Visual Recognition using Machine Learning

Suraj Maniyar
Advisor: Dr. Paul Franzon
Dept. of Electrical and Computer Engineering
North Carolina State University

Original Paper

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 39, NO. 4, APRIL 2017

Long-Term Recurrent Convolutional Networks for Visual Recognition and Description

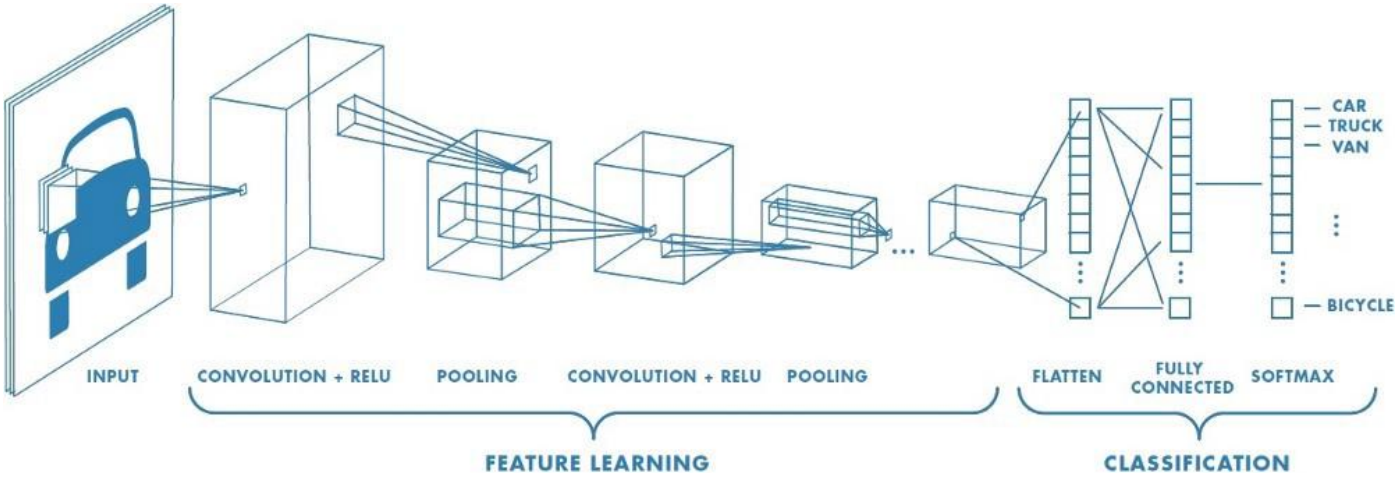
Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell

Abstract—Models based on deep convolutional networks have dominated recent image interpretation tasks; we investigate whether models which are also recurrent are effective for tasks involving sequences, visual and otherwise. We describe a class of recurrent convolutional architectures which is end-to-end trainable and suitable for large-scale visual understanding tasks, and demonstrate the value of these models for activity recognition, image captioning, and video description. In contrast to previous models which assume a fixed visual representation or perform simple temporal averaging for sequential processing, recurrent convolutional models are "doubly deep" in that they learn compositional representations in space and time. Learning long-term dependencies is possible when nonlinearities are incorporated into the network state updates. Differentiable recurrent models are appealing in that they can directly map variable-length inputs (e.g., videos) to variable-length outputs (e.g., natural language text) and can model complex temporal dynamics; yet they can be optimized with backpropagation. Our recurrent sequence models are directly connected to modern visual convolutional network models and can be jointly trained to learn temporal dynamics and convolutional perceptual representations. Our results show that such models have distinct advantages over state-of-the-art models for recognition or generation which are separately defined or optimized.

