PreRNN and BandRNN for Video Understanding

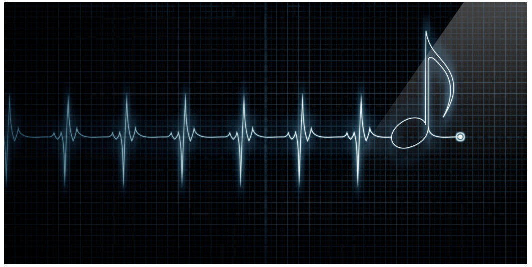
Xiaodong Yang

with Pavlo Molchanov, Matthijs Van Keirsbilck, Alex Keller, Jan Kautz

Sequential Learning Problems

machine translation

handwriting modeling



polyphonic music modeling

speech recognition

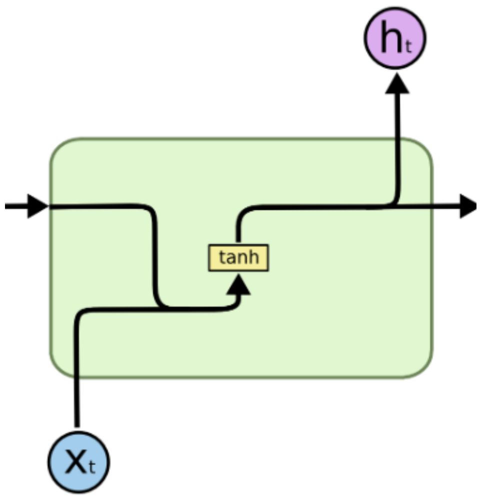
intelligent video analytics

"You mean to imply that I have nothing to eat out of.... On the contrary, I can supply you with everything even if you want to give dinner parties," warmly replied Chichagov, who tried by every word he spoke to prove his own rectitude and therefore imagined Kutuzov to be animated by the same desire.

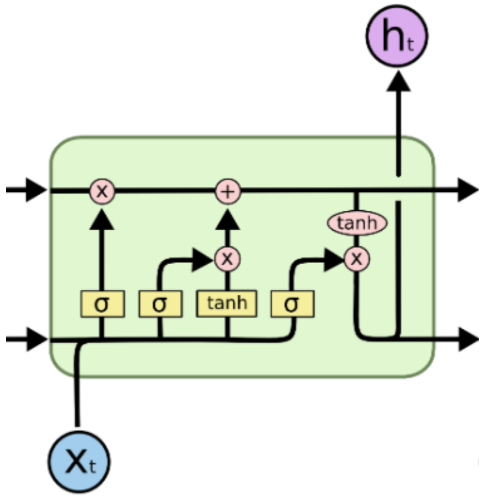
Kutuzov, shrugging his shoulders, replied with his subtle penetrating smile: "I meant merely to say what I said."

language modeling

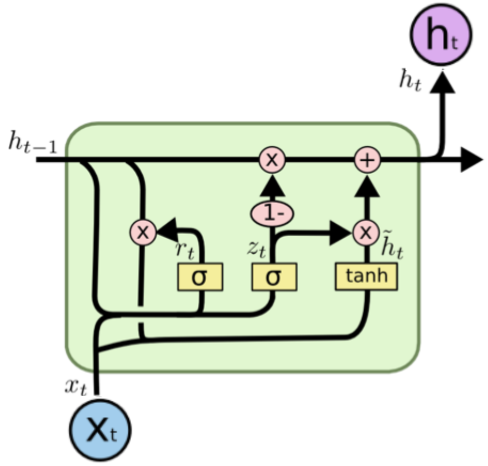
RNNs in Sequential Learning



Vanilla RNN (VRNN)



Long Short-Term Memory (LSTM)

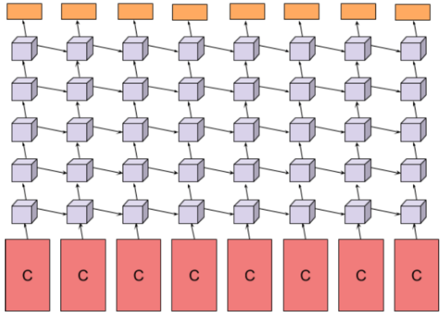


Gated Recurrent Unit (GRU)

