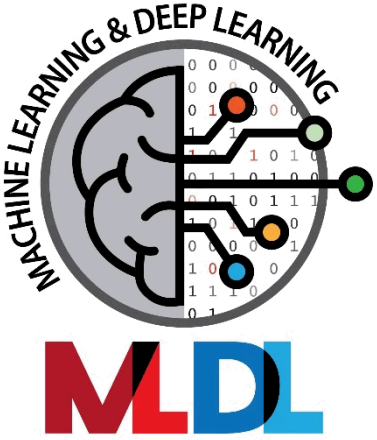
SAND2017-8453C



MLDL

Machine Learning and Deep Learning Conference 2017

LSTM-Based Video Action Recognizer

Daniel Barrett/6321

Funded through the DOE via Deep Learning Capability Stewardship Project
216 / 30.84.63.20.17.07
Unclassified Unlimited Release (UUR)

Outline

- Problem: Video Action Recognition
- Related Work
  - Non-Deep Learning
  - Deep Learning
- Network Architecture
- Results
  - Visualization
  - Quantitative
- Future Work

Problem Description

- Given:
  - a training set of videos
    - Videos are sequences of images of variable length

  - an action label for each video
    - E.g. **Wave**, jump, run, dig
  - (maybe) the location of the action in each frame of each training video

- Learn a model that
  - takes a new, unseen video as input

  - returns the action class(es) that are depicted in the video: **Wave**
  - (maybe) returns the location of the action in each frame

Different kinds of Action Classification/Recognition/Detection

- Different flavors of the problem:
  - Classify video into 1 of K classes
  - K separate binary present/absent judgements for each action class
  - Temporally segment a video and classify segments
  - Detect and track action-instances and return a list of labeled time intervals

- Many methods do not perform any tracking
  - Often average response over image(s)
    - Difficult to perform localization
    - Difficult to detect multiple simultaneous actions
    - Difficult to model interactions between multiple people and/or objects
    - Have a tendency to heavily use the background for classification
  - Tracking is difficult

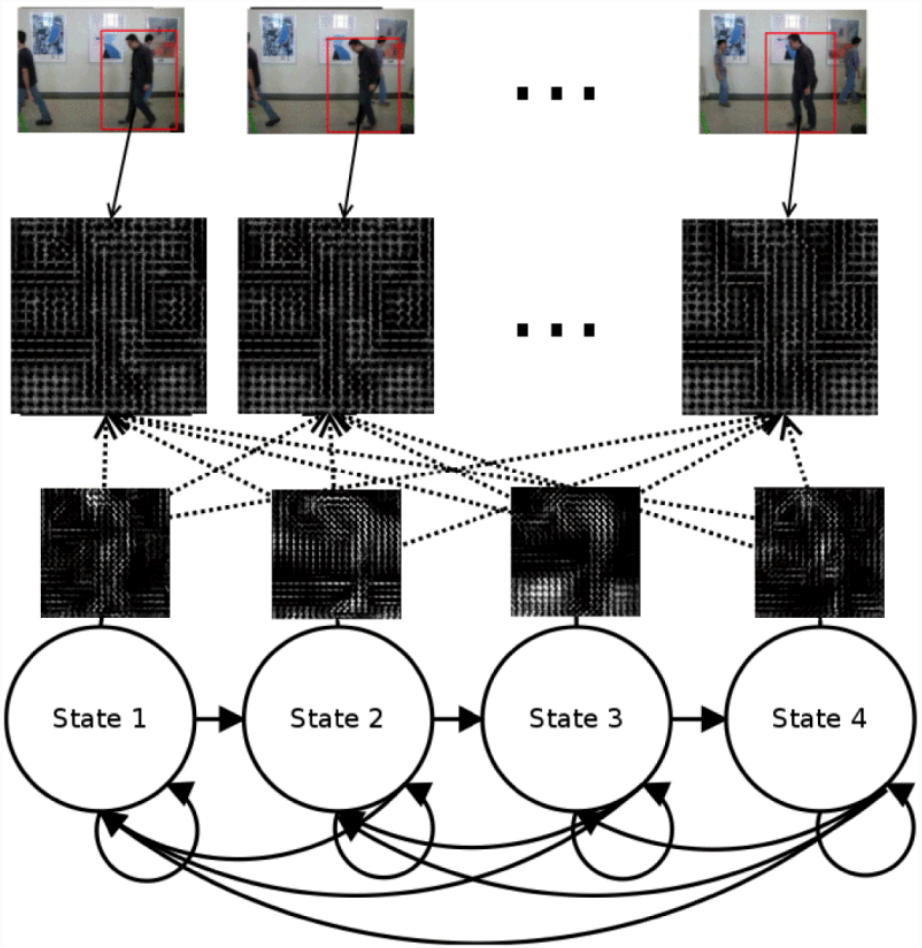
Related Prior Work (not Deep Learning)

