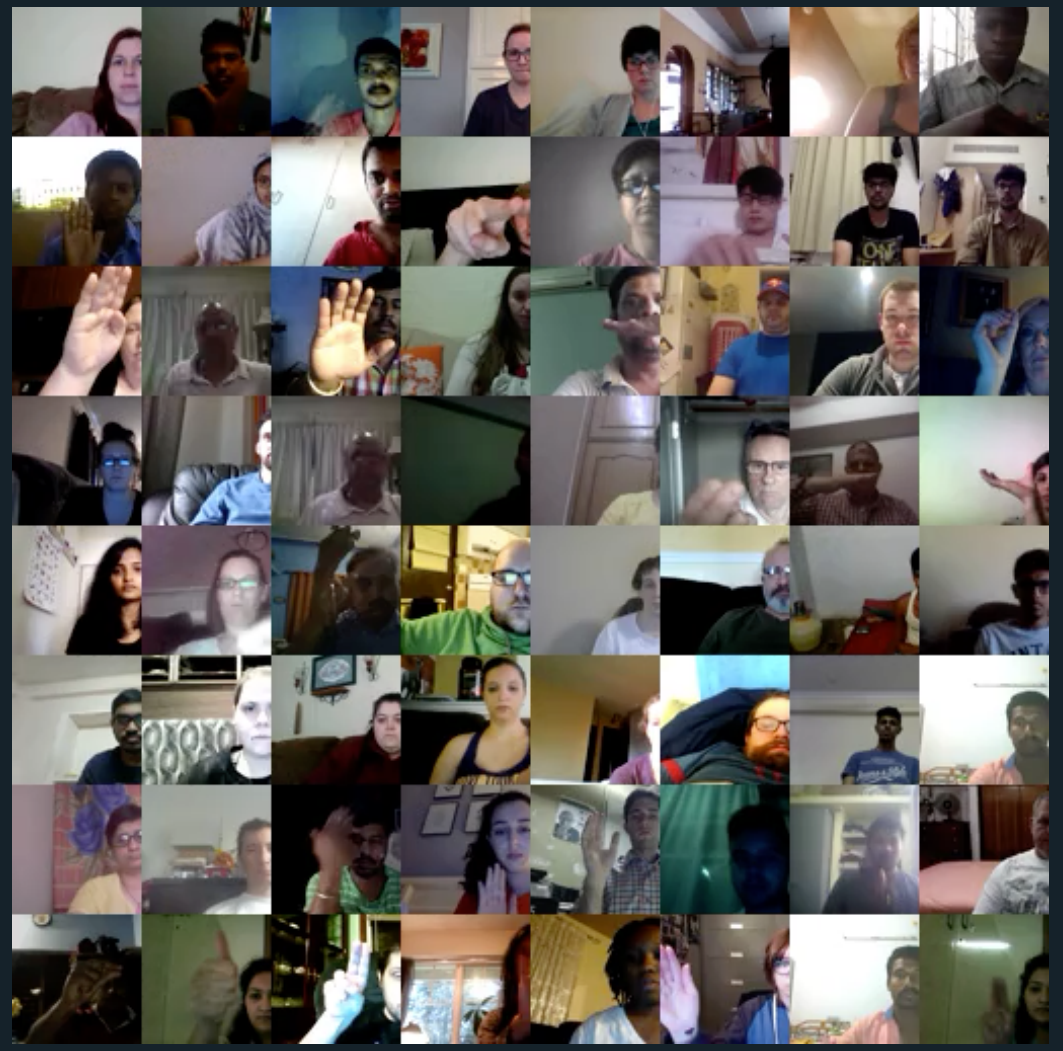
Human Hand Gesture Classification with 3D Convolutional Neural Network and ResNet



Video Classification: Why is it useful?

Training neural networks to classify video content, namely of human hand actions, has implications for several industries.

