Folded Recurrent Neural Networks for Future Video Prediction

Marc Oliu[1,3], Javier Selva[2,3], Sergio Escalera[2,3]

{[1]Universitat Oberta de Catalunya, [2]Universitat de Barcelona}, Barcelona, Spain [3]Centre de Visió per Computador, Cerdanyola del Vallès, Spain

Motivation

Future video prediction:

- **Applications**: Unsupervised learning scene structure and spatio-temporal relationships
- **Challenges**: High variability and non-specificity of future frames

We introduce double-mapping Gated Recurrent Units (dGRU). Standard GRUs update an output state given an input. We also consider the input as a **recurrent state**, using an **extra set of logic gates** to update it given the output, allowing for:

- Lower memory and computational costs
- Mitigation and recovery from temporal error propagation
- An identity function during training, helping convergence
- Model explainability/pruning through layer removal

Method

We propose stacking multiple conv. dGRU layers, allowing for:

- **Bidirectional flow** of information between pixel space and deepest representations
- **Stratified encoding** of the information

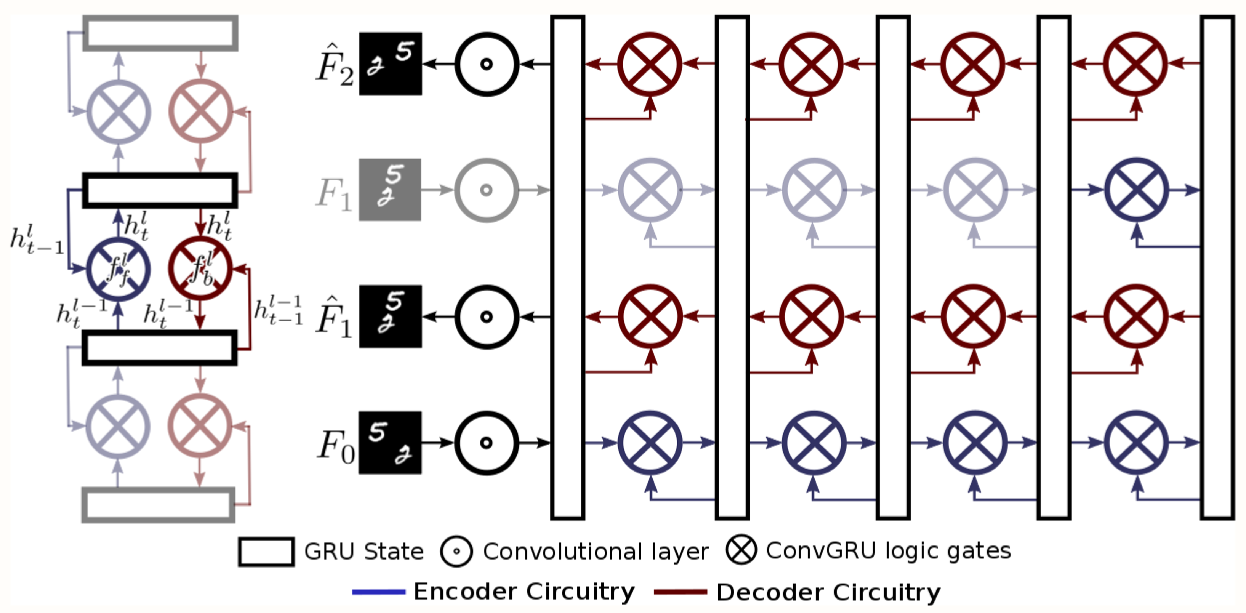

Figure 1: dGRU connectivity (left) and fRNN topology (right)

double-mapping GRU

Both input h_t^{l-1} and output h_t^l are considered recurrent states and updated based on their previous state and current output/input. Two sets of gates are used. This results in a **double-mapping** where either the input or output can be updated, projecting the states into the future.

$$h_t^l = f_f^l(h_t^{l-1}, h_{t-1}^l)$$
$$h_t^{l-1} = f_b^l(h_t^l, h_{t-1}^{l-1}) \tag{1}$$

Folded Recurrent Neural Network

Stacking dGRU layers results in an autoencoder-like structure:

- First layer serves as both input and output
- Forward and backward functions f_f^l and f_b^l serve as the encoder/decoder

Since **states are shared** between encoder and decoder, executing one updates the other, allowing us to:

- Only use the encoder for consecutive inputs
- Only use the decoder for consecutive predictions
- Avoid prediction feedbacks
- Halve the computational and memory costs

References

[1] Lotter, W., Kreiman, G., ... *Deep predictive coding networks for video prediction and unsupervised learning.* In ICLR 2016

[2] Srivastava, N., Mansimov, E., ... *Unsupervised learning of video representations using lstms.* In PMLR (2015)

