Multi-view fusion for activity recognition using deep neural networks

Rahul Kavi[a], **Vinod Kulathumani**[a, *], **FNU Rohit**[a], **Vlad Kecojevic**[b]
[a]Department of Computer Science and Electrical Engineering, West Virginia University, Morgantown, WV, 26506-6109
[b]Department of Mining Engineering, West Virginia University, Morgantown, WV, 26506-6109

Abstract. Convolutional neural networks (ConvNets) coupled with Long Short Term Memory (LSTM) networks have been recently shown to be effective for video classification as they combine the automatic feature extraction capabilities of a neural network with additional memory in the temporal domain. This paper shows how multi-view fusion can be applied to such a ConvNet LSTM architecture. Two different fusion techniques are presented. The system is first evaluated in the context of a driver activity recognition system using data collected in a multi-camera driving simulator. These results show significant improvement in accuracy with multi-view fusion and also show that deep learning performs better than a traditional approach using spatio-temporal features even without requiring any background subtraction. The system is also validated on another publicly available multi-view action recognition dataset that has 12 action classes and 8 camera views.

Keywords: recurrent neural networks, LSTM, driver distraction, multi-camera systems, camera network.

*Vinod Kulathumani, vinod.kulathumani@mail.wvu.edu

1 Introduction

Convolutional Neural Networks (ConvNets) are simple feed forward networks with automatic feature extraction capability using randomly initialized convolution filters. ConvNets have been applied for pattern recognition applications with great success in recent years.[1–7] They can be extended for classification in the temporal domain using recurrent neural networks. Long Short Term memory networks (LSTMs) are a particular type of recurrent neural networks that have become very popular in speech recognition, hand-writing recognition[8,9] because they can learn mappings from sequential inputs to single or sequential outputs. Noting that the combination of ConvNets with LSTMs can be used to classify data in a temporal domain, recent studies have applied this idea successfully for video classification.[10–13]

In this paper, we seek to extend the ConvNet LSTM architecture for multi-view camera systems. Multi-camera networks have been shown to improve the robustness of the system by provid-

ing complementary views of a scene.[14,15] In this paper, we present two fusion techniques that can be applied for video classification using ConvNet LSTMs. We evaluate our multi-view deep neural network in the context of a driver activity recognition system, which has several applications in vehicular safety. Our work is specifically motivated by a critical need for quantitative analysis of driver distraction in operators of surface mine equipment who are distracted by the presence of several safety gadgets inside haul trucks such as push button radios, navigation sensors, tire sensors and proximity warning systems.[16]

We carry out experiments in a driving simulator equipped with 3 cameras and classify actions into 7 different classes. We do not make any assumption regarding the length, start time and end time of each action sequence being known before hand. Since LSTMs inherently learn temporal relationships, the classifier can be directly applied to continuous video streams. We evaluate using a 10-fold cross validation and our results show significant improvement by utilizing multiple cameras. On the same dataset, we also implement a popular, more traditional approach using Histogram of Gradients on motion history images.[14,17,18] Our comparative study shows that the deep learning technique performs better in terms of recognition accuracy and more importantly, it is able to do so without requiring any background subtraction on the input video.

Next, we also validate our technique on a publicly available multi-view action recognition dataset. Specifically, we use the WVU dataset,[19] which is a comprehensive 12 action dataset that contains data from 8 camera views. Data from all views is synchronized in time. For each action, there are about 47 samples available from each view. We train our ConvNet LSTM on two-thirds of the samples and use the remaining for testing with a 3 fold cross validation. We compare the recognition accuracy with prior results that use spatio-temporal motion history image as features.[14,17] The ConvNet LSTM is found to be more tolerant to missing views and has a

significantly higher accuracy even with single or two views. Moreover, spatio-temporal techniques require reliable foreground extraction while the ConvNet LSTM is directly fed the input samples.

2 Related Work

Action recognition is a widely studied topic in computer vision with applications in several areas such as surveillance, gesture recognition and gaming.[20,21] Much of the traditional techniques for activity recognition have been based on supervised feature learning techniques that exploit the spatio-temporal characteristics of an action for classification. However, such techniques often rely on accurate foreground extraction techniques in order to be able to discern the characteristics of an activity being performed. Foreground extraction is challenging in applications such as driver activity analysis inside surface mines because of vehicle motion in uneven terrain and camera motion. Moreover, camera deployment is challenging inside vehicles and the view obtained of each driver may change because of changes in height and seat position. Therefore, in this paper we explore the use of automatic feature learning techniques using deep neural networks for activity recognition. Moreover, we do not make any assumption regarding the length, start time and end time of each action sequence being known before hand. Since LSTMs inherently learn temporal relationships, the classifier can be directly applied to continuous video streams.

