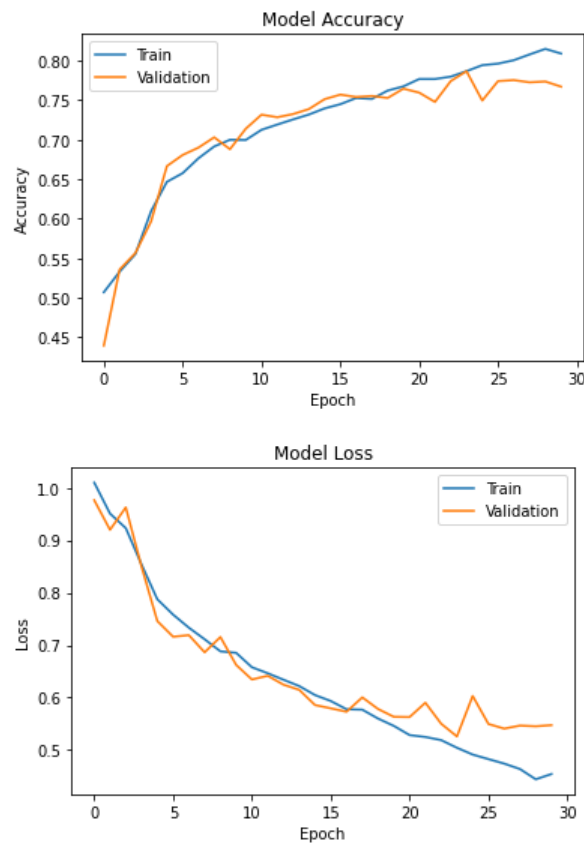# Model Summary - Hackathon 1

For hackathon 1: Age Detection of Actors, 4 models are designed and tested. First model gives approximately 74% accuracy which convert images from different sizes to 64x64x3 size and take it as input. Then followed by 4 conv layer, 4 max pool layer and then passes to fully connected layers and also given a dropout of 20%. The batch size is taken as 64 and number of epochs is 50. The second model designed can give accuracy of 76% within 30 epochs with batch size of 72 and also given a dropout of 20%. In model 2 the number of hidden layers are increased. But after 76% accuracy on validation data, train accuracy keeps increasing whereas validation accuracy seems reducing. To improve this, model 3 is implemented which can still give 76% accuracy. The only differences is that reduced some number of hidden layers and nodes and increased dropout from 0.2 to 0.4 in model 3. Also added batch normalization in model 3. Model 3 shows significantly better result than other models.

Accuracy and loss in model 3

MODEL 3:

conv2d_70_input: InputLayer	input:	[(?, 32, 32, 3)]
	output:	[(?, 32, 32, 3)]

conv2d_70: Conv2D	input:	(?, 32, 32, 3)
	output:	(?, 32, 32, 64)

conv2d_71: Conv2D	input:	(?, 32, 32, 64)
	output:	(?, 32, 32, 64)

max_pooling2d_36: MaxPooling2D	input:	(?, 32, 32, 64)
	output:	(?, 16, 16, 64)

conv2d_72: Conv2D	input:	(?, 16, 16, 64)
	output:	(?, 16, 16, 24)

conv2d_73: Conv2D	input:	(?, 16, 16, 24)
	output:	(?, 16, 16, 24)

max_pooling2d_37: MaxPooling2D	input:	(?, 16, 16, 24)
	output:	(?, 8, 8, 24)

conv2d_74: Conv2D	input:	(?, 8, 8, 24)
	output:	(?, 8, 8, 12)

conv2d_75: Conv2D	input:	(?, 8, 8, 12)
	output:	(?, 8, 8, 12)

conv2d_76: Conv2D	input:	(?, 8, 8, 12)
	output:	(?, 8, 8, 12)

max_pooling2d_38: MaxPooling2D	input:	(?, 8, 8, 12)
	output:	(?, 4, 4, 12)

dropout_12: Dropout	input:	(?, 4, 4, 12)
	output:	(?, 4, 4, 12)

flatten_8: Flatten	input:	(?, 4, 4, 12)
	output:	(?, 192)

dense_61: Dense	input:	(?, 192)
	output:	(?, 120)

dense_62: Dense	input:	(?, 120)
	output:	(?, 240)

dense_63: Dense	input:	(?, 240)
	output:	(?, 120)

dense_64: Dense	input:	(?, 120)
	output:	(?, 120)

dense_65: Dense	input:	(?, 120)
	output:	(?, 120)

dense_66: Dense	input:	(?, 120)
	output:	(?, 60)

dense_67: Dense	input:	(?, 60)
	output:	(?, 20)

dense_68: Dense	input:	(?, 20)
	output:	(?, 3)