Supplementary Material: Recurrent Convolutions for Causal 3D CNNs

Gurkirt Singh Fabio Cuzzolin
Visual Artificial Intelligence Laboratory (VAIL), Oxoford Brookes University
gurkirt.singh-2015@brookes.ac.uk

In this supplementary material, we first present a discussion Section 1, followed by implementation details in Section 2. Next, we show a more in-depth analysis to understand the parameters of recurrent convolution in Section 3.

1. Discussion

1.1. Effect of Weight Initialisation.

Weights of inflated 2D layers in ResNet (base networks) [3] based I3D and our RCN networks are initialised with weights from a pre-trained ImageNet model train on RGB images. In the case of RCN inflation in the third dimension is 1, practically, it is 2D weight matrix with the third dimension being 1. As a result, we do need to replicate the weights like in the 3D layer of I3D.

Random initialisation for hidden state parameters resulted in suboptimal training. Thus, in all the experiments with RCN, we used identity matrix initialisation instead. Here we show the result of training RCN with different initilisation of hidden-state w_{hh} convolution and spatial w_{xh} convolution in Table 1. The first row of the table where both w_{xh} and w_{hh} of all the RCUs in RCN is initialised randomly using normal initialisation process described in [2].

Table 1 show the result of different initialisation of w_{xh} and w_{hh} respectively. It is clear from the last row of the table that RCN performs best if w_{xh} and w_{hh} are initialised with ImageNet and identity matrix respectively. It is noteworthy that the performance difference is relatively small in first three rows of that table as compared to the last row jump.

w_{xh} init	w_{hh} init	Clip-Acc%	Video-Acc%
Random	Random	49.0	61.2
Random	Identity	49.3	61.8
ImageNet	Random	50.4	62.5
ImageNet	Identity	**53.0**	**65.1**

Table 1. Video-level and clip-level action recognition accuracy on the Kinetics validation set for different initilisation of 2D layer (w_{xh}) and hidden state unit weights (w_{hh}) in RCU with ResNet-18-based RCN models trained on 8 frame-long clip as an input. These results were obtained using our older training setup, which was is suboptimal than the current one presented in the main paper. The main difference is due to data, and number iterations have increased.

1.2. ImageNet initialisation

proves to be useful for both the I3D and our RCN models. While (2+1)D performs (Table **??**, row 6) better than RCN (row 5) with the random initialisation, our RCN recovers to improve over (2+1)D (row 6) with ImageNet initialisation, whereas (2+1)D cannot make use of free ImageNet initialisation. This seems to be a severe drawback for the (2+1)D model, and a big advantage for I3D and RCN. One may argue that, if the purpose is to build on existing 2D models, then RCN and I3D are a better choice, whereas if new 3D models are preferred then (2+1)D might prove useful. The latter does provide better performance with the random initialisation, but at the price of requiring many more parameters than RCN.

1.3. On Input Clip Length

Input clip length is another factor in determining the final performance of any network. We can see from the Table that all the Inception-based models are trained on 64 frame-long clips, one of the reasons why Inception nets work better that ResNet models while using fewer parameters. Among the latter, ResNet101-I3D-NL [5] is shown to work better with an even longer input clip length. Thus, the evidence supports that training on larger input sequences boosts performance while clashing with memory limitations.

In our experiments, as mentioned, we stuck to 8 or 16 frame clips as input and compared our proposed RCN with baseline I3D models. We think this provides enough evidence of the validity of our proposal to move away from temporal convolutional networks [6, 5, 4, 1], and replace them with more sophisticated *and* causal structures. As with the role of additional layers, it is fair to predict that more extensive training on more extended clips (32,64,128) has a serious potential to take RCN to outperform state of the art in absolute terms.

1.4. On the Efficient Training of 3D Networks

Two basic things are clear from our experience with training heavy 3D models (I3D, (2+1)D, RCN) on large-scale datasets such as Kinetics. Firstly, training is very computationally expensive and memory bulky; secondly, longer input clips are crucial to achieving better optimisation which, however, renders the first issue even more severe. We feel that how to train these model efficiently is still a wide-open problem, whose solution is essential to speed up the process for broader adoption. We observed that ImageNet initialisation does speed up the training procedure, and helps reach local minima much quicker. In the case of both I3D and RCN, ImageNet initialisation improves the video classification accuracy on Kinetics by almost 3% compared to random initialisation when using the same number of training iterations, as shown in the first and last row of Table 1.