[3] Mathieu, M., Couprie, C., ... *Deep multi-scale video prediction beyond mean square error.* In ICLR (2016)

[4] Villegas, R., Yang, J., ... *Decomposing motion and content for natural video sequence prediction.* In ICLR (2017)

Results

	MMNIST			KTH			UCF101		
	MSE	PSNR	DSSIM	MSE	PSNR	DSSIM	MSE	PSNR	DSSIM
Baseline	0.06989	11.745	0.20718	0.00366	29.071	0.07900	0.01294	22.859	0.15043
RLadder	0.04254	13.857	0.13788	**0.00139**	**31.268**	**0.05945**	0.00918	23.558	0.13395
Lotter [1]	0.04161	13.968	0.13825	0.00309	28.424	0.09170	0.01550	19.869	0.21389
Srivastava [2]	0.01737	18.183	0.08164	0.00995	21.220	0.19860	0.14866	10.021	0.42555
Mathieu [3]	0.02748	15.969	0.29565	0.00180	29.341	0.10410	0.00926	22.781	0.16262
Villegas [4]	0.04254	13.857	0.13896	0.00165	30.946	0.07657	0.00940	23.457	0.14150
fRNN	**0.00947**	**21.386**	**0.04376**	0.00175	29.299	0.07251	**0.00908**	**23.872**	**0.13055**

Table 1: Average results over 10 time steps.

Results on three datasets, comparing with other *sota* methods and a baseline with bridge connections, show (Tab.1, Fig. 4):

- fRNN is best on MMNIST and UCF101
- Recurrent baseline is best on KTH
- fRNN has greater stability through time

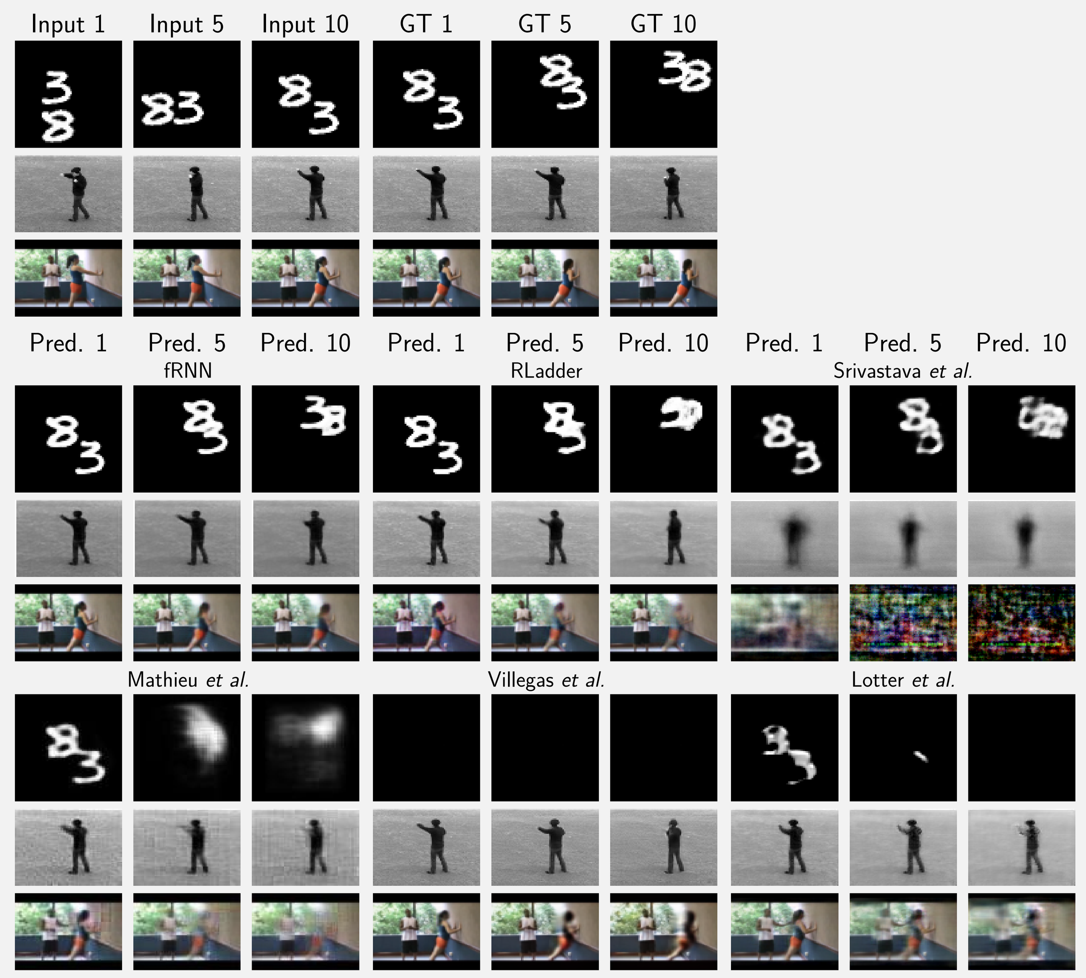

Figure 2: Predictions at 1, 5, and 10 time steps from the last ground truth frame.