- Action Recognition by Time Series of Retinotopic Appearance and Motion Features

  Daniel P. Barrett and Jeffrey. M. Siskind

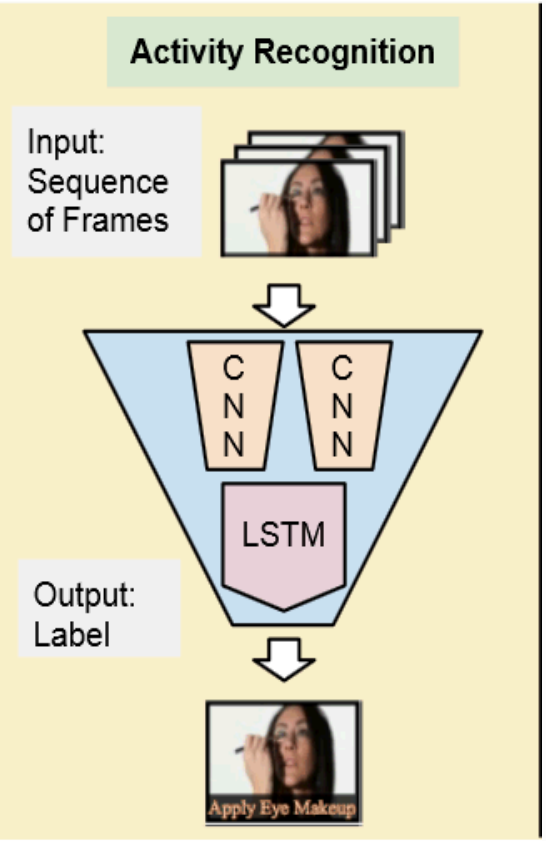
  IEEE TCSVT, Nov 2015

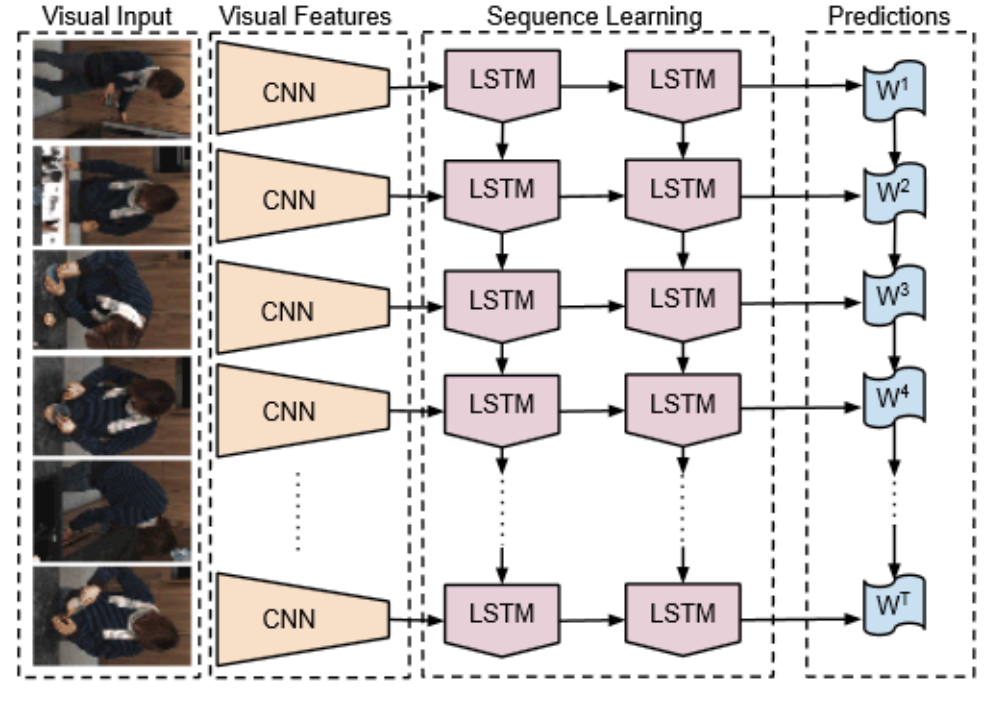
  - *Sequence of action-centered frame models*
    - *HOG object detectors*
    - *HOF motion pattern detectors*
  - *Hidden Markov Model (HMM)*
  - *Simultaneous tracking and action recognition:*
    - *Finds the track that maximally depicts the given action using dynamic programming*
  - *Trained with gradient descent*