The bottom line is that we should strive for more efficient implementations of 3D models for the sake of their adoption.

2. Implementation Details

All models with 8 and 16 long clip as input are trained with batch size of 64 and 32 respectively. We used maximum for four 1080Ti GPUs (11GB VRAM each) while training all the models. We used Pytorch library to implement all of the models from scratch, and we used the ResNet architecture of 2D model from pytorch library's model zoo[1].

Input Data Preparations. We apply the same set of input data transformations to each frame in an input clip. Data agumentation transformations include random crop, radom horizontal flip, affine transformations, and temporal jittering.

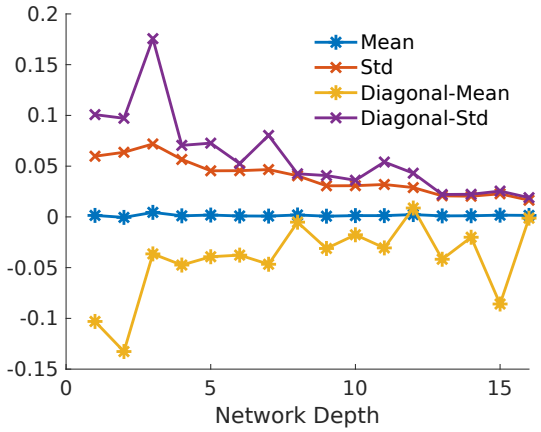
3. Hidden State Parameters

Figure 1 show the mean and standard deviation (Std) of weight matrixes (w_{hh}) of the hidden state at every RCU layer in our RCN network. We see the drop in standard deviation across different models (ResNet-18 in Figure 1 and ResNet-34 in Figure 2(b)) and different input clip-lengths used for training ((8 in Figure 1 and 16 in Figure 2(b))). The drop in standard deviation and an increase in mean values of diagonal values means towards sparsity. The increase in sparsity with depth increase can also be observed in the Figure 3, especially in the last layer (Layer 16), elements on diagonal have higher weights, and the first layer (Layer 1) has higher values distribute all around in the matrix.

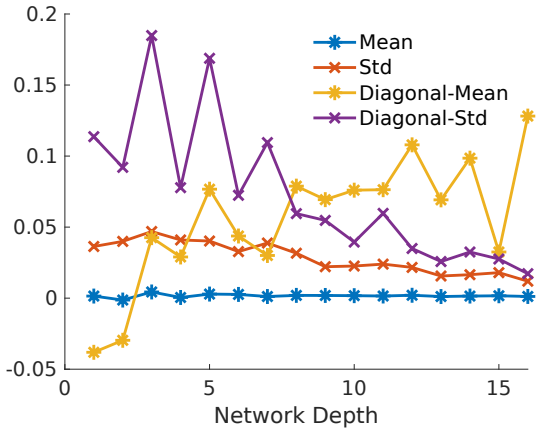
We can conclude the increase in sparsity increases focuses on diagonal elements, hence, the same feature focus more on the time dimension of the same feature than other features from the previous hidden-state. It is a reasonable explanation because we expect the network to learn from the different feature at small depth-level and focus more on same feature (i.e. focus on time aspect of the same feature) more with the increase in the depth-level.

Acknowledgement: This work was partly supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 779813 (SARAS).