Key Takeaways

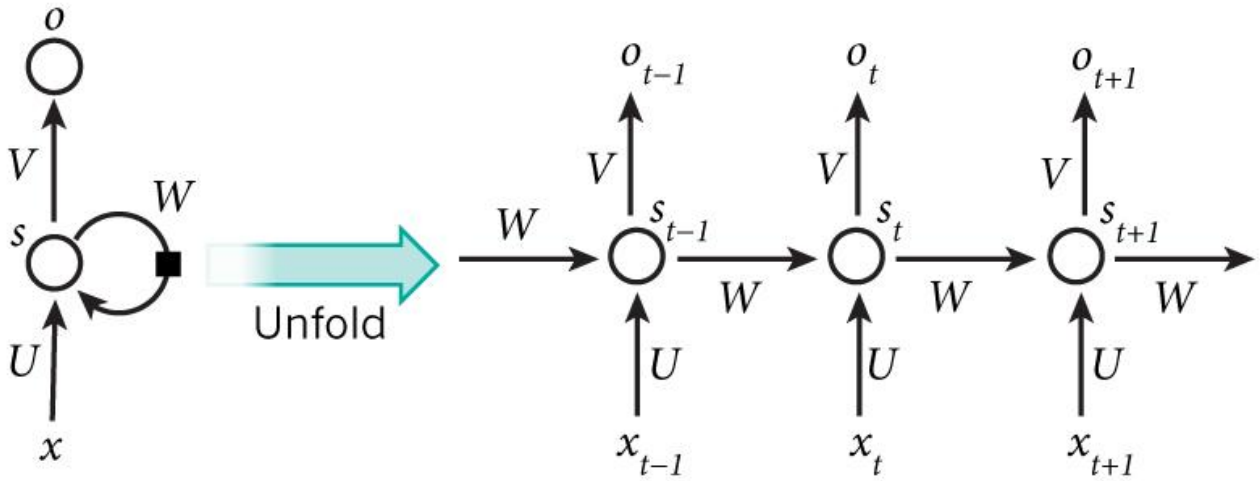
- Combines convolutional layers and long-range temporal recursion (*Long Short Term Memory*) in an end-to-end trainable format
- CNN provides feature transformation of image converting it into fixed-length vector representation which are passed into recurrent sequence learning module
- Feature transformation is time-invariant and independent at each time step making batch-processing and end-to-end optimization efficient

Convolutional Networks



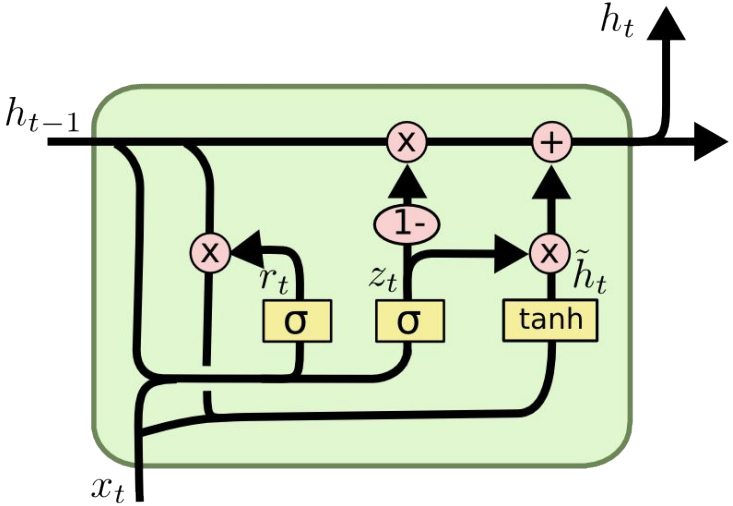
https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d

Recurrent Networks



http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

Long-Short Term Memory



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

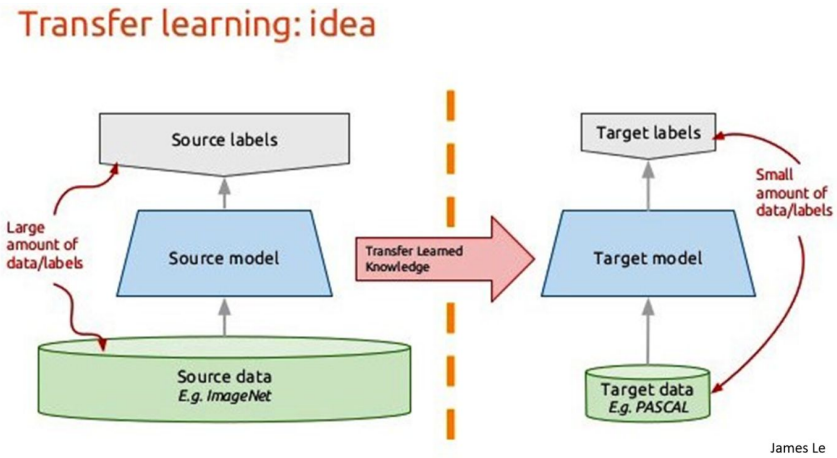
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