Convolutional neural networks have been successfully applied for pattern recognition applications in recent years.[1–7] For classification of actions, there is a need to capture relationships in the temporal domain. One way to do so is by applying convolution filters that extend in time, with higher layers progressively learning features over larger intervals of time.[22] Another method to extend ConvNets into the temporal domain is using recurrent neural networks such as LSTMs.[8,9] The combination of ConvNets with LSTMs can thus be used to classify spatio-temporal data.[10–13,23–25]

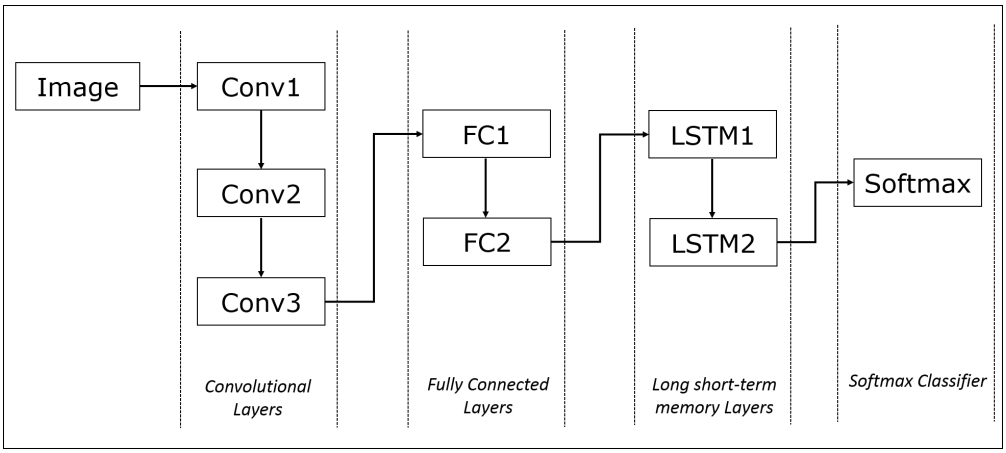

Fig 1 ConvNet LSTM architecture: The original image is downsized to 256 x 256 and is used as the input layer. This is followed by 3 convolutional neural network layers. The first layer consists of 20 kernels of size 5 x 5. The second layer consists of 50 kernels of size 5 x 5. The third layer consists of 50 kernels of size 4 x 4. These operations result in 50 channels of 29 x 29 resolution which are fed to two dense, fully connected layers. Finally, two LSTM layers are used to discover long-range temporal relationships.

This is the idea we seek to exploit in this paper for activity recognition, augmented with a multi-view camera network. Specifically, we show how such deep neural networks can be used to easily fuse features from multiple views and apply it for the problem of driver activity analysis. 3D convolutional neural networks[26] were not explored in this work as they are computationally expensive to train as compared to a 2D convolutional neural network.

3 System architecture

In this section, we describe our multi-view ConvNet LSTM architecture. In Section 3.1, we first describe the components of our basic ConvNet LSTM architecture for processing data from a single camera. Then, in Section 3.2, we show how this architecture can be extended to combine data from multiple cameras.

Fig 2 Illustration of the convolution operations (using a few convolution filters) on image data

3.1 Building blocks of the deep neural network

As seen in Fig. 1, our ConvNet LSTM architecture consists of 3 convolutional neural network layers ($conv1$, $conv2$ and $conv3$), 2 fully connected layer blocks ($FC1$ and $FC2$), and 2 LSTM layer blocks. The design and functionality of each of the blocks are described below.

The original image is a grayscale image of size 640 x 480 pixels. This is resized to 256 x 256 and passed as an input to the first convolutional layer ($conv1$). The first convolutional layer consists of 20@5 x 5 kernels. All the convolutions are applied with a stride of 1. Note that a convolution operation with a kernel of size c x c (and stride 1) on an image of size n x n gives rise to an image that is of size m x m, where m, n and c are related as $m = n - c + 1$.

Thus, the convolution operation at $conv1$ gives 20 channels of reduced sized images, each with a size 252 x 252. A $2D$ max pooling operation is then performed on the output from $conv1$, which reduces the size of the input to 126 x 126. Fig. 2 illustrates the convolution operation on a sample input image using 5 out of the 20 convolution filters and highlights how the convolutional neural network blocks act as automatic visual feature extractors.

The output from the first ConvNet block is passed through convolutional layer $conv2$, consist-

ing of $50@5x5$ kernels (with a stride of 1), followed by a second pooling layer. The 50 channels of output from $conv2$ are passed to a third convolutional layer $conv3$ and convolution operation is performed with $50@4x4$ kernels (with a stride of 1). The output from this layer is passed to a third pooling layer which gives 50 output channels each of size 29 x 29. The output from $conv3$ is connected to the first fully connected layer (FC1) of size 1000. The output from $FC1$ is connected to the second fully connected layer ($FC2$) of size 500. The feature vector representation at $FC2$ is passed to the subsequent LSTM layers, each with a hidden layer size of size 512.

The goal of the LSTM layers is to use information between multiple successive frames in making the classification. LSTMs use memory cells to discover long-range temporal relationships. We specifically use LSTMs with peephole connections as shown in.[11] Let $x = (x_1, x_2, ...x_T)$ denote the input sequence, $h = (h_1, h_2, ...h_T)$ denote the hidden vector sequence and $y = (y_1, y_2, ...y_T)$ denote the output sequence. The hidden layer of the LSTM is computed using the following set of equations. At each time t, let i_t, f_t, o_t and c_t denote the input gate, forget gate, output gate and the cell state respectively. W is used to denote the respective weight matrices. b terms denote the bias vectors. For example, b_o is the output bias vector. $\otimes$ denotes the element-wise product.

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \tag{2}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \tag{4}$$

$$h_t = tanh(c_t) \otimes o_t \tag{5}$$

The output is computed using the following equation.

$$y_t = W_{ho} \otimes h_t + b_o \tag{6}$$

In our system, we have obtained results with both 1 and 2 layer LSTMs and compared their performance. Each LSTM layer is constructed with 512 memory cells. The output of the LSTM layers is a 1 x 100 vector that is fed to a Softmax classifier. The Softmax classifier outputs a vector of size 1 x K where K is the number of actions to be classified. This 1 x K output represents the probability of each action class at every time t.