Examples

- Machine translation/recognition of sign language
- The reduction of driver distractions by increasing hands-free capabilities in vehicles
- Video indexing/categorization
- Automated surveillance
- Virtual reality software
- Human emotion recognition/classification

Description of 20BN-JESTER Dataset

- 20BN-JESTER dataset is comprised of 148,097 video clips with 27 possible labels of various human hand gestures
- Labels consist of human hand gestures such as swiping right, swiping left, giving a thumbs up, and waving in a clockwise or counterclockwise direction
- Video clips are disaggregated into JPG images, which were extracted from the videos at 12 frames per second
- Due to varying video length, number of JPG files per video varies
- All JPG images are of 100px height and variable width

Data Pre-Processing

- Because videos vary in length, a median number of 36 frames was used as the standard duration length
- Videos shorter than 36 frames were padded on both “ends”, with the beginning and ending frame of that video
- For videos longer than 36 frames, frames were randomly dropped so that only 36 frames were kept
- Each JPG image was standardized to have a height of 100 pixels and width of 176 pixels.
    - Some videos less than 176 pixels wide; these were zero-padded on either side to make them 176 pixels wide

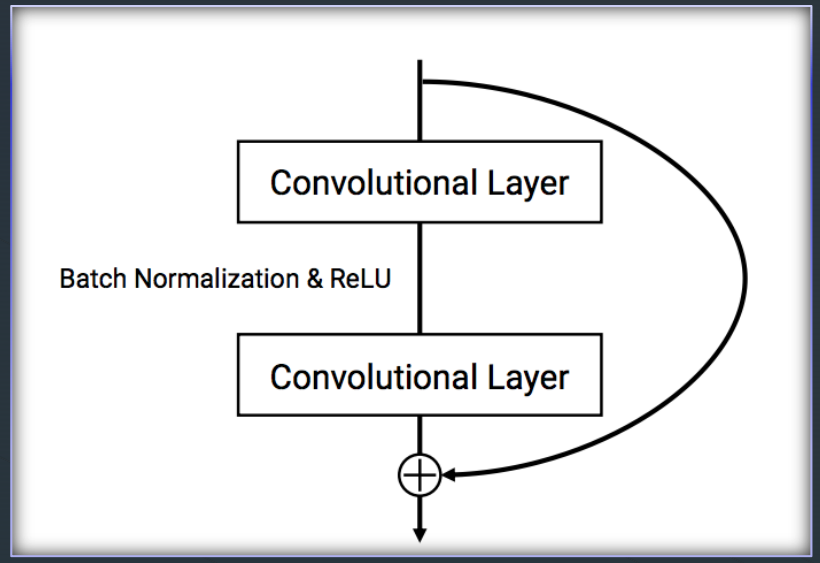
Network Architecture for Video Classification Problems

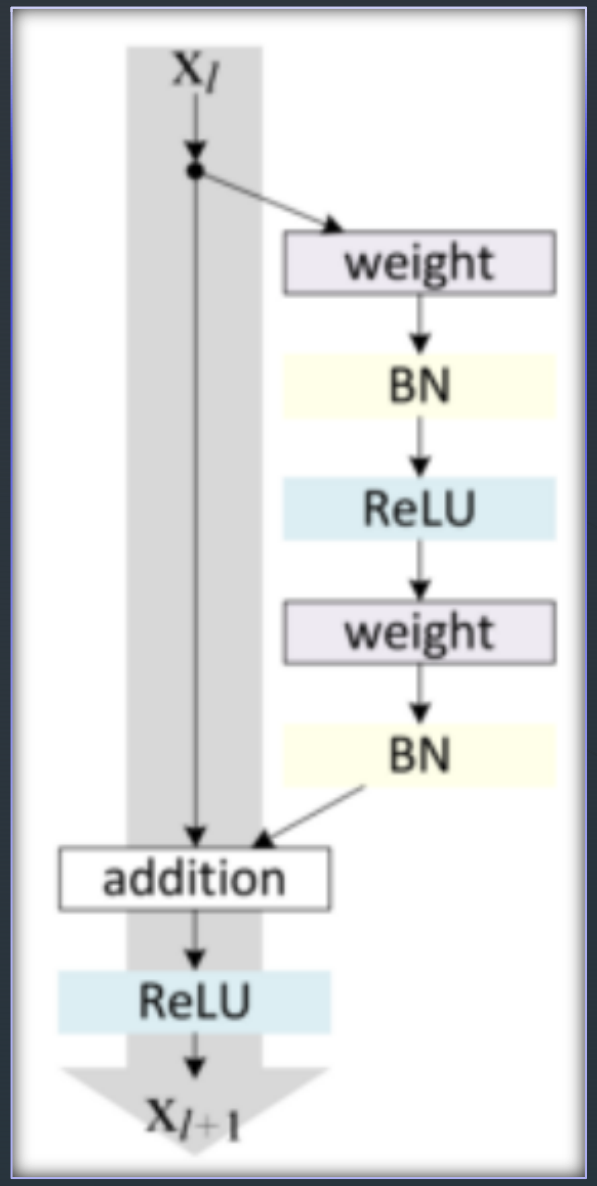
Two of the more popular approaches for video classification problems are CRNNs and 3D ResNets:

CRNNs consist of convolutional networks/blocks that feed into a recurrent neural network that uses time delays to learn the relationships between the image features at different time steps.