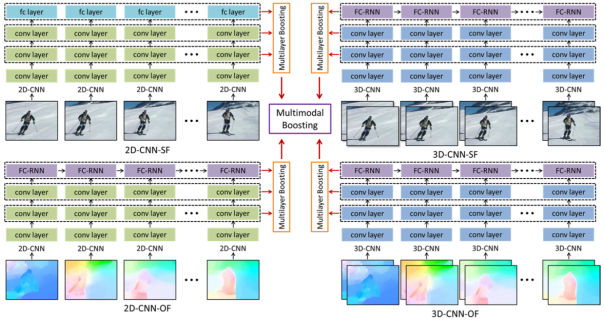
Image courtesy of Christopher Olah.

RNNs in Video Understanding

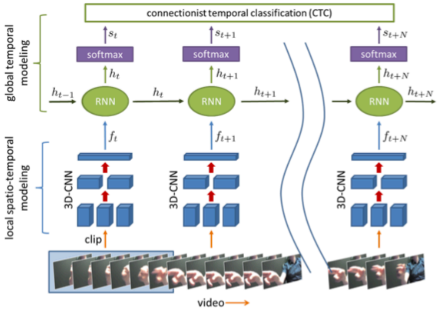
Examples



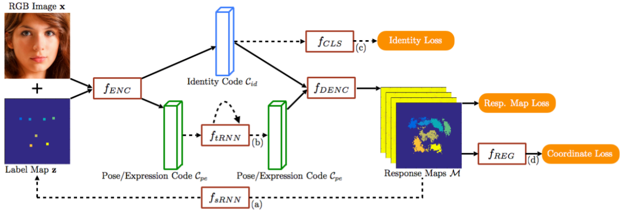
Ng et al. CVPR 2015



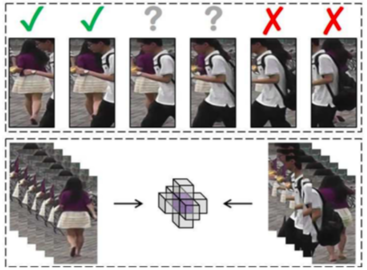
Yang et al. ACMMM 2016



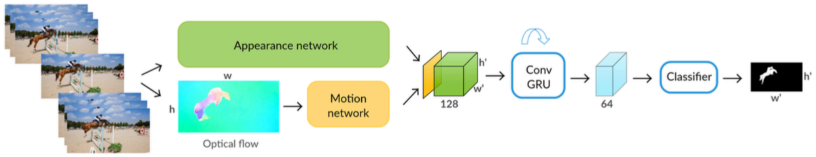
Molchanov et al. CVPR 2016



Peng et al. ECCV 2016



Zhou et al. CVPR 2017



Tokmakov et al. ICCV 2017

RNNs in Video Understanding

Distinct Properties of Videos

- Processing unit in a more structured format such as image or snippet
  - CNNs serve as backbone networks
  - Pre-trained on large-scale image or video datasets
  - How to construct RNNs to better leverage the pre-trained CNNs
- Large redundancy and diverse temporal dependencies on different applications
  - Such as facial alignment, hand gesture recognition, activity recognition
  - Poorly understood which recurrent structure or which gating mechanism best suits

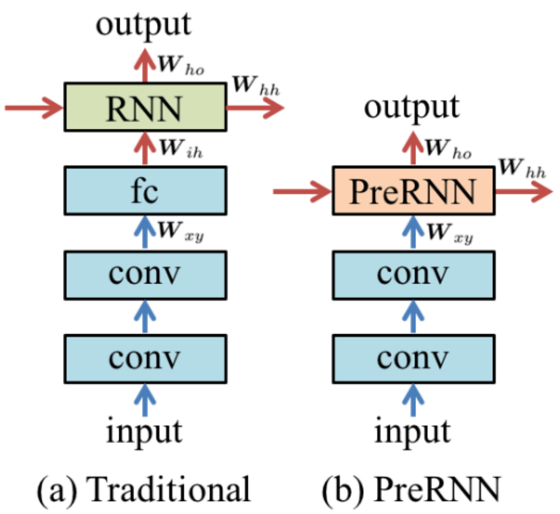
PreRNN+BandRNN for Video Understanding

Overview

- PreRNN: make pre-trained CNNs recurrent by transforming pre-trained convolutional or fully connected layers into recurrent layers
- PreRNN-SIH: simplify input-to-hidden states and reduce recurrent parameters
- BandRNN: sparsify hidden-to-hidden weights and further reduce recurrent parameters