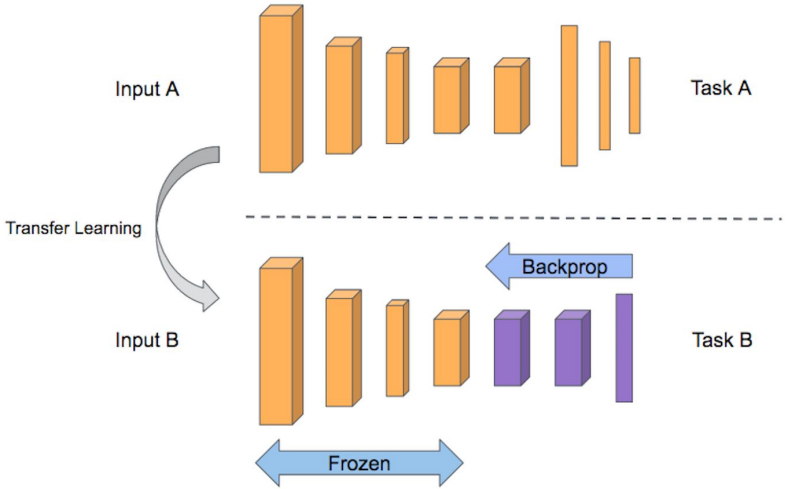
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Transfer Learning



https://www.datasciencecentral.com/profiles/blogs/transfer-learning-deep-learning-for-everyone

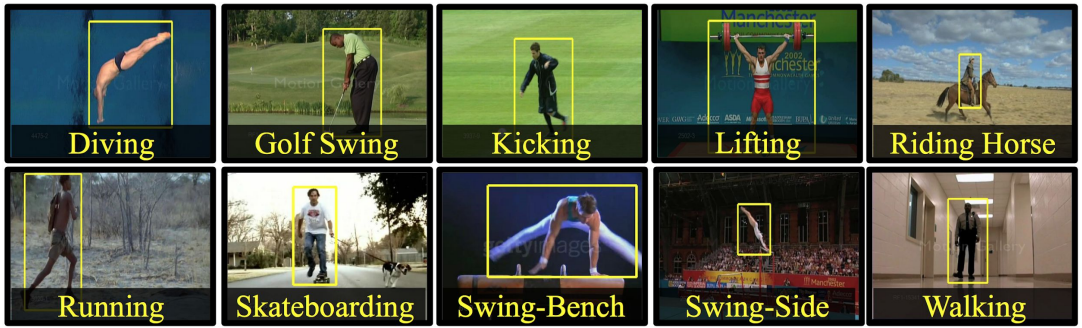


https://medium.com/@subodh.malgonde/transfer-learning-using-tensorflow-52a4f6bcde3e

Dataset

UCF-101

- Action Recognition dataset of realistic videos collected from YouTube
- Contains 13,320 videos from 101 action categories
- Images extracted from videos at 30 fps
- Restricted our implementation to 7 activities



http://crcv.ucf.edu/data/UCF_Sports_Action.php

Dataset

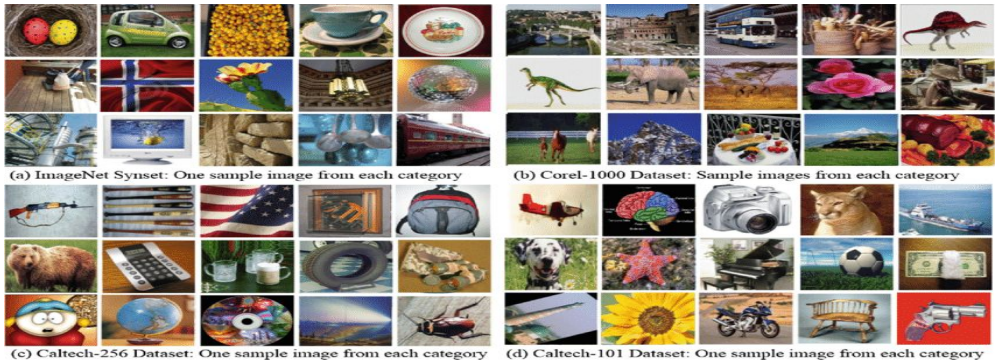
ImageNet

- Contains ~14 million URLs of images
- ~22,000 categories

(In Context of Deep Learning and Convolutional Neural Networks)

ImageNet Large Scale Visual Recognition Challenge (ILSVRC):

