 89.47 |
| Means | 91.09 | 91.48 | 85.65 | 86.64 | 90.89 | 89.49 |
| Uncertainty | 5.43 | 4.21 | 3.65 | 5.36 | 4.57 | 5.01 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **All augmentations, 128x128 input size, validation set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 82.06 | 86.63 | 84.32 | 93.34 | 91.32 | 91.09 |
| 2nd run | 88.37 | 88.54 | 88.95 | 90.16 | 95.49 | 93.00 |
| 3rd run | 90.34 | 84.14 | 87.96 | 89.93 | 93.06 | 94.50 |
| Means | 86.92 | 86.44 | 87.08 | 91.14 | 93.29 | 92.86 |
| Uncertainty | 4.14 | 2.20 | 2.32 | 1.71 | 2.09 | 1.71 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **All augmentations, 128x128 input size, test set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 83.56 | 86.69 | 90.39 | 80.9 | 76.39 | 74.42 |
| 2nd run | 88.43 | 82.64 | 75.69 | 81.02 | 79.75 | 76.85 |
| 3rd run | 90.39 | 93.87 | 89.24 | 85.3 | 78.47 | 89.12 |
| Means | 87.46 | 87.73 | 85.11 | 82.41 | 78.20 | 80.13 |
| Uncertainty | 3.42 | 5.62 | 7.35 | 2.20 | 1.68 | 7.35 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Flip augmentations, 128x128 input size, validation set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 94.10 | 95.37 | 95.14 | 87.09 | 97.28 | 96.06 |
| 2nd run | 87.96 | 87.44 | 87.38 | 92.19 | 93.00 | 93.23 |
| 3rd run | 87.73 | 90.57 | 95.89 | 93.23 | 95.95 | 95.14 |
| Means | 89.93 | 91.13 | 92.80 | 90.84 | 95.41 | 94.81 |
| Uncertainty | 3.19 | 3.97 | 4.26 | 3.07 | 2.14 | 1.42 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Flip augmentations, 128x128 input size, test set accuracies** | | | | | | |
| Number of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 84.19 | 86.03 | 90.26 | 85.45 | 86.25 | 80.76 |
| 2nd run | 92.59 | 93.40 | 96.88 | 82.52 | 87.96 | 83.22 |
| 3rd run | 95.25 | 90.86 | 84.84 | 86.11 | 92.01 | 86.00 |
| Means | 90.68 | 90.10 | 90.66 | 84.69 | 88.74 | 83.33 |
| Uncertainty | 5.53 | 3.69 | 6.02 | 1.80 | 2.88 | 2.62 |

The means and uncertainties are displayed in the plots below.

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

The overall trend for validation accuracy appears to be an increase in accuracy with number of layers, with the flip augmentations and 128x128 image size performing best. The test set appears to have a slight negative correlation, potentially due to poor model regularisation in the higher numbers of layers.

26/11/20

**Second model, smectic phase classification**

We now attempt to classify smectic textures from the following phase categories:

* Fluid smectic (smectic A and C)
* Hexatic (smectic I and F)
* Soft crystal (smectic X)

This could prove to be more challenging due to the greater similarity between the images from the categories. Image splitting, cropping and pre-processing is carried out in the same way as for the V3 models, using frame data from the same set of videos. We will first use images of size 256x256 and only flip augmentations. The data is distributed as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Fluid smectic | Hexatic | Soft crystal | Totals |
| Training | 1759 | 486 | 600 | 2845 |
| Validation | 372 | 90 | 144 | 606 |
| Test | 389 | 90 | 96 | 575 |
| Totals | 2520 | 666 | 840 | 4026 |

The current dataset is highly imbalanced in the favour of the fluid smectic class due to a limited number of hexatic and soft crystal videos. We first train some sequential models similar to the ones for the 4 phase classification, but with reduced overall capacity to prevent overfitting on the small dataset. After trying 1-6 convolutional layers, the common result is mode collapse into the fluid smectic class due to the imbalance, resulting in test set confusion matrices like this:

Chart

Description automatically generated

We will instead try a model based on Google’s “Inception” architecture, in which multiple convolutional layers with different kernel sizes are placed in parallel in order to extract features of different sizes, before being concatenated together. We will refer to a structure of this type as an inception block. We use flip augmentations only and 256x256 image size and the same training configurations as for V3. Representations of models with increasing numbers of inception blocks from 1 to 3 are displayed below, along with their accuracies and test set confusion matrices.

**Table

Description automatically generated1 inception block**:

Validation accuracy: 82.29%

Test accuracy: 72.24%

Chart

Description automatically generated

**Table

Description automatically generated2 inception blocks:**

Validation accuracy: 78.13%

Test accuracy: 83.27%

Chart, waterfall chart

Description automatically generated

**A close up of text on a black background

Description automatically generated3 inception blocks:**

Validation accuracy: 81.01%

Test accuracy: 85.48%

Chart

Description automatically generated

As can be seen from the confusion matrices, the models are relatively accurate at identifying the fluid smectic class, most likely due to the large imbalance in favour of fluid smectic. The model with 3 inception blocks performed best overall in terms of accuracy. We may need more data for the hexatic and soft crystal classes to improve these models further.