The output from the Softmax layer can be used in a number of ways: (i) the output class can be predicted independently for each image frame as the maximum of probabilities for each class or (ii) a history of Softmax output over the last several frames can be used to group a set of frames into a particular action class. For computing the system accuracy in this paper, we have used the former approach, i.e., the output label for each image frame is the action with the maximum probability as predicted by the Softmax classifier. Thus, we use the output predicted at each time frame by the Softmax classifier to match it with the labeled data and then compute the accuracy. We note that by using suitable outlier detection strategies and by using temporal clustering of the output labels, we can further improve the performance of the action recognition system, but this analysis is beyond the scope of this paper.

3.2 Multi-view fusion using deep neural networks

In this section, we describe two strategies for fusing data from multiple cameras using deep neural networks. Let N_c denote the number of cameras. A block diagram of the two approaches is shown in Fig. 3.

3.2.1 Using ConvNet LSTMs as feature generators

In this technique, we feed the data from each camera into N_c different ConvNet LSTM blocks and thus the output from each view is a 1 x 100 vector per frame. The total dimension of the feature vector from the multi-view system is thus 1 x $N_c * 100$. A linear SVM classifier is then trained on these feature vectors to classify each action. Thus, in this technique, we use the LSTMs as feature generators for multi-view classification and fuse the feature vectors before classification.

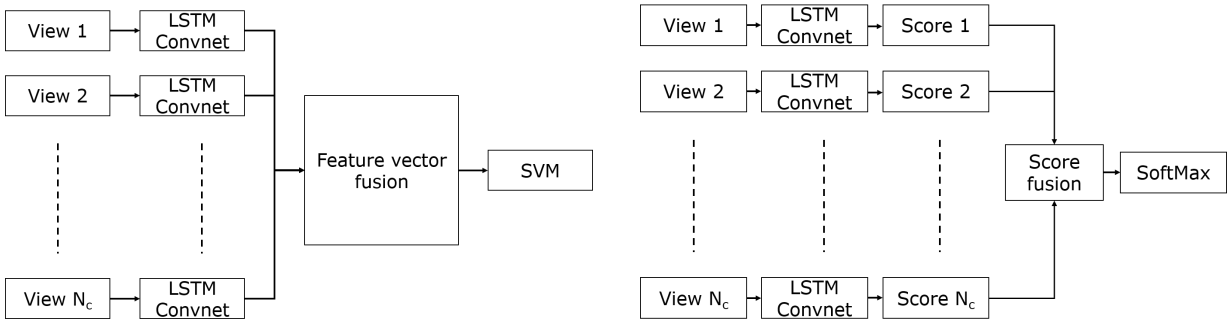

Fig 3 Two fusion architectures used for deep neural networks. (a) Feature vector fusion and (b) Score fusion

Note that due to minor differences in sampling rate and I/O processing rate, data frames may not be available from each camera at the exact same rate. For instance, if the sampling rate is 15 frames per second, over a given second one camera may only generate 14 frames while another may generate 15. To correct this, we only feed data into the multi-view classifier for those time instants when data from all N_c cameras is available. Thus, over any given time period T, the

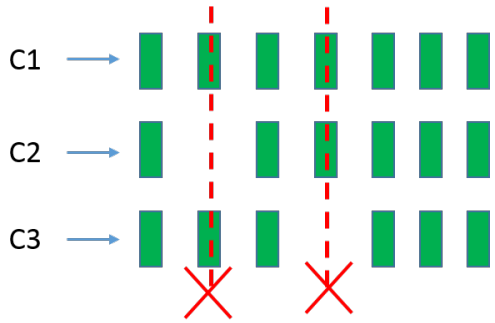

Fig 4 Due to frame rate inconsistencies across multiple cameras, we only generate outputs at time instants when images from all 3 cameras are available.

number of frames for which output is generated corresponds to the lowest frame processing rate in that time period. This is illustrated in Fig. 4.

3.2.2 Score fusion

In the score fusion approach, the output from the softmax layer from each view is treated independently and then simply added across the N_c views to determine the score distribution across different actions. Let $S_{a,v}$ determine the score for a given action a in view v. We then calculate fused score F_a for a given action s as described in Eqn. 7.

$$F_a = \frac{\sum S_{av}}{N_c} \tag{7}$$

The action a with the maximum score is labeled as the output class. We have experimented with both these fusion strategies and the results are described in the next section.

4 Experimental evaluation

4.1 Evaluation on data from a multi-camera driving simulator

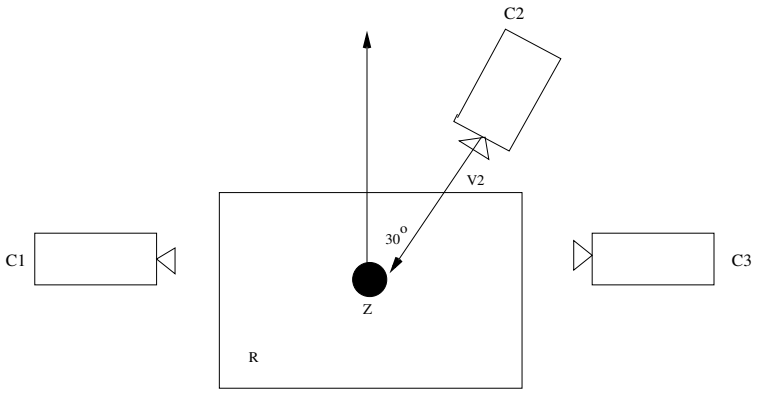

Fig 5 Schematics of the 3 camera network in a driving simulator. One of the cameras faces the driver at an angle of approximately 30 degree angle while the other two cameras are on the two sides.