3D ResNets (residual networks) are a type of 3D convolutional network:

- Usually very deep with many sets of convolution and batch normalization layers
- Three dimensions are time (frames in this case), height, and width
- Key component that differentiates ResNet from traditional concolution layer: the input of the last layer is added back into the transformed output





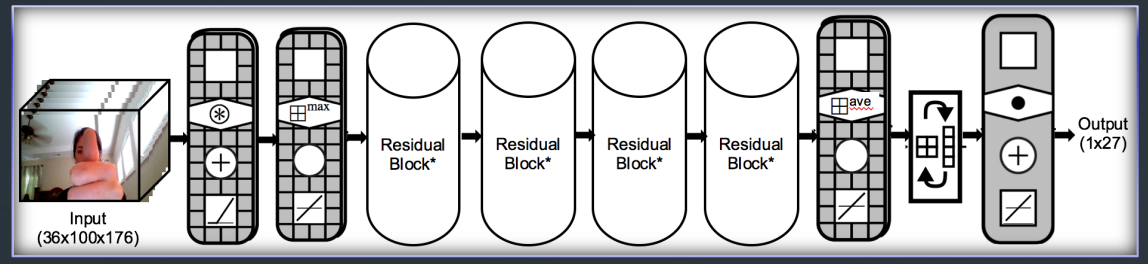
- Each residual block aims to learn the true underlying function, but because the inputs are added back into the output of the weight layers, weights are really being updated in order to learn the residual - the difference between the input and the true underlying function

- This can help fix the “degradation problem”, or “vanishing gradient”, where accuracy can degrade with networks that are very deep

The sequence of layers in each 3D ResNet block consists of: Convolutional layer -> Batch normalization -> ReLU -> Convolutional layer -> Batch normalization -> Add residual input -> ReLU

Proposed Network Architecture: 3D ResNets

- The input, which is a sequence of images, goes through a 3D convolution layer followed by a batch normalization layer (not shown) and max pooling layer
- Passes through a series of four residual blocks
- Passes through an average pooling layer before being transformed into a vector
- Passes through a final linear layer with 27 outputs, one for each class



Experimental Set-up and Evaluation

Used 8 GPU instance

Reduced amount of training and testing data by creating random sets

- Training: 10,000 videos
- Testing: 3,284 videos

Parameter Evaluation

- Batch size
- Learning Rate
- Dropout Layers
- Number of kernels
- Size and shape of max and average pooling layers

- Number and types of experiments limited by available time and computation power
- Training and testing accuracies examined and compared to detect overfitting
- Training accuracy much higher than test set accuracy considered evidence of overfitting
- Test accuracy that decreased from one epoch to the next was a sign of overfitting

Results

- Network was trained and tested using six different batch sizes, with smaller batch sizes producing better test set accuracy.

Batch Size	Test Set Accuracy
10	9.99%
25	54.25%
50	59.10%
100	52.31%
150	53.59%
300	48.32%

- Several different learning rates were used. Learning rates that were too low overfit, and too high underfit.

Learning Rate	Test Set Accuracy
0.0001	41.83%
0.001	52.31%
0.005	47.90%
0.01	48.30%
0.025	20.77%

Results Cont.

