Notes -

Flower Dataset -

Morphological Transformations -

1. Thresholding

convering RGB to Gery scale; threshold intensity above is 1 and below is 0

2. Erosion

Shrinks bright parts, enlarges dark parts

3. Dilation

shirnks dark parts, enlarges bright parts

4. Opening

Erosion followed by dilation

Remove small bright parts

5. Closing

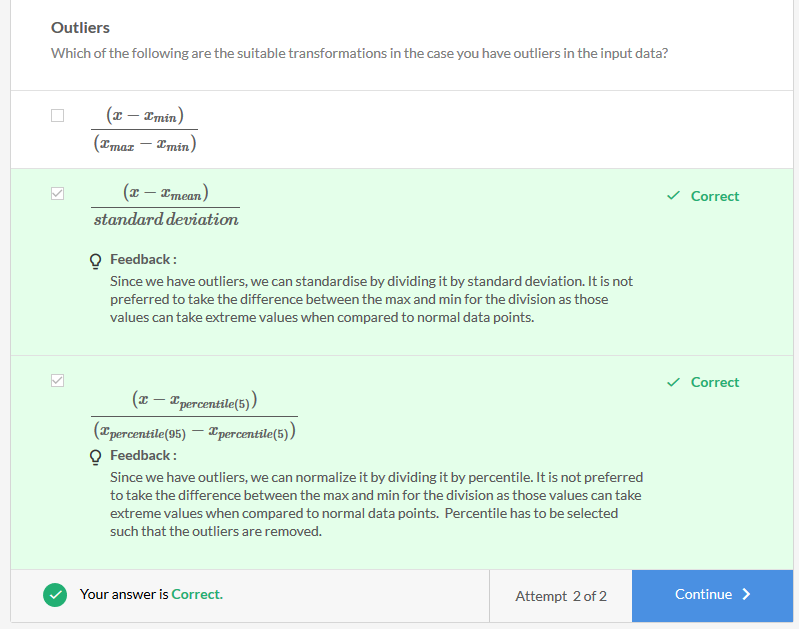
Dilation followed by erosion

rmemoves small dark parts

Normalisation makes the training process much smoother. This is an important preprocessing step, so let's discuss it briefly.

Formula for normalisations are:

* (image−np.min(image))/(np.min(image)−np.max(image))
* (image−np.percentile(image,5))/(np.percentile(image,95)−np.percentile(image,5))



1. Reasons for Normalisation

* Contrast and lighting conditions
  + We need to account for variation in pictures, or different settings of machines taking images
* Gradient Propagation
  + Normalised images make for much better gradient propagation

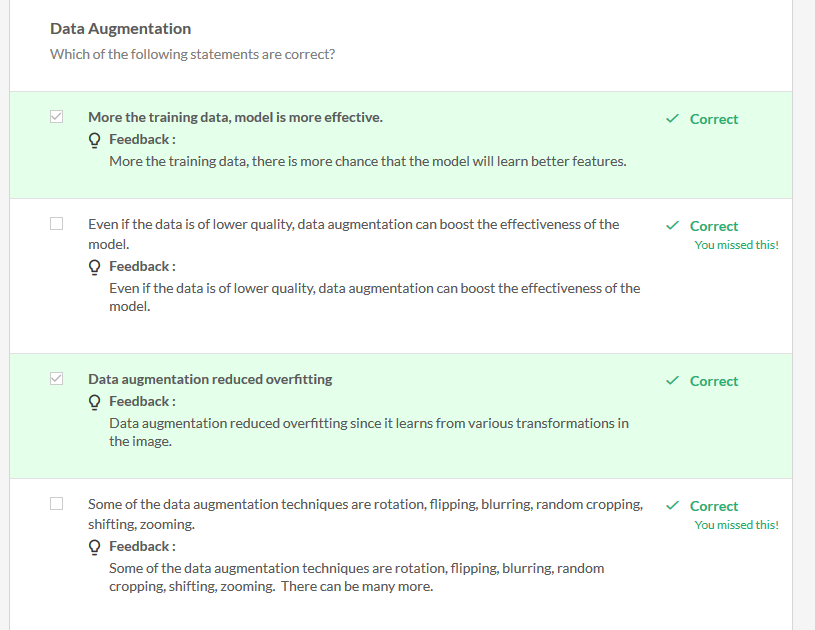
**Data Augmentation -**

creating extra data out of existing data.

