Adrian Marinovich
Springboard Data Science Career Track
Capstone Project #2 - Milestone Report 1
February 14, 2019

Project: Classification of human actions in videos

Problem statement: Why it's a useful question to answer and details about the client:

What is the problem you want to solve?
The problem is to classify human actions as seen in videos.

Who is your client and why do they care about this problem?
A range of clients may be interested in human action classification in videos. Potential applications include the development of instructional tools for robots, accident prevention in industrial sites, and monitoring of secure locations. Such a tool might also allow robots and self-driving cars to better predict the intentions and possible outcomes of human actions and thus allow safer response decisions.

Description of the dataset, how you obtained, cleaned, and wrangled it
The data are obtained from the UCF101 Action Recognition Data Set (http://crcv.ucf.edu/data/UCF101.php), which consists of 13320 videos labelled with 101 action categories. The videos were collected from YouTube, and consist of a diverse set of realistic examples in varied settings. There are five types of action categories: human-object interaction, body-motion only, human-human interaction, playing musical instruments, and sports. There are 25 unique groups of videos per action, with each group consisting of 4-7 video clips segmented from a larger video. The videos have a frame rate of 25, and range in length from 1.06 to 71.04 seconds, with a mean length of 7.21 seconds.

Python code was adapted from https://github.com/harvitronix/five-video-classification-methods to extract the UFC101 data into an appropriately created directory structure, and obtain frame-by-frame JPEG still image file sequences from each video using the FFmpeg multimedia framework. The resulting JPEG images are in 3 colors, with 240 x 320 pixel dimensions.

The data were split into train and test sets using pre-made lists prepared by the UFC research group, which were designed ensure that groups of video clips obtained from the same original video did not get split between sets. The train set comprised 9537 videos, and the test set comprised 3783 videos. As the data contain arbitrary action categories without immediate use-case application, and are instead intended for development of models for later training on 'real-world' datasets with more relevant classifications, no hold-out set was considered necessary.

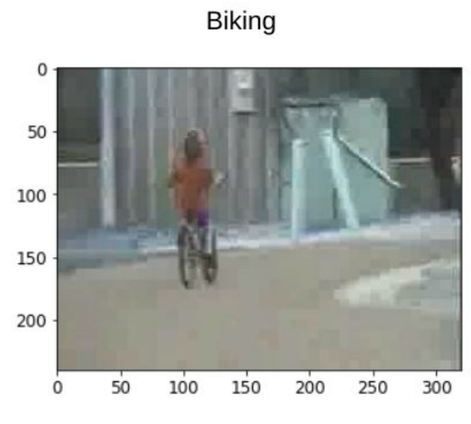
Video classification was performed using a variety of both Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) recurrent neural network (RNN) models that were fed features extracted from the image sequences using a pretrained InceptionV3 model. The image data, originally with 240 x 320 x 3 shape, were reshaped to 299 x 299 x 3 shape to fit the required input dimensions of the InceptionV3 convolutional neural network (CNN) model. The feature extraction stage produced arrays with a 2048 feature dimension and additional dimension according to sequence length. The LSTM and GRU models require a fixed sequence length. The code referenced above approached this constraint by setting 40 frames as the lower limit, and subsampling down all videos to this limit, with a top limit for subsampled videos of 300 frames, resulting in exclusion of 1,306 videos. To allow use of all videos, and to examine a range of fixed sequence lengths for their impact on model accuracy, the code was further modified to replace the standard 40-frame subsampled sequence length with a variable sequence length, of 50, 80 and 100 frames. This was achieved by subsampling longer sequences as before by regular skipping of frames, and additionally by interpolating shorter sequences using repetition of frames at regular intervals.

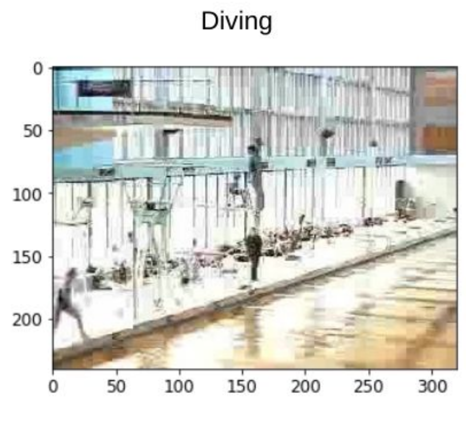
Machine learning was performed using Python, primarily in Keras with a TensorFlow backend, on an NVIDIA GTX 1060 GPU with 6 GB memory.

Initial findings from exploratory analysis - Summary of findings:
Overall, the LSTM and GRU models gave top-1 validation accuracies between 64% and 75%. There was worse performance by the LSTM model, although GPU memory constraints limited development of LSTM models. The GRU models were used for building deeper models, and further hyperparameter tuning was performed using these models.