Fig 6 Driving simulator used for data collection. The three monitors present the road view to the driver. The tablet represents the console interface operated by the driver. This setup emulates the cabin inside haul trucks at surface mines.

To evaluate the performance of the multi-view deep neural network classifier, we applied it in the context of a driver activity recognition system where the goal is to identify a driver's action into one of 7 different classes shown in Table 1. The experiment was carried out using a driving simulator and a 3 camera system. A schematic of the camera layout is shown in Fig. 5. The driving simulator consists of three monitors that present the road view to the driver. The tablet represents the console interface operated by the driver. This setup emulates the cabin inside haul

Action	Label
Gears	S1
Driving	S2
Talking on Phone	S3
Picking up Phone	S4
Controls	S5
Looking Right	S6
Looking Left	S7

Table 1 Multi-view Simulator ConvNet LSTM action legend.

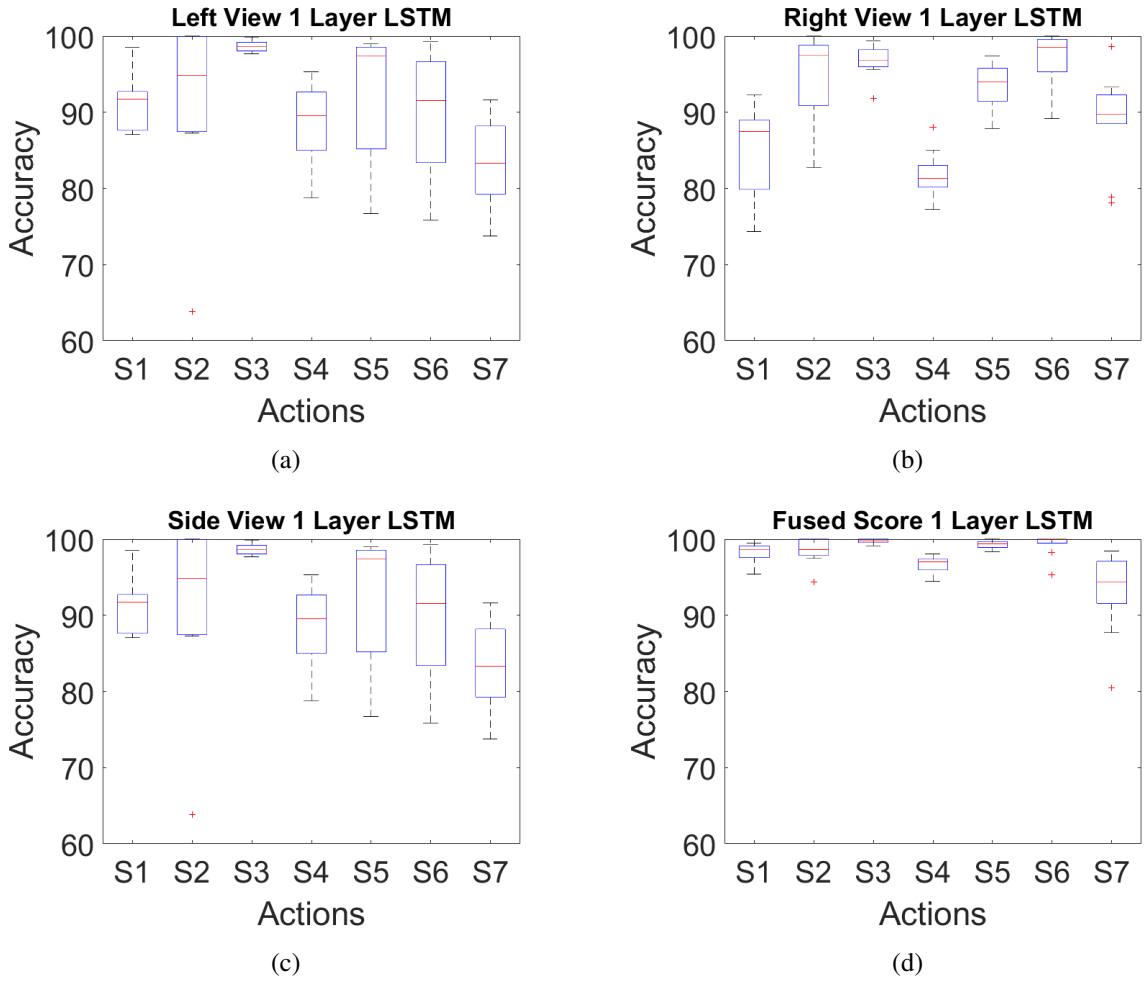

Fig 7 Comparison of classification accuracy across different views (a) Left view only (b) Right view only (c) Front Side view Only (d) With all three views and score fusion

trucks at surface mines. Data was collected on 3 different subjects, each performing the 7 actions. The dataset consists 30 minutes of video for each subject collected at 15 frames per second with a resolution of 640 x 480. A 10-fold cross validation was performed on the dataset (10 sets of data

each with 90% data for training and 10% data for testing).