Related Work (Deep Learning)

- *Long-Term Recurrent Convolutional Networks for Visual Recognition and Description*
  - *Jeffrey Donahue et al, CVPR 2015*

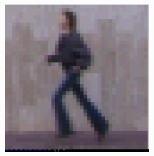




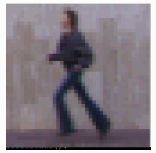
Implementation: Preprocessing

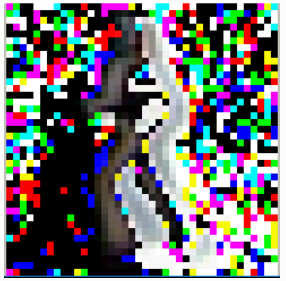
- Given bounding boxes around the person performing an action in a video

  - Extract cropped images from bounding boxes in each frame



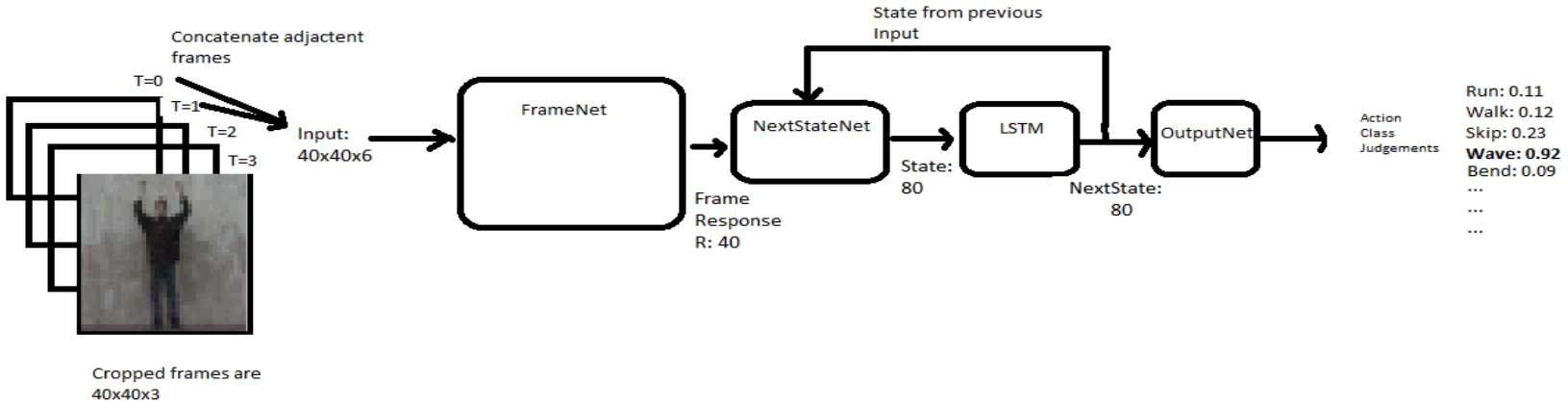
  - Add noise and jitter
  - Scale to 40x40
  - Extract additional image from adjacent frame with same noise and jitter
    - Give the network the opportunity to learn optical-flow-like features
  - Package the two 40x40x3 tensors into 40x40x6 tensor
  - Input consists of a sequence of 20 such pairs