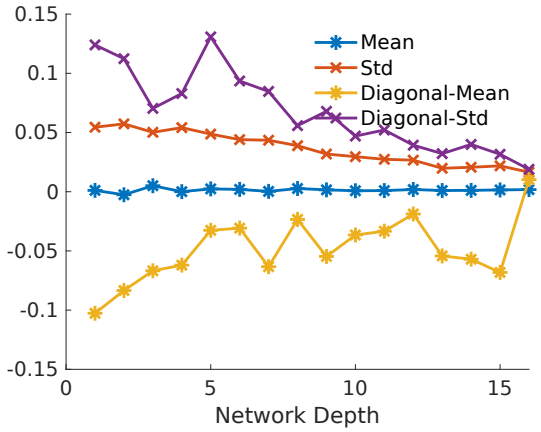
[1] https://github.com/pytorch/vision/tree/master/torchvision/models



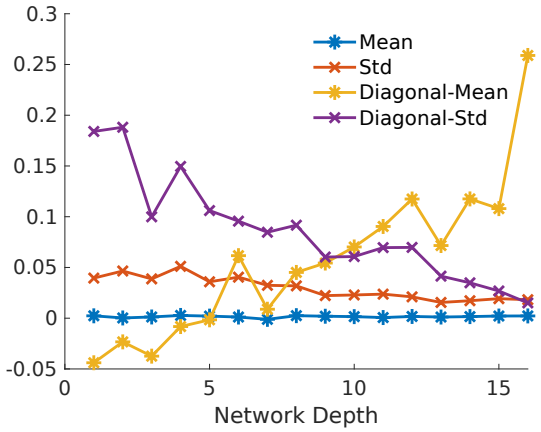
(a) 2D: Random; Recurrent: Random



(b) 2D: Random; Recurrent: Identity

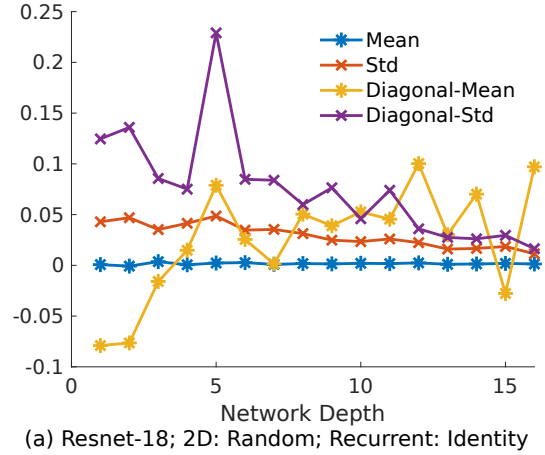


(c) 2D: ImgNet; Recurrent: Random

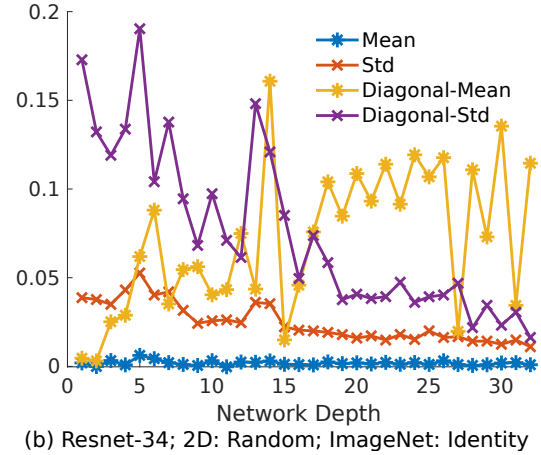


(d) 2D: ImgNet; Recurrent: Identity

Figure 1. Mean and standard deviation (Std) of weight matrixes (w_{hh}) of the hidden state at every RCU layer in the network based on ResNet-18, along with the mean, and Std of diagonal elements. Networks are trained on an input clip-length of 8.



(a) Resnet-18; 2D: Random; Recurrent: Identity



(b) Resnet-34; 2D: Random; ImageNet: Identity

Figure 2. Mean and standard deviation (Std) of weight matrixes (w_{hh}) of the hidden state at every RCU layer in the network based on ResNet-18 (a) and ResNet-34 (b), along with the mean, and Std of diagonal elements. Networks are trained on an input clip-length of 16.