The neural network was built using Theano[27](in Python). The network was trained on a Quadro K2200 GPU (with 4 GB memory). The learning rate was set to $\alpha = 10^5$. Dropout[28] was used to avoid over-fitting in the network. Smaller values of dropout probability are likely to not drop enough connections, thereby leading to overfitting and reliance on too few neurons for performance. Initially, we started with dropout value of 0.3 in convolutional layers and fully connected layers. Performance started to fall drastically with values higher than 0.6. So, it was set at 0.6 for our network in the convolutional layers and 0.5 in the fully connected layers. The input to the network was an image of size 256 x 256 with mean subtraction. The non-linear layers used were ReLU (Rectified Linear Unit) units. The batch size for mini-batch gradient size was set to 32 and RMSProp[29] was used to optimize the gradients.

Fig. 7 shows the classification accuracy when only data from a single view is utilized in the ConvNet LSTM classifier, and compares that with the case where data from all three views are used with score fusion strategy. We notice significant improvement in accuracy when combining data from all views. In Fig. 8(a) and Fig. 8(b), we compare the performance of feature vector fusion and score fusion respectively with a single layer LSTM. Both the fusion strategies perform equally well, although a slight improvement is observed with feature vector fusion when comparing the average and maximum error rates - this is illustrated more clearly in Table 2.

All of the above results are using only a single layer LSTM. Addition of a second LSTM layer does not have significant impact as seen in Fig. 8(c) and Fig. 8(d), which shows the classification accuracy using 2 layer LSTMs. The combined average and maximum error rates across all actions are listed in Table 2 when single and two layer LSTMs are used. We observe that with score fusion strategy, there is a slight decrease in error rates when 2 layer LSTMs are used. But in the feature

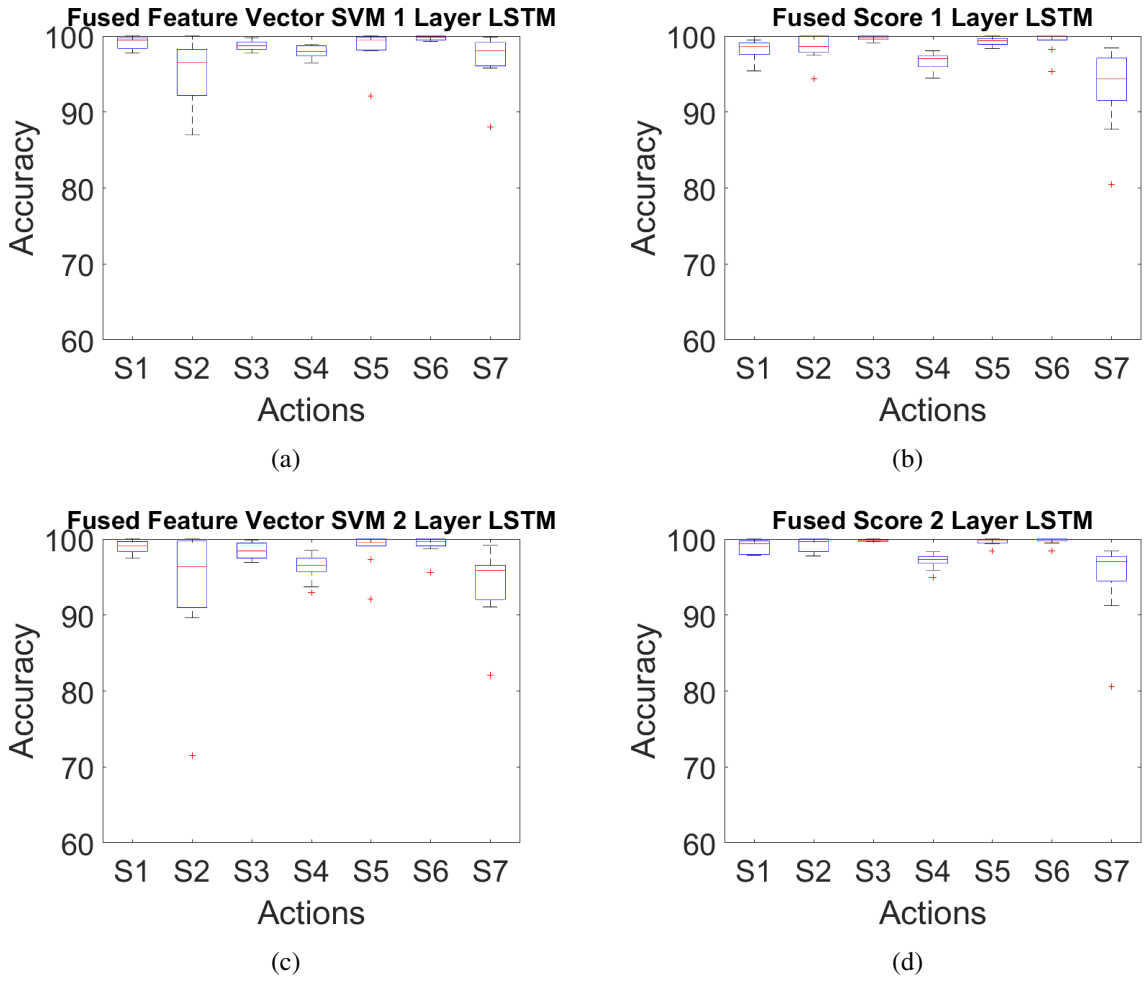

Fig 8 Classification accuracy with multi-view fusion: (a) One layer LSTM and feature vector fusion (b) One layer LSTM and score fusion (c) Two layer LSTM and feature vector fusion (d) Two layer LSTM and score fusion

vector fusion strategy there is actually a slight increase in error rate. We observe from these results that multi-view fusion has a far more significant impact compared to additional LSTM layer.