Types:

1. Linear Transformation
   1. Matrix operations
      1. Rotation
      2. Flipping
2. Affine Transformations
   1. Translation followed by Linear Transformations

pooling increases the Invariance (Pooling is the process of extracting the features from the image output of a convolution layer. This will also follow the same process of sliding over the image with a specified pool size/kernel size. Max Pooling is being used widely and it will just keep the highest number in the pool and discard the rest)



Hyperparameter Tuning

* Learning rate & Variation and Different optimizers
* Different types of Augmentation

**Summary**

In this session, you learnt to set-up a typical end-to-end pipeline for training CNNs. Specifically, you learnt the following:

**Data Preprocessing**

**Morphological Transformations**: This refers to changing the shape and size of images. The typical transformations are erosion, dilation, opening and closing.

**Augmentation**: Refers to making changes related to rotation, translation, shearing, etc. Augmentation is often used in image-based deep learning tasks to increase the amount and variance of training data. Augmentation should only be done on the training set, never on the validation set.

**Normalisation**: Refers to rescaling the pixel values so that they lie within a confined range. One of the reasons to do this is to help with the issue of propagating gradients.

**Network Building**

Choosing the architecture: For this demo, we used the 'ResNet' architecture. Its biggest upside is that the 'skip connections' mechanism allows very deep networks.

**Ablation Experiments**: These refer to taking a small chunk of data and running your model on it - this helps in figuring out if the model is running at all.

**Overfitting on Training Data**: This tells you whether the model is behaving as expected or not.

**Metrics**: Depending on the situation, we choose the appropriate metrics. For binary classification problems, AUC is usually the best metric.

**Hyperparameter tuning**: We tune hyperparameters such as the learning rate, augmentation of images, batch size, etc. Also, we only change the architecture of the network if we have already tried tuning all other hyperparameters.

1. Data Preparation:
   1. Made sure all our images were of the same resolution.
   2. Placed the images in two different folders - 'rose' and 'daisy'. This method will work for any application where you're trying to train using images.
2. Data Pre-processing: Morphological Operations
   1. Did thresholding on the image - converted it from a grey image to a binary image.
   2. Looked at Erosion, Dilation, Opening, Closing.
3. Data Pre-processing: Normalisation
   1. Understood the need for normalisation.
   2. Saw some commonly used methods of normalisation.
4. Data Pre-Processing: Augmentation
   1. Understood the need for data augmentation.
   2. Learnt about two types of transformations for augmentation - linear and affine.
   3. Saw different ways to augment - translation, rotation, scaling, etc.
5. Model Building
   1. Running ablation experiments
   2. Overfitting on a smaller version of the training set
   3. Hyperparameter tuning
   4. Mode training and evaluation

**Application to Chest X-rays**

Data set from - <https://www.kaggle.com/nih-chest-xrays>

For the flowers dataset, we did all the augmentations we possibly could. However, for the CXR data, we had some specific constraints.

1. Vertical flip needs to be set as 'False'. This is because CXR images have a natural orientation - up to down.
2. We cannot (i.e. should not) do a centre crop for CXR images, as the anomaly can be in an area outside the cropped portion of the image.

What did we do differently here as compared to the flowers dataset?

1. Since the CXR images are not "natural images", we do not use the "divide by 255" strategy. Instead, we take the max-min approach to normalisation. Since you do not know for sure that the range of each pixel is 0-255, you normalise using the min-max values.
2. In the conditional statement 'if mode == train', use a random number generator so that only a fraction of the images get transformed (rather than all of them).

## Data Preprocessing

****Augmentation:****Because this is a specific industry use-case, we put constraints of the augmentations that we can or cannot do (e.g. no vertical flipping). Also, we ensure that augmentation is only done on the training data.

****Resizing:**** We need to be careful while resizing images. The two most common ways to resize images are to 1) crop and 2) to resize by 'averaging' the pixels (similar to pooling). The risk of cropping is that you lose some pixels, while the risk of resizing by averaging is that the finer edges etc. may be lost. Since in most realistic problems you have to reduce the image size (to meet computational constraints), you should choose between these two carefully. If you do not care about finer edges etc. (e.g. when coarse-grained features will suffice), you can do averaging. If you care about finer features but only a central region of the image, you can use cropping.