10/12/20

**Smectic A/C binary classifier**

We shall now investigate the binary classification of the fluid smectic phases A and C. These phases’ structures do not differ by much, resulting in similar textures. The distribution of the data set is as follows:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Smectic A | Smectic C | Totals |
| Training | 719 | 1067 | 1786 |
| Validation | 174 | 183 | 357 |
| Test | 204 | 198 | 402 |
| Totals | 1097 | 1448 | 2545 |

We will use 256x256 image size and flip augmentations only. We will test the same inception models as the previous smectic classification (aside from having 1 unit in the final dense layer as it is a binary classification task), as well the V3 sequential models but with reduced channels in each layer to reduce overfitting. The new number of channels for each model in each layer, in order of convolutional layers is:

* 1 conv layer: 32
* 2 conv layers: 16, 32
* 3 conv layers: 8, 16, 32
* 4 conv layers: 4, 8, 16, 32
* 5 conv layers: 2, 4, 8, 16, 32
* 6 conv layers: 1, 2, 4, 8, 16, 32

The only change to the usual training specifications is the loss function, which will now be the binary cross entropy instead of categorical cross entropy. Each model is trained 3 times, the results are displayed in the tables below. The uncertainty is calculated as half the range as before.

|  |  |  |  |
| --- | --- | --- | --- |
| **Inception models, validation set accuracies** | | | |
| No. of inception blocks | 1 | 2 | 3 |
| 1st run | 92.05 | 98.3 | 98.58 |
| 2nd run | 98.01 | 92.61 | 100 |
| 3rd run | 100 | 100 | 91.48 |
| Mean | 96.69 | 96.97 | 96.69 |
| Uncertainty | 3.98 | 3.69 | 4.26 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Inception models, test set accuracies** | | | |
| No. of inception blocks | 1 | 2 | 3 |
| 1st run | 95.57 | 96.35 | 96.61 |
| 2nd run | 95.57 | 94.53 | 94.27 |
| 3rd run | 95.83 | 99.22 | 93.49 |
| Mean | 95.66 | 96.70 | 94.79 |
| Uncertainty | 0.13 | 2.34 | 1.56 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sequential models, validation set accuracies** | | | | | | |
| No. of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 94.03 | 92.61 | 86.08 | 99.15 | 98.58 | 94.03 |
| 2nd run | 91.48 | 88.07 | 94.03 | 97.44 | 96.59 | 95.45 |
| 3rd run | 90.06 | 96.31 | 95.45 | 98.3 | 92.9 | 98.86 |
| Mean | 91.86 | 92.33 | 91.86 | 98.3 | 96.02 | 96.12 |
| Uncertainty | 1.99 | 4.12 | 4.69 | 0.85 | 2.84 | 2.41 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sequential models, test set accuracies** | | | | | | |
| No. of conv. layers | 1 | 2 | 3 | 4 | 5 | 6 |
| 1st run | 98.18 | 96.61 | 97.14 | 96.09 | 91.41 | 96.09 |
| 2nd run | 96.35 | 97.14 | 94.53 | 97.4 | 96.09 | 93.49 |
| 3rd run | 85.42 | 97.14 | 86.46 | 97.66 | 97.14 | 90.89 |
| Mean | 93.32 | 96.96 | 92.71 | 97.05 | 94.88 | 93.49 |
| Uncertainty | 6.38 | 0.26 | 5.34 | 0.78 | 2.86 | 2.6 |

The means and uncertainties are plotted below.

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

All of the models performed well on this task, with none having an accuracy below 90%. The inception models performed best overall compared the sequential ones, but the single model with the highest mean accuracy in both validation and test sets was the 4 layer sequential one.

15/12/20

The confusion matrices below are for the saved models with the highest in (a) and lowest in (b) test set accuracies. The highest accuracy was a 2 block inception model at 99.22%, and the lowest was a 1 convolutional layer sequential model with 85.42%. The inception model’s accuracy is extremely high, potentially due to a “lucky” fluctuation. The sequential model misidentifies 35% of smectic A phases as smectic C, possibly due to a slight imbalance in the dataset in favour of smectic C.

**Chart, waterfall chart

Description automatically generated**

05/01/21

**Future work**

Some ways to extend the project next semester are:

* Expanding the dataset, especially the hexatic smectic and soft crystal phases, in order to train models that can classify a greater a number of phases with high accuracy.
* Investigate new types of state-of-the-art CNN architectures.
* Attempt to implement transformer networks, a different type of deep learning algorithm that has only recently begun to show high potential in image classification.
* If we can obtain the correct labelled data, regression models for properties of liquid crystals such as cholesteric pitch length could be implemented.

08/02/2021

**Timeline for second semester**

1. Improve dataset
2. Consider more systematic uncertainty evaluation
3. Consider metrics for unbalanced datasets
4. Consider methods to tackle class imbalance
5. Implement and investigate transformer networks
6. Investigate further state of the art CNNs
7. Ambitious goal: accurate classifier of many phases

**Minimum uncertainty**

Going forward, a minimum percentage uncertainty will be assigned to model test accuracy predictions, as the square root of the total number of test set examples divided by itself.