4.1.1 Comparison with spatio-temporal motion history feature technique

Next, we also implement a more traditional technique based on spatio-temporal features to compare with the deep learning technique. Specifically, for each action sample, we first perform background subtraction on the images and then compute the motion history image[17] across each action video sample. A motion history image, computed across a sequence of images, captures the amount of motion at each pixel in the image. Histogram of Gradient features[30] are then obtained on this

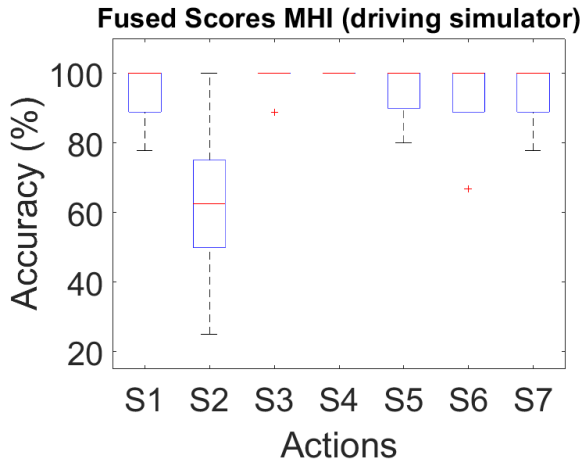

Fig 9 Recognition accuracy for individual actions using histogram of gradients on motion history images as feature vectors.

motion history image and are used as feature vectors for training and testing.[14] To obtain the HoG features, we have used 8 orientations, 8 x 8 pixels per cell and 24 x 24 cells per block. An SVM is used for classification per view and the score fusion approach described above is used to combine data from multiple views. Here also, we have used a 10 fold cross validation.

As seen in Fig. 9, recognition accuracy for actions such as $S2$, i.e., *driving*, is quite low. This is possibly because of the low spatio-temporal motion content associated with this action. As the number of action classes increase, we expect the spatio-temporal signatures to become more similar and thus increase the error rates. This is illustrated in Sec 4.2, where we compare on a dataset with 12 action classes. In Table 2, we show the maximum and average error rates using this spatio-temporal motion history feature technique. This table also shows that error rates with the deep learning approach is lower. Note that the maximum error rates are especially high with the spatio-temporal feature approach. But more importantly, we note that computing spatio-temporal features requires a good foreground extraction technique so that the motion history images contain relevant features from the human subject performing an action. This is relatively difficult in appli-

Classification Technique	Max Error Rate	Average Error Rates
Left View ConvNet LSTM (1 layer LSTM)	17.13	9.21
Right View ConvNet LSTM (1 layer LSTM)	18.08	8.90
Side View ConvNet LSTM (1 layer LSTM)	17.13	9.21
Fused Score ConvNet LSTM (1 layer LSTM)	**6.92**	**2.21**
Fused Features ConvNet LSTM (1 layer LSTM)	**4.75**	**1.92**
Left View ConvNet LSTM (2 layer LSTM)	13.01	5.77
Right View ConvNet LSTM (2 layer LSTM)	15.54	6.34
Side View ConvNet LSTM (2 layer LSTM)	13.625	7.54
Fused Score ConvNet LSTM (2 layer LSTM)	**5.23**	**1.51**
Fused Features ConvNet LSTM (2 layer LSTM)	**6.41**	**2.95**
Left view Motion history HOG	35	7.2
Right view Motion history HOG	35	6.9
Side view Motion history HOG	50	9.3
Fused Motion history HOG	**37.5**	**8.9**

Table 2 Comparison of error rates with 1 layer LSTM, 2 layer LSTM and spatio-temporal motion history image technique on data from 3 camera driving simulator

cations such as driver activity analysis, especially in rugged terrains such as surface mines, where the cameras are subject to large vibrations. The ConvNet LSTM approach does not require this step. The input videos are directly fed to the neural network after downsizing to the required 256 x 256 resolution.

4.2 Evaluation on WVU multi-view action recognition dataset

In this section, we describe the evaluation of the ConvNet LSTM approach on the publicly available WVU multi-view action recognition dataset[19] (cited in[31–34]). This dataset has data of subjects performing a set of 12 actions, captured from 8 cameras. The multi-camera network system in this dataset consists of 8 cameras that provide completely overlapping coverage of a rectangular region (about 50 x 50 feet) from 8 different viewing directions. The schematic of camera deployment is shown in Fig. 10. The data from all the cameras is synchronized in time. The dataset is provided in