Implementation: Network Architecture

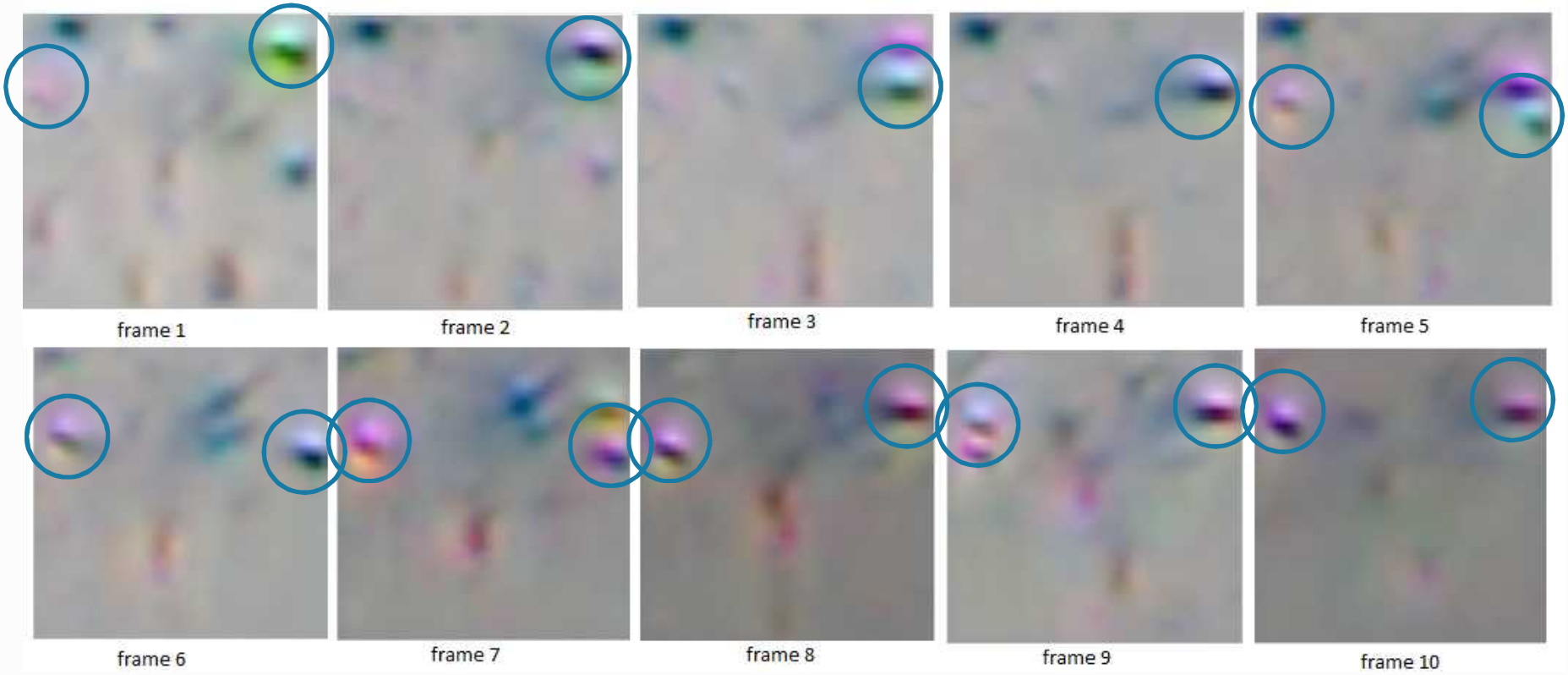
- Recurrent network run sequentially on each pair of frames in video
- Long Short-Term Memory (LSTM) block holds state information from previous frames