Stratification analysis

fRNN is resilient to layer removal, allowing for a **visual analysis of the behaviour** encoded at each layer (Fig. 3).

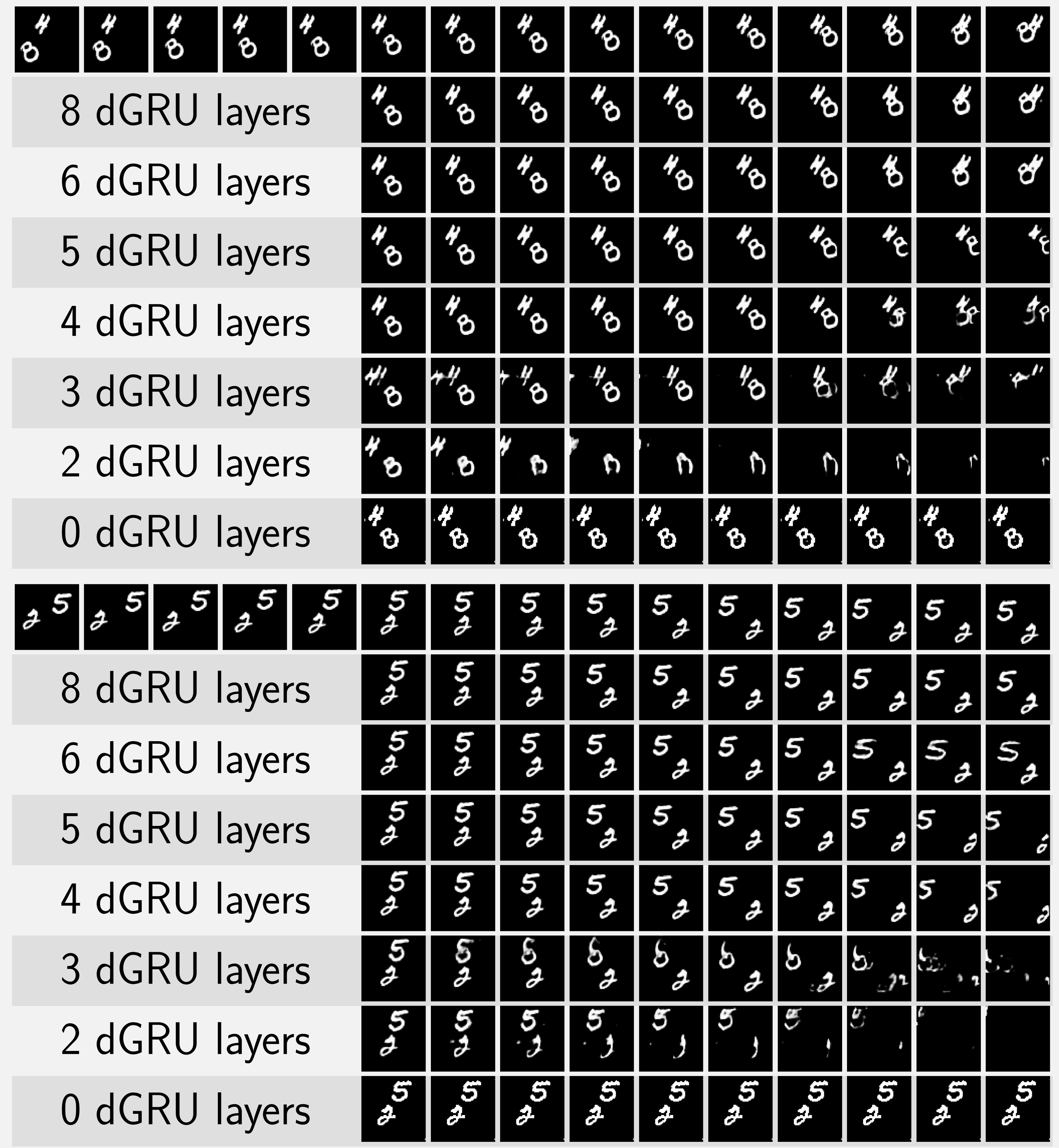

Figure 3: Moving MNIST predictions with fRNN layer removal.

fRNN provides an **identity function** during training (see last row), facilitating convergence on homogeneous backgrounds.

Figure 4: Quantitative results on MMNIST, KTH and UCF-101 in terms of the number of timesteps since the last input frame.

European Conference on Computer Vision 2018, Munich, Germany

https://github.com/moliusimon/frnn