It doesn’t state that ConvLSTM will give you better results than Conv3D. The explanation should be as detailed as possible so that the logic behind the decision is conveyed. Also, there are a lot of things you can experiment with in the generator function and elsewhere. Please do not forget to specify the exact metric values, here Accuracy which drives your decision.

You can draw inspiration from the concepts taught in the Industry demo in CNNs to experiment with the data and different architectures.

Experiment Number	Model	Case	Result	Decision + Explanation
1	**Conv3D**	**Batch size = 70 image size(160,160, 30, 3)**	**OOM Error: Out of memory exception**	**This means that the GPU memory cannot fit that much into memory. Reduce to 50.**
2	**Conv3D**	**Batch size = 40 30 Epochs image size(160,160, 30, 3)**	**Validation Categorical Accuracy: 0.3 Training Categorical Accuracy: 0.98**	**Model is overfitting with: with a subset of the data.**
3	**Conv3D**	**Use image size(160,160, 30, 3) Added some more layers to the model**	**ResourceExhaustedError: Graph execution error**	**Model not trainable as a lot of parameter Decision: Resize the image to have smaller dimensions**
4	**Conv3D**	**Use image size(120,120, 30, 3) Set model padding to retain same image dimensions**	**ResourceExhaustedError: Graph execution error**	**Model not trainable as a lot of parameter Decision: Allow shrinkage of the dimensions**
5	**Conv3D**	**Image size (120,120, 15, 3) Using half of the sequences**	**ValueError: Error while initializing model**	**Model not trainable as the shrinkage get dimension to zero or less Decision: Increase the number of the sequences**

6	Conv3D	Image size (80,80, 30, 3) Using half of the sequences 30 epochs The layers were also reduced and padding applied to reduce dimension reduction More data was used in training	Model was not learning categorical accuracy: 0.203 validation categorical accuracy: 0.1180	It appears that the cropping has removed some important features required to classify the image sequences. Decision: reduce cropping and use resize to reduce size of image but still maintain most of the image features needed for classification
7	Conv3D	Image size (80,80, 15, 3) Using half of the sequences The layers were also reduced and padding applied to reduce dimension reduction	Faster training categorical accuracy: 1.0000 validation categorical accuracy: 0.2000	Model is overfitting because it was trained on a small part of the data
8	Conv3D	I changed model architecture by replacing global average pooling by flattening Used all the train and validation data available	categorical accuracy: 0.9759 validation categorical accuracy: 0.2400 validation loss as consistently increasing while training loss was consistently reducing	The model seem to perform poorly on validation data because it is unable to identify the patterns there. There are still signs of overfitting
9	Conv3D	Augmented the validation data by adding 10% of	categorical accuracy: 0.9759	This seems to be the best possible case with the Conv3D model.

		train data to validation data increased the dropout on final out before softmax introduced again the Global average pooling Image size: (80,80, 15, 3) Batch size: 80	validation categorical accuracy: 0.2638	The data may be insufficient to adequately train the model so that it does not learn the specific data set but generalizes overall and predicts more accurately unseen data.
10	CNN_GRU	Batch size: 50 Image size: (120,120, 30, 3) Using only 20% of data for training gru_cells: 15	categorical accuracy: 0.9900 validation categorical accuracy: 0.2250	It can be seen that there is overfitting. Decision: increase drop out and change input to GRU model a bit
11	CNN_GRU	pretrained model pooling = max Image size: (120,120, 30, 3) Using only 20% of data for training gru_cells: 15	categorical accuracy: 0.29000 validation categorical accuracy: 0.10850	The learning of the model seems to have been greatly impacted by this singular modification.
12	CNN_GRU	pretrained model pooling = avg Image size: (120,120, 15, 3)	categorical accuracy: 1.0000 validation categorical accuracy: 0.2000	The validation accuracy was constantly 0.2000 where as the validation loss was increasing constantly and the training loss constantly reduce

		Using only 20% of data for training dropout = 0.5 before final output gru_cells: 15		Remove: dropout in final and increase time series data for learning
13	CNN_GRU	pretrained model pooling = avg Image size: (80,80, 15, 3) batch_size = 100 Using only 20% of data for training gru_cells: 15 Epochs: 30	categorical accuracy: 0.9100 validation categorical accuracy: 0.4000	This could have significantly improved with more epochs Decision: Try with full data set and more epochs
14	CNN_GRU	pretrained model pooling = avg Image size: (80,80, 15, 3) batch_size = 100 Using full data set gru_cells: 15 Epochs: 30	categorical accuracy: 0.9698 validation categorical accuracy: 0.6000	There could be some more improvements if trained over more epochs.
15	CNN_LSTM	pretrained model pooling = avg Image size: (80,80, 15, 3)	categorical accuracy: 0.6700 validation categorical	The result is a bit indecisive because training accuracy should be greater than validation accuracy.

		batch_size = 100 Using only 20% of data for training units: 5 Epochs: 40	accuracy: 0.4250	Decision: increase the number of units and use the full data set to train the LSTM model.
16	ConvLSTM	pretrained model pooling = avg Image size: (80,80, 15, 3) batch_size = 100 Full data set units: 5 Epochs: 30	categorical accuracy: 0.4100 validation categorical accuracy: 0.4500	.
Final Model	CNN_GRU.	pretrained model pooling = avg Image size: (80,80, 15, 3) batch_size = 100 Using full data set gru_cells: 15 Epochs: 30	categorical accuracy: 0.9698 validation categorical accuracy: 0.6000	This is the best model that has the least overfitting tendency and the best generalization and accuracy on validation data.