- FrameNet is deep convolutional ResNet
  - 12 ResNet Blocks, 3x3 convolutional filters
  - Pooling layer every 3 blocks
  - First 6 blocks have 16 filters, last 6 have 32 filters
  - Final layer is fully connected
- NextStateNet and OutputNet are single Resnet Blocks with fully-connected layers
- LSTM implementation from PyTorch used
- All layers use Batch Normalization and Relu Nonlinearity

Results -Visualization

- Optimize input to maximize the response of a particular neuron
  - Generate optimal sequence of 10 images to maximize “wave2” neuron



Results so far- Weizmann Dataset

- Weizmann Dataset (90 videos, 10 classes, 9-fold cross-validation)
- Accuracy of the network on the test sets: **83 %** (Barrett 2015 got **96%)**
- confusion matrix (rows for truth, columns for predictions)
  - ['bend', 'jack', 'jump', 'pjump', 'run', 'side', 'skip', 'walk', 'wave1', 'wave2']

```
[[ 18.   0.   0.   0.   0.   0.   0.   0.   0.   0.]
 [  0.  16.   0.   2.   0.   0.   0.   0.   0.   0.]
 [  0.   0.  15.   1.   0.   0.   2.   0.   0.   0.]
 [  0.   0.   3.  15.   0.   0.   0.   0.   0.   0.]
 [  0.   0.   0.   0.  17.   0.   1.   0.   0.   0.]
 [  2.   0.   2.   1.   0.  13.   0.   0.   0.   0.]
 [  0.   0.   2.   0.   1.   0.  15.   0.   0.   0.]
 [  0.   0.   2.   0.   1.   0.   0.  15.   0.   0.]
 [  1.   1.   2.   1.   1.   0.   0.   0.  11.   1.]
 [  0.   0.   0.   0.   0.   0.   0.   0.   3.  15.]]
```

Results So Far: UCF Sports Dataset

- UCF Sports: 150 videos, 11 classes, 9-fold cross-validation
- Accuracy of the network on the test sets: **52 %** (Barrett 2015 got **94%**)
- confusion matrix (rows for truth, columns for predictions)
  - ['dive', 'golf', 'kick', 'lift', 'ride', 'run', 'skate', 'swing-bench', 'swing', 'walk']

```
[[ 53.   0.  10.   0.   0.   0.   0.   7.   0.   0.]
 [  2.  26.  17.   0.   0.  17.  10.   0.   0.  18.]
 [  5.   0.  87.   0.   0.   0.   0.   0.   0.   8.]
 [  5.   0.   6.  18.   0.   0.   0.   1.   0.   0.]
 [  8.   0.   7.   1.  32.   2.   0.   0.   1.   9.]
 [ 13.   0.   7.   0.   0.  20.   4.   0.   3.  18.]
 [  8.   2.  13.   0.   4.  25.   3.   0.   1.   4.]
 [  5.   0.  10.   0.   0.   1.   0.  77.   6.   1.]
 [  0.   0.  10.   0.   0.   0.   0.   5.  50.   0.]
 [ 16.   4.  20.   1.  12.  22.   6.   0.   4.  25.]]
```

Potential Future Work

- Ways to improve "mediocre" results:
  - Most likely overfitting to small datasets
    - Can probably improve results with training tricks:
      - Use negative training data that depicts no actions
      - Mine dataset for the most difficult negative training data
    - Running on larger "harder" datasets may help
    - Use additional unlabeled data
      - Generative models, adversarial models, unsupervised pre-training
    - Explore architecture changes

- Implement tracking within deep learning framework
  - Adapt dynamic programming tracker from Barrett (2015)
  - Convolutional LSTM (Shi et al, NIPS 2015)
- Implement models that recognize interactions between multiple tracked objects or people