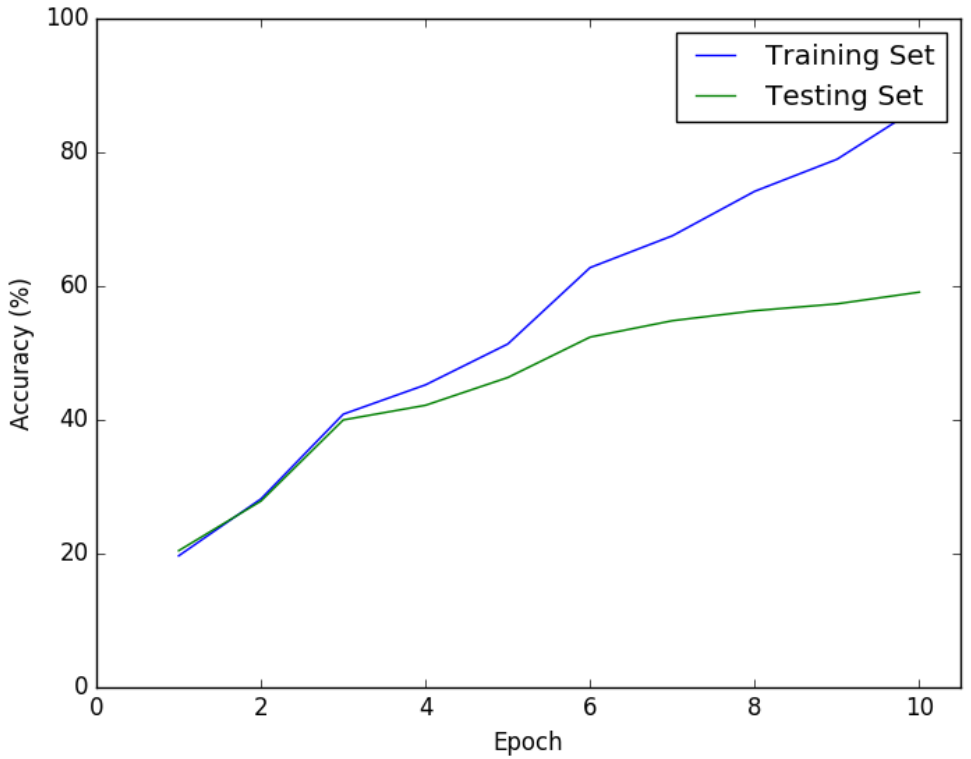
- Inclusion of added residual input at end of each 3D ResNet block does aid learning.
- Max pooling layer at end of initial convolution and average pooling layer at tail of last residual block yields best test set accuracy.
- Use of Dropout layer before final linear layer did not improve test set accuracy.

Block Architecture	Test Set Accuracy	Notes
With Added Input	52.31%	Test accuracy continued to increase up to 10th epoch
Without Added Input	16.47%	Training was stopped after test accuracy decreased between second and third epochs

Pooling Layer Types	Test Set Accuracy
Initial: max pool; Final: average pool	52.31%
Initial: average pool; Final: average pool	40.53%
Initial: max pool; Final: max pool	48.44%

Dropout Fraction	Test Set Accuracy
None	52.31%
25%	50.91%
50%	50.28%

The network that produced the highest test accuracy had a batch size of 50, with an Adam optimizer, a learning rate of 0.001, using an average pool layer with a 2x3x5 kernel and no dropout layer. This network achieved a test accuracy of 59.10% after 10 epochs.