****Normalisation:****We usually normalise using a min-max range. Make sure to not divide by 255 blindly since all images (such as X-Ray scans) don't necessarily have the range 0-255.

## Network Building

****Architecture:****We used a ResNet together with a decaying learning rate and a weighted cross-entropy loss (to account for class imbalance). We used AUC as the metric.

****Metrics:**** While choosing a metric for medical images with a prevalence problem, we pick recall over precision. We don't want to miss out on any cases of effusion. In any case, working with AUC and a manual threshold is the best option.

****Weighted Cross-entropy:****This loss is used when the error in one direction is costlier than the other, for example, it is much more undesirable to diagnose 'effusion' as 'normal' than the other way around. This is done by assigning 'higher weights' to the errors in certain classes.

### Industry Demo: Detecting Vehicles in Videos

Pre-processing Frames - I

* converting the colour image to grayscale,
* blurring it ( Gaussian blur or some other blurring),
* applying thresholding,
* dilation
* Erosion

Steps

1. Capture 2 consecutive frames from video
2. Convert to gray scale
3. Difference between two frames or images
4. Threshold the image
5. Remove noise using dilation and erosion

Image Processing steps:

1. Convert RGB to Gray Image
2. Convert Gray to Binary Image - helps in identifying the edges

Gaussian Blur: Image smoothens but it may also result in blurring the edges

1. Removes Granininess
2. Kernl maintains central pixel and smoothens out the pixels around it.
3. Perform Convolution using the Kernel

Structuring Element - Trade off

1. Small kernel - imperfect dilation, fast processing
2. Large kernel - perfect dilation, slow processing

* we perform morphological operations (such as dilation and erosion) on the thresholded image to remove the white noise from the image
* after the thresholded image is clear, there are lots of holes in it
* identify the outline of the vehicle so that it can be extracted using contour
* Dilation should be more than erosion since dilation will expand the broken boundary and also fills the gaps in the vehicle. Note that the vehicle is represented by white and erosion expands the bright region.

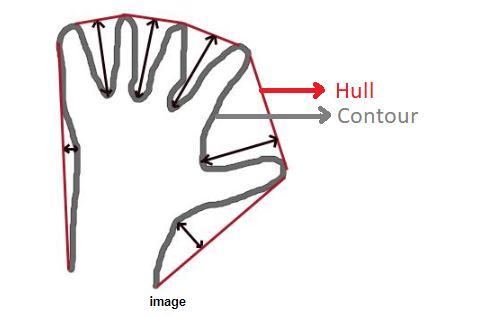
Contours:

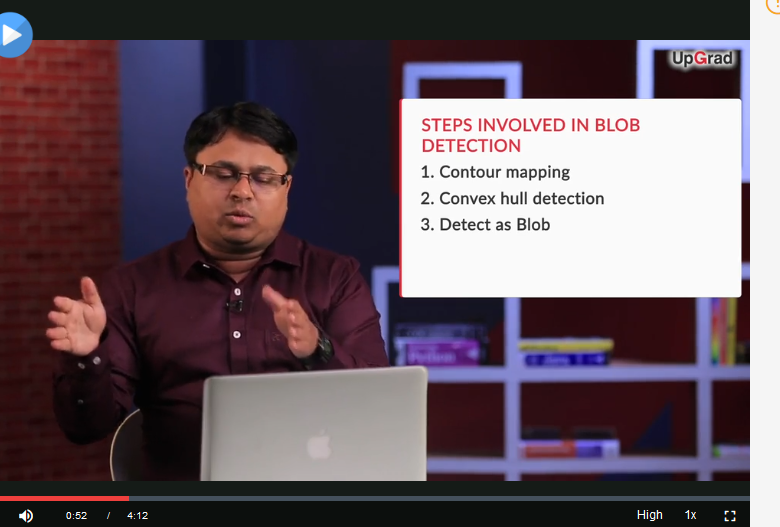
1. Connected edges, differs from edge detection - Contours tells if the edges are connected.
2. Used to find the objects / object segmentation
3. Cv2.findcontours (binImage)

The edges only show the boundary where the difference between the pixels is maximum. There is no relation between two or more pixels. Whereas, contours are curves joining all the continuous points (along with the boundary), having the same colour or intensity

* three main parts of object detection are:
  + Contour mapping
  + Convex hull detection
  + Blob detection and validation

# Blobs and Hulls





**OBJECT TRACKING**