- ~1.2 million training images
- ~1,000 categories



http://www.image-net.org/

Training Data

- Videos taken from 7 activities (3-Similar, 4-Different)
  - Apply Eye Makeup, Apply Lipstick, Brushing Teeth (Similar Activities)
  - Basketball, Diving, Nunchucks, Punching (Different Activities)

- RGB images resized to 224 x 224

- Frames extracted at 30 fps

Training Data

	Videos/Activity	Frames/Video	Frames/Activity	Total Frames
Train Data	8	500	4,000	28,000
Cross Validation Data	10	100	1,000	7,000
Test Data	10	100	1,000	7,000

Training Data

Classification Model

Train Data:

X = (29566, 224, 224, 3)

Y = (29566, 7)

Cross-Validation Shape:

X = (7367, 224, 224, 3)

Y = (7367, 7)

Train Shape:

X = (7320, 224, 224, 3)

Y = (7320, 7)

Sequential Model

Train Data:

X = (14783, 16, 224, 224, 3) x 2

Y = (14783, 16, 7) x 2

Cross-Validation Shape:

X = (7351, 16, 224, 224, 3)

Y = (7351, 16, 7)

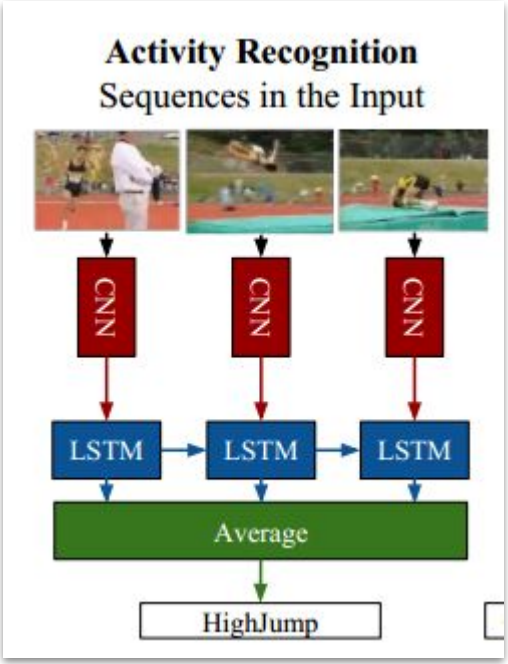
Train Shape:

X = (7304, 16, 224, 224, 3)

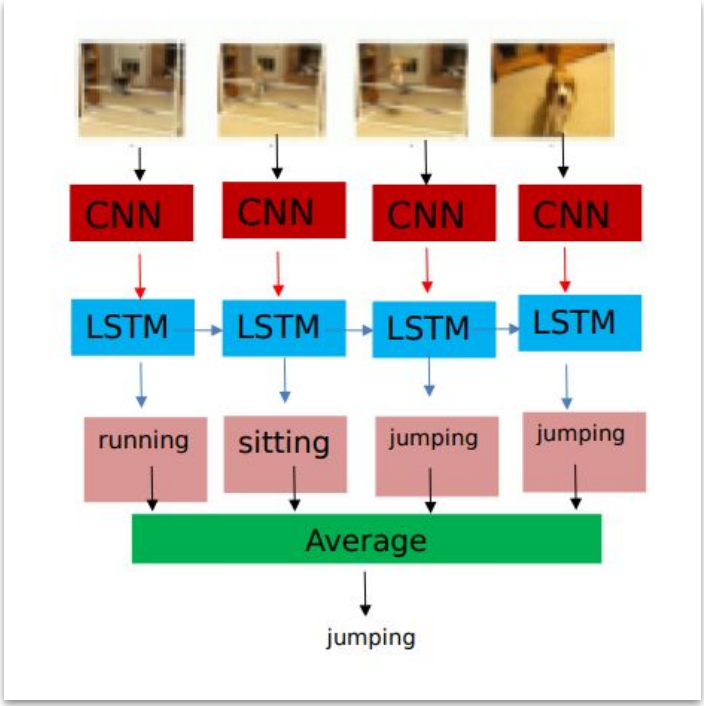
Y = (7304, 16, 7)

NOTE: Due to memory constraints on Henry Cluster, LSTM was trained in 2 sets with ~14,000 frames each

Model Architecture



Original Paper



https://pdfs.semanticscholar.org/4d7f/4c0ad34f7431e136ffeb9e344110ec6efaee.pdf

Implementation Details

- Language : Python3
- Framework : Keras with Tensorflow backend (Float precision : 32 bit)
- GPU : NVIDIA Tesla P100 x 2 (Henry2 Linux cluster)
- Memory Type : High Bandwidth Memory 2
- Total number of cores : 3584 x 2 = 7168
- GPU Memory : 16GB x 2 = 32GB
- Total Bandwidth : 732 GB/s x 2 = 1464 GB/s
- Max Power consumption : 250W x 2 = 500W
- RAM : 251GB (~4GB used)