The training and testing accuracy by epoch for the highest performing network

	Doing other things	Drumming Fingers	No gesture	Pulling Hand In	Pulling Two Fingers In	Pushing Hand Away	Pushing Two Fingers Away	Rolling Hand Backward	Rolling Hand Forward	Shaking Hand	Sliding Two Fingers Down	Sliding Two Fingers Left	Sliding Two Fingers Right	Sliding Two Fingers Up	Stop Sign	Swiping Down	Swiping Left	Swiping Right	Swiping Up	Thumb Down	Thumb Up	Turning Hand Clockwise	Turning Hand Counterclockwise	Zooming In With Full Hand	Zooming In With Two Fingers	Zooming Out With Full Hand	Zooming Out With Two Fingers
Doing other things	281	11	10	3	15	18	2	1	3	3	2	2	2	1	7	2	3	4	3	19	10	4	2	7	7	1	3
Drumming Fingers	7	123	0	0	3	0	0	1	2	3	0	0	0	0	0	1	1	0	0	3	1	0	1	1	3	1	1
No gesture	2	0	135	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	2	0	0
Pulling Hand In	3	1	0	64	24	8	0	1	1	1	2	0	1	0	3	6	0	0	13	2	0	2	0	1	1	2	1
Pulling Two Fingers In	6	1	0	8	104	9	0	1	1	2	0	0	1	2	1	3	1	0	9	0	1	0	1	0	0	3	1
Pushing Hand Away	7	0	0	5	2	121	0	0	0	2	0	0	0	0	2	0	0	0	0	1	2	0	0	1	0	0	0
Pushing Two Fingers Away	10	4	0	4	32	42	26	0	1	0	3	0	0	1	1	2	0	1	8	1	7	1	0	1	0	2	2
Rolling Hand Backward	1	6	0	1	2	0	0	43	71	0	1	0	0	0	0	5	0	0	1	1	0	0	0	1	0	0	0
Rolling Hand Forward	1	7	0	0	0	0	0	10	120	0	0	0	0	0	0	2	0	0	3	0	0	0	0	0	0	0	0
Shaking Hand	8	4	0	0	3	8	0	0	0	99	0	0	0	0	7	1	0	0	3	2	0	0	4	8	0	5	0
Sliding Two Fingers Down	3	1	0	2	6	3	2	1	2	0	95	0	0	6	3	30	0	0	3	1	6	0	0	0	1	0	1
Sliding Two Fingers Left	2	3	0	1	0	0	0	0	0	0	0	90	5	0	0	1	37	1	0	0	0	0	0	0	0	0	0
Sliding Two Fingers Right	6	1	0	1	2	2	0	0	0	0	0	1	106	0	0	0	4	35	0	0	1	2	2	0	1	0	0
Sliding Two Fingers Up	5	0	1	3	5	8	4	0	1	0	13	0	0	58	4	5	0	0	30	0	4	0	0	1	1	1	2
Stop Sign	10	1	2	0	6	27	0	0	0	6	0	0	1	0	104	2	1	0	0	1	4	1	0	2	4	2	0
Swiping Down	7	1	0	7	14	14	2	0	1	2	14	1	0	0	5	75	1	1	10	2	1	0	0	0	2	2	2
Swiping Left	2	3	0	0	2	4	0	0	0	1	0	19	1	0	0	1	116	2	0	0	0	0	1	0	0	0	0
Swiping Right	1	0	1	1	0	0	0	1	2	0	0	0	36	0	0	0	6	87	0	0	0	1	0	0	0	0	0
Swiping Up	1	4	0	9	7	15	1	0	2	1	5	0	1	9	1	8	1	0	92	1	1	0	0	2	0	1	0
Thumb Down	13	3	0	0	0	4	0	0	3	3	0	0	0	0	0	0	0	0	2	119	0	0	0	3	0	3	1
Thumb Up	16	1	0	0	24	15	8	0	0	1	3	0	0	2	14	0	0	0	4	0	62	0	0	2	3	1	1
Turning Hand Clockwise	12	12	1	3	3	6	1	0	0	6	0	1	2	0	0	0	0	1	3	0	0	34	18	3	1	2	0
Turning Hand Counterclockwise	7	7	0	2	5	5	1	0	0	7	0	1	0	0	1	0	2	0	2	0	0	12	48	2	0	4	1
Zooming In With Full Hand	3	8	1	2	5	8	2	0	0	4	0	0	0	0	8	0	0	1	0	0	1	10	3	83	18	7	2
Zooming In With Two Fingers	9	4	1	0	4	2	0	0	1	1	3	0	1	2	5	0	0	0	0	0	4	4	1	14	79	5	6
Zooming Out With Full Hand	4	8	0	0	3	3	1	0	0	6	2	0	0	0	2	2	0	0	0	2	0	1	2	8	1	83	25
Zooming Out With Two Fingers	10	6	0	0	5	1	0	0	0	3	2	0	0	0	1	1	1	1	0	1	7	1	2	2	17	12	86

This confusion matrix shows which classes were misclassified as other classes. Several interesting misclassifications are highlighted below in red. For example, “Pushing two fingers away” was misclassified as “Pushing hand away” more often than it was classified correctly.

Conclusions

- Value of included residual at the end of each residual block was one of the most important findings
  - Without it, network did not learn and seemed to encounter “degradation problem”
  - Evident when batch loss quickly plateaued and accuracy on testing set stagnated

- Batch sizes too large or too small decreased accuracy; 50 determined to be ideal
- Range of learning rates between 0.001 – 0.01 suitable; learning rates too low or too high resulted in extreme overfitting or inability to learn effectively
- Overall, most adjustments to the model and network architecture led to the discovery of overfitting rather than actual improvement on the test set.

Conclusions

- Performance was encouraging considering the limited time and computational resources
- Likely that the methods used in this project could be used to train a network that performs much better if more computational resources were available so that a deeper network could be trained with more epochs, using the entirety of the dataset