The highest top-1 validation accuracy of 75% currently is attained using a model with 3 GRU layers and a dense layer, using an 50-frame sequence length and batch size of 8. While tuning of batch sizes using the 50-frame sequence reveals improvement in validation accuracy with smaller batch sizes, this tuning has yet to be completed in larger sequence length models.

Visuals and statistics to support findings
Example images from videos are shown here:





The model descriptions and hyperparameter tuning table are shown below.

Model descriptions:
('seq' = sequence length)

GRU-3 - *top performer*:

```
Layer (type)            Output Shape          Param #
=================================================================
gru_1 (GRU)             (None, seq, 2048)     25171968
_________________________________________________________________
gru_2 (GRU)             (None, seq, 2048)     25171968
_________________________________________________________________
gru_3 (GRU)             (None, 2048)          25171968
_________________________________________________________________
dense_1 (Dense)                (None, 512)             1049088
_________________________________________________________________
dropout_1 (Dropout)            (None, 512)             0
_________________________________________________________________
dense_2 (Dense)                (None, 101)             51813
=================================================================
Total params: 76,616,805
```

GRU-2:

```
Layer (type)            Output Shape          Param #
=================================================================
gru_1 (GRU)             (None, seq, 2048)     25171968
_________________________________________________________________
gru_2 (GRU)             (None, 4096)          75509760
_________________________________________________________________
dense_1 (Dense)                (None, 512)             2097664
_________________________________________________________________
dropout_1 (Dropout)            (None, 512)             0
_________________________________________________________________
dense_2 (Dense)                (None, 101)             51813
=================================================================
Total params: 102,831,205
```

Model descriptions (continued):

GRU-4:
```
Layer (type)              Output Shape            Param #
=================================================================
gru_1 (GRU)               (None, seq, 2048)       25171968
_________________________________________________________________
gru_2 (GRU)               (None, seq, 2048)       25171968
_________________________________________________________________
gru_3 (GRU)               (None, seq, 1024)       9440256
_________________________________________________________________
gru_4 (GRU)               (None, 1024)            6294528
_________________________________________________________________
dense_1 (Dense)                   (None, 512)             524800
_________________________________________________________________
dropout_1 (Dropout)               (None, 512)             0
_________________________________________________________________
dense_2 (Dense)                   (None, 101)             51813
=================================================================
Total params: 66,655,333
```

LSTM-3:
```
Layer (type)              Output Shape            Param #
=================================================================
lstm_1 (LSTM)                     (None, seq, 2048)       33562624
_________________________________________________________________
lstm_2 (LSTM)                     (None, seq, 1024)       12587008
_________________________________________________________________
lstm_3 (LSTM)                     (None, 1024)            8392704
_________________________________________________________________
dense_1 (Dense)                   (None, 512)             524800
_________________________________________________________________
dropout_1 (Dropout)               (None, 512)             0
_________________________________________________________________
dense_2 (Dense)                   (None, 101)             51813
=================================================================
Total params: 55,118,949
```

Table 1. Hyperparameter tuning results.

Model	Sequence length (frames)	Batch size	Number of epochs until validation loss stopped improving	Validation accuracy	Validation loss
GRU-3	50	8	13	0.750	1.027
GRU-3	50	16	10	0.731	0.993
GRU-3	50	32	10	0.696	1.120
GRU-3	50	120	26	0.700	1.090
GRU-3	80	8	13	0.741	1.032
GRU-3	80	32	26	0.741	0.966
GRU-2	80	32	12	0.715	1.052
GRU-4	80	32	20	0.702	1.068
LSTM-3	80	32	14	0.637	1.359
GRU-3	100	32	17	0.709	1.109

Future directions:
Additional GRU modeling will determine if the smaller batch sizes also improve performance in models using larger sequence lengths.

Once tuning is complete, a machine-readable tuning table, and a tuning plot will be produced. In addition, feeding the top-performing model with variable sequence lengths from original video image sets will be attempted.

Due the limitations of the above approach, with fixed sequence lengths, single stream training, and lack of readily available visualization tools for LSTM/GRU models, another modeling approach may be considered, which implements two-stream temporal-spatial neural network models using videos of variable lengths within an end-to-end training framework that contains convolutional layers that can be visualized.

Project link:

https://github.com/adriatic13/springboard/tree/master/dsct_capstone2

References:

https://crcv.ucf.edu/data/UCF101.php

https://github.com/fchollet/deep-learning-with-python-notebooks

https://github.com/harvitronix/five-video-classification-methods