Approach #1.1

Classification Model

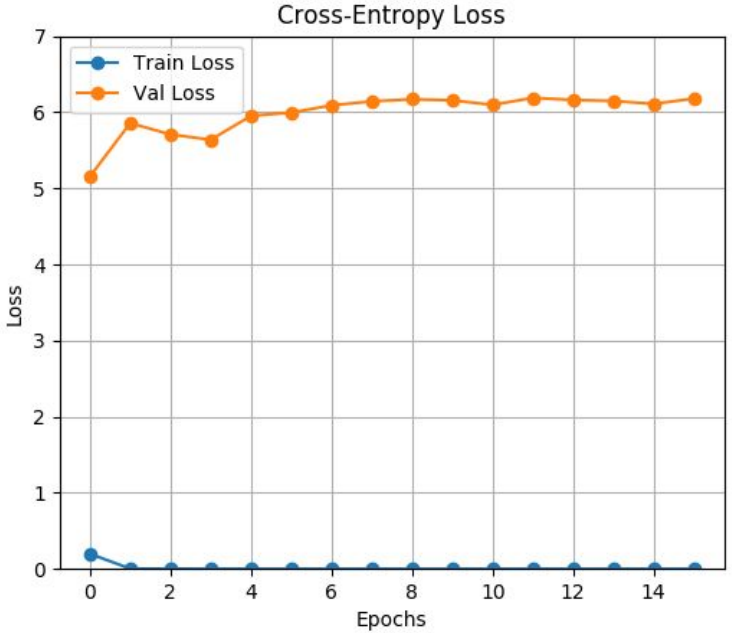
- 5 x Convolutional Layers
- 2 x Max Pooling Layers
- 2 x Batch Normalization Layers
- 2 x Fully connected Layers (1024 & 512)
- Activation : ReLU
- Classification : Softmax
- Loss : Categorical Cross Entropy
- Optimizer : Adam
- Total Parameters : 48,326,343

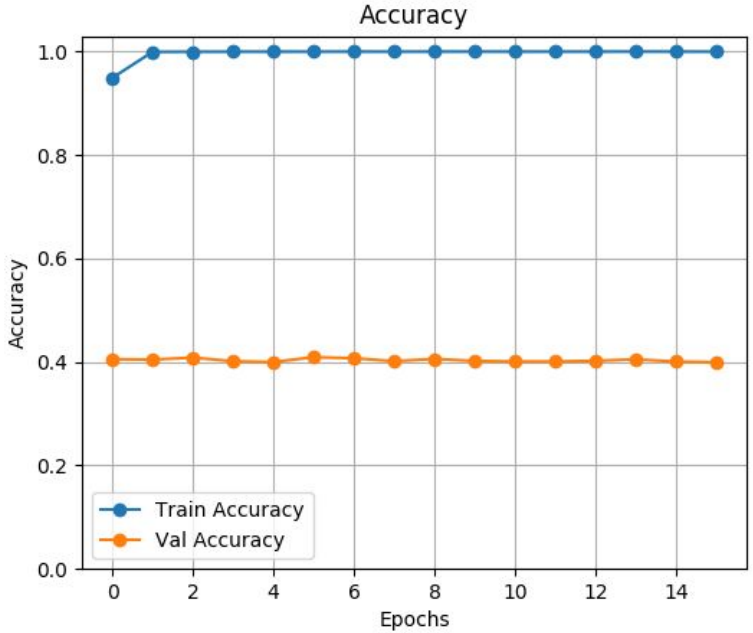
Hyper-parameters

- Regularization : Dropout (Probability : 0.5)
- Batch Size : 128
- Learning Rate : 1e-4
- Decay Rate : 1e-2

Weights initialized using normal distribution.