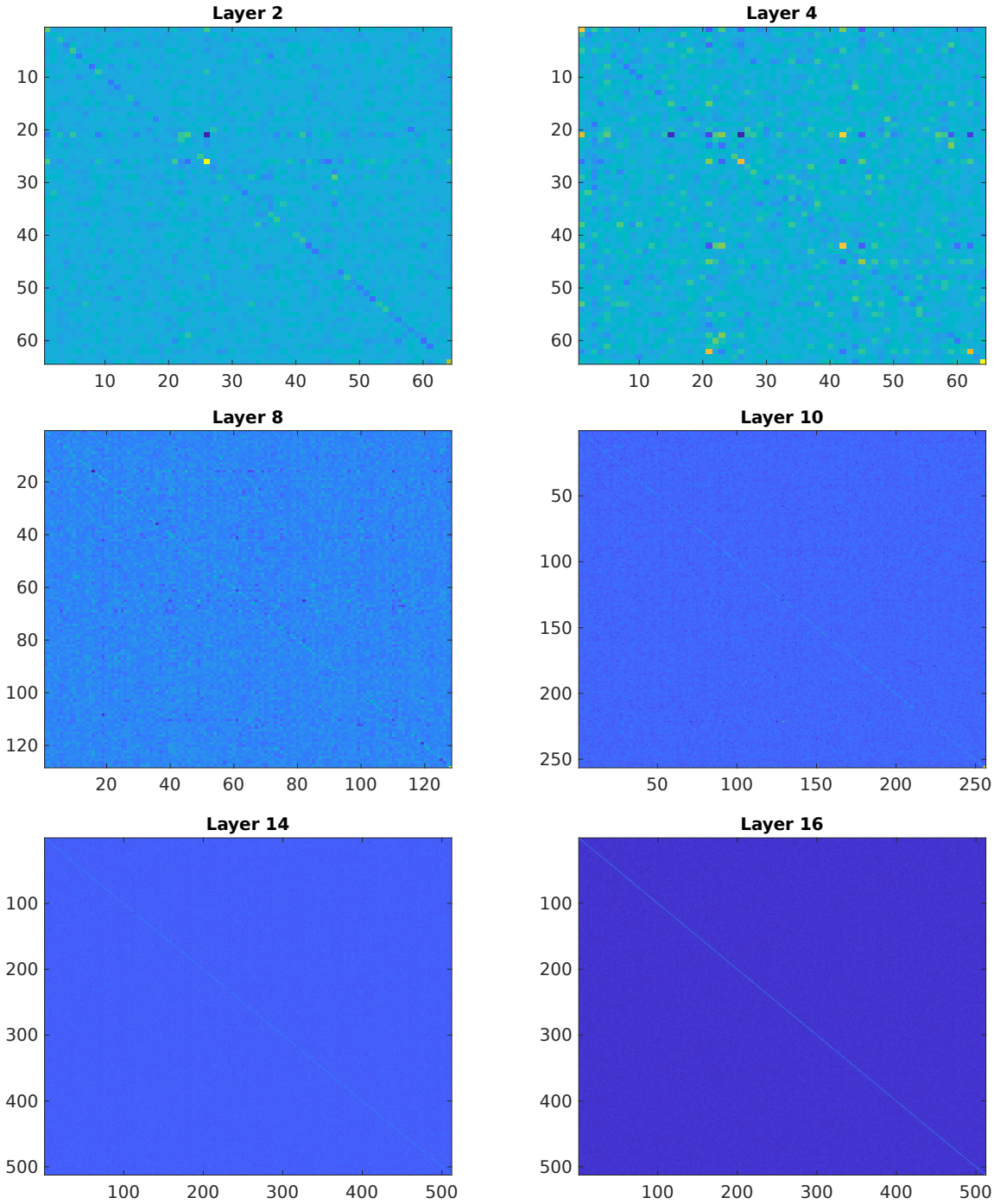


Figure 3. Heat map of weight matrixes (w_{hh}) of the hidden state at every RCU layer in the network based on ResNet-18. Parula colourmap (from matlab) is used, where blue is smallest values and yellow is the highest i.e. dark to bright.

References

[1] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4724–4733. IEEE, 2017. 1

[2] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 1

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[4] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 1

[5] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

[6] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proc. European Conf. Computer Vision*, pages 305–321, 2018. 1