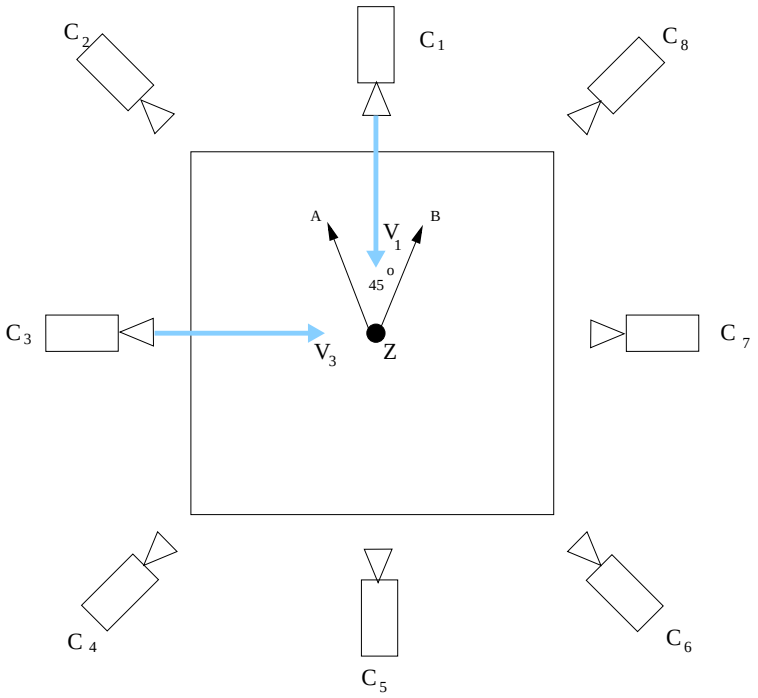

Fig 10 Schematic of WVU multi-view action recognition dataset. The multi-camera network system consists of 8 cameras that provide completely overlapping coverage of a rectangular region (about 50 x 50 feet) from 8 different viewing directions. The subject is at the center and performs actions facing camera C1.

3 parts for each action.[19] We have used data from parts 1 and 2 in which the subject is always at the center of the rectangular region. This ensures that the resolution of the subject performing actions is consistent across all images and there is no need to scale and normalize the images. There are 47 action samples per action from each view. We divide this into a $2:1$ train-test ratio and use 3 fold cross validation. The data has a frame rate of 20 fps with 640 x 480 resolution.

We train the neural network with the same configuration as described in Section 4.1. We compute the recognition accuracy with different number of available camera views. The views to be retained are the ones with the lowest motion energy, i.e., the lowest information content. The most favorable views in terms of action content are removed.[14] Suppose that in Fig. 10, the subject performs the actions facing camera c_1. Then, we remove the views in the following order (C_1, C_5, C_3, C_7, C_2, C_8, C_4, C_6).

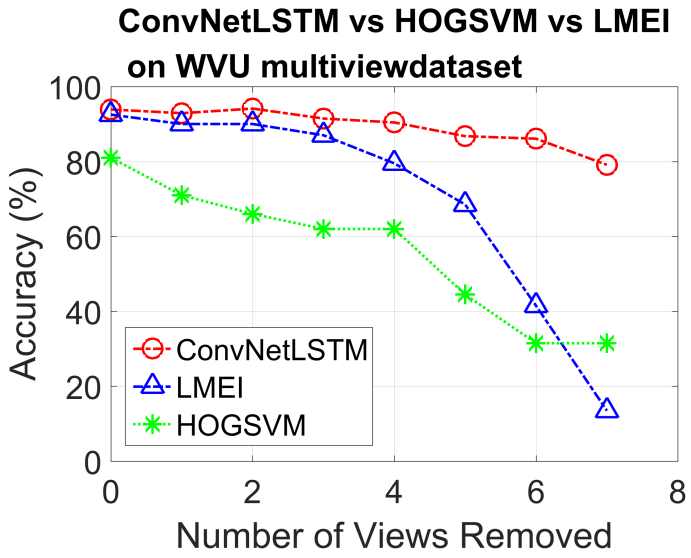

Fig 11 Recognition accuracy of ConvNet LSTM on WVU multi-view action recognition dataset with different number of available views. Accuracy is compared with those using motion history images as described in.[14]

Fig. 11 shows the recognition accuracy using the deep learning technique and compares it with the results obtained using the more traditional approach using spatio-temporal features (motion history images) as used in.[14] In Fig. 11, LMEI corresponds to the technique where the image is divided into rectangular blocks and motion history images are computed per block. The localized motion energy images are used as feature vectors. This technique incorporates spatial locality in the feature vectors.[14] In the HOG-SVM technique, histogram of gradients is computed for the motion history image as described in Section 4.1.1 and used as feature vectors. The result shows that the ConvNet LSTM technique has higher accuracy even with fewer views. As number of views increase towards 8, both spatio-temporal and deep learning techniques perform almost equally. However, the ConvNet LSTM technique does not show a sharp drop in accuracy when there are fewer views available. Note that the results with motion history image in this dataset are worse than those in the driving simulator dataset, especially with fewer views. This is because the number of action classes are larger and thus the spatio-temporal motion history signatures become

more similar across the actions.

5 Conclusion

In this paper,we have shown how deep neural networks with LSTMs can be used to easily fuse features from multiple camera views and have applied it for the problem of driver activity analysis. We have presented two different fusion techniques and our results show significant improvement in accuracy using both these approaches. We also evaluated the impact of multiple layers of LSTMs and we observed that an additional LSTM layer had much lesser impact than the addition of camera views. We also showed that the deep learning technique performs better than using motion history images by comparing on the driving simulation dataset as well as another publicly available multi-view action recognition dataset with 8 camera views and 12 action classes.

As an immediate next step, we would like to design higher level classifiers that can utilize the sequential output of the multi-view ConvNet LSTMs and use that to temporally cluster the output labels, identify outliers and predict start and end of action sequences. We would also like to apply the activity analysis technique designed here to study the effectiveness of interactive consoles on vehicles and quantitatively evaluate the distraction caused to drivers because of these interactions.