Results





Train Loss	0.0004
Validation Loss	**6.1779**
Test Loss	**5.4833**

Train Accuracy	99.98%
Validation Accuracy	**39.94%**
Test Accuracy	**44.84%**

Approach #1.2

Sequential Model

- 5 x Convolutional Layers
- 2 x Max Pooling Layers
- 2 x Batch Normalization Layers
- 2 x Fully connected Layers (1024 & 512)
- 2 x LSTM Layers (64 & 64)
- Activation : ReLU
- Classification : Softmax
- Loss : Categorical Cross Entropy
- Optimizer : Adam
- Total Parameters : 48,326,343

Hyper-parameters

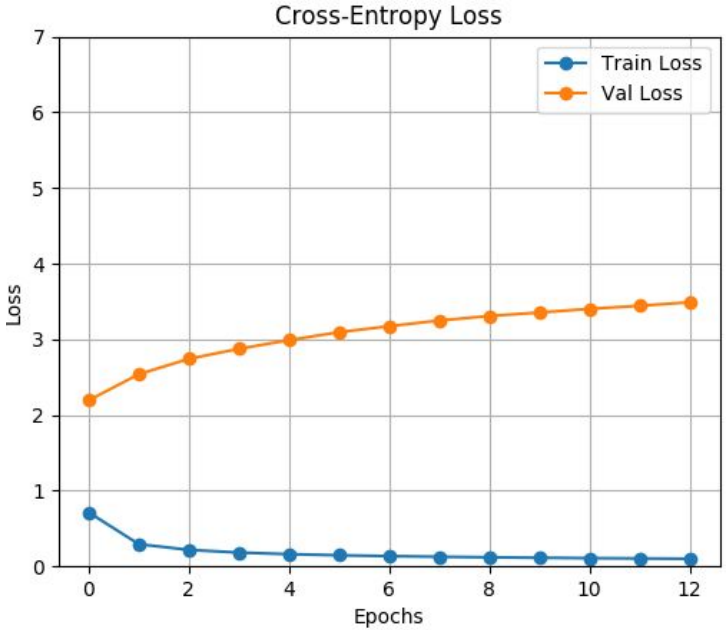
- Sequence Length : 16
- Regularization : Dropout (Probability : 0.5)
- Batch Size : 64
- Learning Rate : 1e-4
- Decay Rate : 1e-2

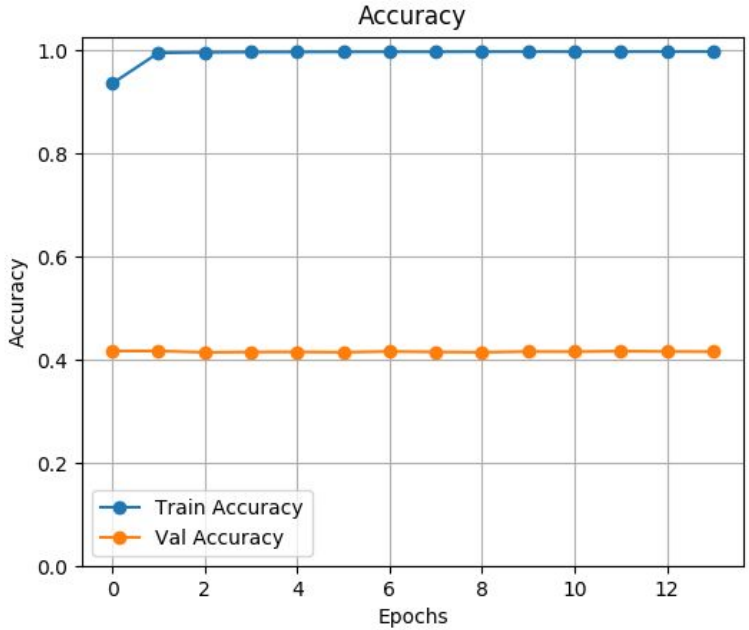
- ConvNet weights initialized from classification model : Non-Trainable
- Fully-Connected and LSTM Layers : Trainable

Analysis

- Total number of trainable parameters : 48,326,343
- Keras uses 32 bit floating point in backend
- Total Size = 48,326,343 x 32 = 184.3 MB

Results





Train Loss	0.0999
Validation Loss	**3.4890**
Test Loss	**2.0164**

Train Accuracy	99.82%
Validation Accuracy	**41.62%**
Test Accuracy	**44.71%**

Shortcomings

- Due to memory constraints on Henry Cluster, batch size is reduced to **64**
- Training accuracy is very high compared to test accuracy : Model overfits the data
- Network weights are randomly initialized

Shortcomings

- Original paper uses the same network pre-trained on ImageNet dataset (47 GB)
- On a system equipped with 4 NVIDIA Titan Black GPUs, training a single net took 2–3 weeks depending on the architecture*

* K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition in ICLR, 2015

Approach #2.1

- Transfer Learning from Network trained on ImageNet dataset
- Very Deep Convolutional Networks for Large Scale Visual Recognition (Visual Geometry Group, Department of Engineering Science, University of Oxford)
- VGG-16 : Uses only 3x3 convolutional layers stacked on top of each other in increasing depth (Maximum depth : 16)

Approach #2.1

https://www.quora.com/What-is-the-VGG-neural-network

Approach #2.1

Classification Model

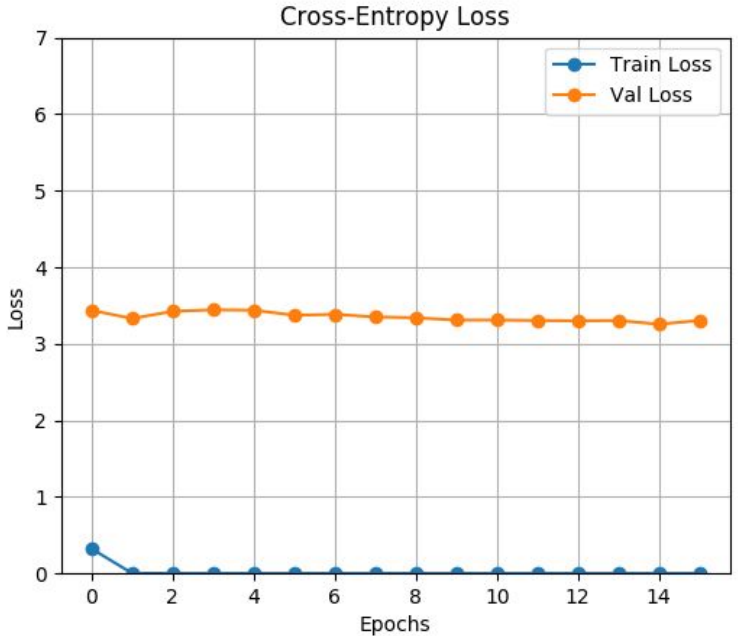
- Convolutional : VGG-16 Model
- 1 x Fully-Connected layer (256)
- Activation : ReLU
- Classification : Softmax
- Loss : Categorical Cross Entropy
- Optimizer : Adam
- Total parameters : 6,424,583

Hyper-parameters

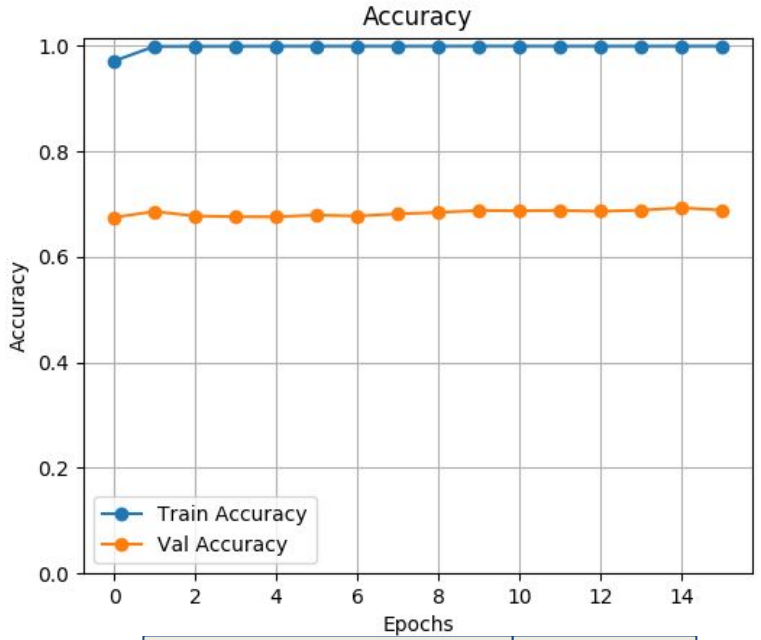
- Regularization : Dropout (Probability : 0.5)
- Batch Size : 128
- Learning Rate : 1e-4
- Decay Rate : 1e-2

- Convolutional part initialized from VGG-16 : Non-Trainable
- Fully-Connected Layer : Trainable