6 Acknowledgments

This study was partially sponsored by the Alpha Foundation for the Improvement of Mine Safety and Health, Inc. (ALPHA FOUNDATION). The views, opinions and recommendations expressed herein are solely those of the authors and do not imply any endorsement by the ALPHA FOUNDATION, its Directors and staff. Their financial contribution is gratefully acknowledged.

References

1 J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks* **61**, 85–117 (2015).

2 A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 1097–1105 (2012).

3 K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

4 P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

5 J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human tracking using convolutional neural networks," *Neural Networks, IEEE Transactions on* **21**(10), 1610–1623 (2010).

6 A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 1725–1732, IEEE (2014).

7 S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **35**(1), 221–231 (2013).

8 M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling.," in *INTERSPEECH*, (2012).

9 D. Soutner and L. Müller, "Application of lstm neural networks in language modelling," in *Text, Speech, and Dialogue*, 105–112, Springer (2013).

10 J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description." arXiv preprint arXiv:1411.4389 (2014).

11 J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification." arXiv preprint arXiv:1503.08909 (2015).

12 Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1110–1118 (2015).

13 H. Fang, H. Si, and L. Chen, "Recurrent neural network for human activity recognition in smart home," in *Proceedings of 2013 Chinese Intelligent Automation Conference*, 341–348 (2013).

14 R. Kavi and V. Kulathumani, "Real-time recognition of action sequences using a distributed video sensor network," *Journal of Sensor and Actuator Networks* **2**(3), 486–508 (2013).

15 M. B. Holte, C. Tran, M. M. Trivedi, and T. B. Moeslund, "Human action recognition using multiple views: a comparative perspective on recent developments," in *Proceedings of the 2011 joint ACM workshop on Human gesture and behavior understanding*, 47–52, ACM (2011).

16 V. Kulathumani, V. Kecojevic, A. Nimbarte, and R. Kavi, "Integrated surface mine safety system: Alpha mining foundation technical report." http://www.alpha-foundation.

org/wp-content/uploads/2016/01/AFC113-15_WVU_-FinalRpt_Approved.pdf (2015).

17 J. W. Davis, "Hierarchical motion history images for recognizing human motion," in *Detection and Recognition of Events in Video, 2001. Proceedings. IEEE Workshop on*, 39–46 (2001).

18 L. Shao, L. Ji, Y. Liu, and J. Shang, "Human action segmentation and recognition via motion and shape analysis," *Pattern Recognition Letters* **33**(2012), 438–445 (2012).

19 V. Kulathumani, R. Kavi, and S. Ramagiri, "Wvu multi-view action recognition dataset." http://www.csee.wvu.edu/~vkkulathumani/wvu-action.html (2011).

20 R. Poppe, "A survey on vision-based human action recognition," *Image and vision computing* **28**(6), 976–990 (2010).

21 D. Weinland, R. Ronfard, and E. Boyer, "A survey of vision-based methods for action representation, segmentation and recognition," *Computer Vision and Image Understanding* **115**(2), 224–241 (2011).

22 A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, 1725–1732 (2014).

23 B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016).

24 Y. Du, Y. Fu, and L. Wang, "Representation learning of temporal dynamics for skeleton-based action recognition," *IEEE Transactions on Image Processing* **25**(7), 3010–3022 (2016).

25 A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+d: A large scale dataset for 3d human activity analysis," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (2016).

26 A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, 1725–1732, IEEE (2014).

27 J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: A cpu and gpu math compiler in python," in *Proc. 9th Python in Science Conf*, 1–7 (2010).

28 N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research* **15**(1), 1929–1958 (2014).

29 T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning* **4**, 2 (2012).

30 N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, **1**, 886–893, IEEE (2005).

31 J. Feng and J. Xiao, "View-invariant human action recognition via robust locally adaptive multi-view learning," *Frontiers of Information Technology & Electronic Engineering* **16**(11), 917–929 (2015).

32 A. K. Kushwaha and R. Srivastava, "Multiview human activity recognition system based on

spatiotemporal template for video surveillance system," *Journal of Electronic Imaging* **24** (2015).

33 M. Edwards, J. Deng, and X. Xie, "From pose to activity," *Comput. Vis. Image Underst.* **144**, 73–105 (2016).

34 S. Dubuisson and C. Gonzales, "A survey of datasets for visual tracking," *Mach. Vision Appl.* **27**, 23–52 (2016).

List of Figures

List of Tables

6.1 Biographies

Rahul Kavi is a Ph.D. candidate in the Department of Computer Science and Electrical Engineering at the West Virginia University. His interests are in Computer Vision, Robotics and Machine Learning. He was part of the team that won the NASA sample return robot challenge 2015 with a $\$100,000$ prize money.

Dr. Vinod Kulathumani is an Associate Professor in the Department of Computer Science and Electrical Engineering at the West Virginia University. His research is in the area of scalable and robust sensor actuator systems with applications in the area of distributed camera networks, mobile ad-hoc networks and vehicular networks.

FNU Rohit is a graduate student in the Department of Computer Science and Electrical Engineering at the West Virginia University. His interests are in Computer Vision, Robotics and Machine Learning.

Dr. Vlad Kecojevic is a Professor of Mining Engineering and the Massey Foundation Professor at West Virginia University (WVU). His research areas of expertise include surface mining, surface mine safety, information technology, and environmental issues in surface mining. Dr. Kecojevic has been elected as the 2015-2016 President of the Society of Mining Professors (SOMP) and he currently serves as the member of the SOMP Council.