Results



Train Loss	0.0011
Validation Loss	**3.3039**
Test Loss	**4.0007**



Train Accuracy	99.98%
Validation Accuracy	**68.88%**
Test Accuracy	**65.71%**

Approach #2.2

Sequential Model

- Convolutional : VGG-16 Model
- 1 x Fully-Connected layer (256)
- 2 x LSTM layers (32 & 32)
- Activation : ReLU
- Classification : Softmax
- Loss : Categorical Cross Entropy
- Optimizer : Adam
- Total Trainable Parameters : 45,543

Hyper-parameters

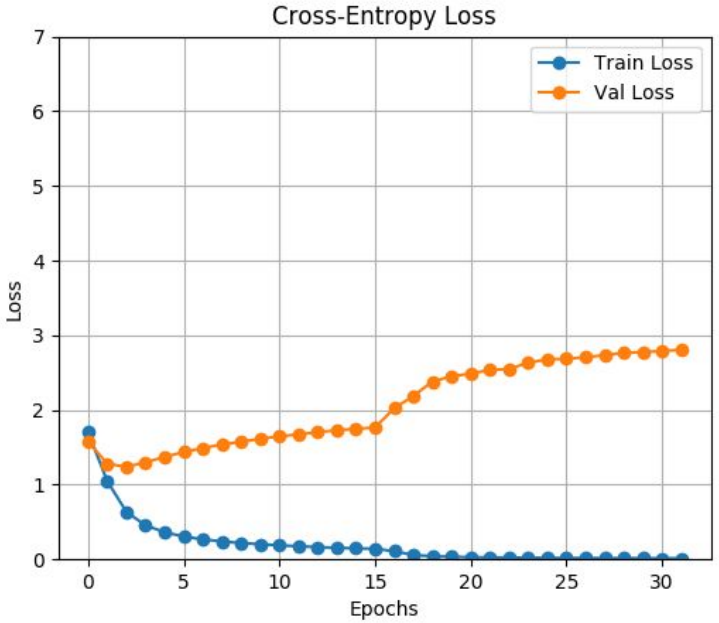
- Sequence Length : 8
- Regularization : Dropout (Probability : 0.5)
- Batch Size : 32
- Learning Rate : 1e-4
- Decay Rate : 1e-2

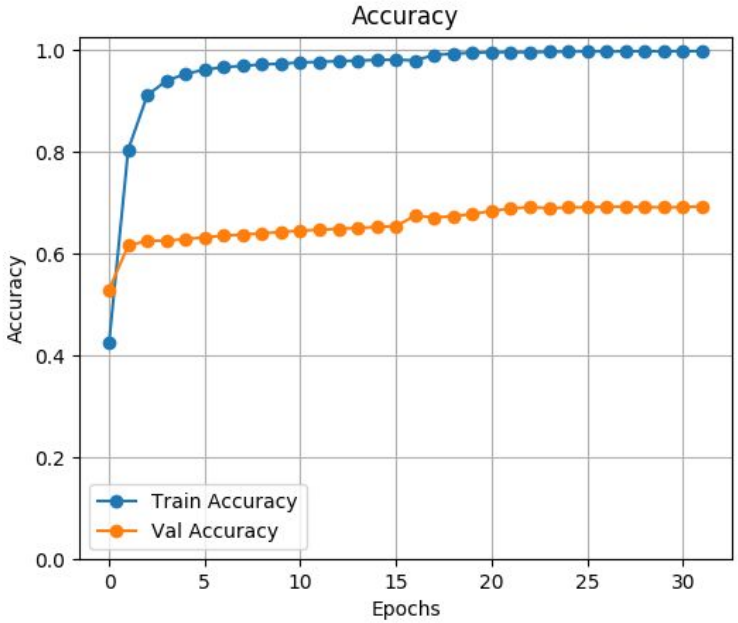
- Convolutional part and Fully-Connected layer initialized from classification model : Non-Trainable
- LSTM Layers : Trainable

Analysis

- Total number of trainable parameters : 45,543
- Keras uses 32 bit floating point in backend
- Total Size = 45,543 x 32 ~ 178 KB
- For end-to-end training:
  - Total number of trainable parameters : 21,183,015
  - Size = 21,183,015 x 32 ~ 80.80 MB

Results





Train Loss	0.0125
Validation Loss	**2.8068**
Test Loss	**4.1999**

Train Accuracy	99.73%
Validation Accuracy	**69.31%**
Test Accuracy	**70.03%**

Results from paper

	Classification Accuracy	LRCN Accuracy
Implementation (7 Activities)	65.71 %	70.03 %
Original Paper (101 Activities)	60.20 %	68.20 %

Questions?