PreRNN for Video Understanding

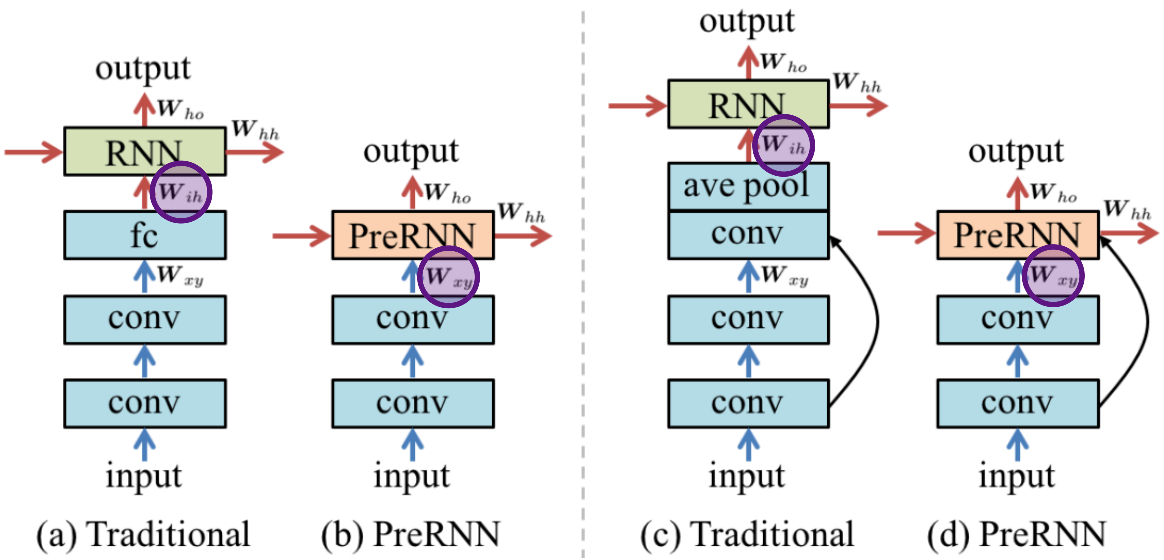
Overview



A schematic overview of the traditional RNN and the proposed PreRNN.

PreRNN for Video Understanding

Overview



A schematic overview of the traditional RNN and the proposed PreRNN.

Traditional RNNs

Notation

- VRNN

$$\boldsymbol{h}_t = \mathcal{H}(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1})$$

- LSTM

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{W}_{ii}\boldsymbol{y}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{W}_{if}\boldsymbol{y}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{W}_{io}\boldsymbol{y}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{ic}\boldsymbol{y}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \qquad \boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$$

- GRU

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{W}_{ir}\boldsymbol{y}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{W}_{iz}\boldsymbol{y}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})) \qquad \boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t$$

$\mathcal{H}$ activation function

Traditional RNNs

Input-to-Hidden State

- VRNN

$$\boldsymbol{h}_t = \mathcal{H}(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1})$$

- LSTM

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{W}_{ii}\boldsymbol{y}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{W}_{if}\boldsymbol{y}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{W}_{io}\boldsymbol{y}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{ic}\boldsymbol{y}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \qquad \boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$$

- GRU

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{W}_{ir}\boldsymbol{y}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{W}_{iz}\boldsymbol{y}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})) \qquad \boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t$$

$\mathcal{H}$ activation function

Traditional RNNs

Hidden-to-Hidden State

- VRNN

$$\boldsymbol{h}_t = \mathcal{H}(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1})$$

- LSTM

$$\boldsymbol{i}_t = \text{sigm}(\boldsymbol{W}_{ii}\boldsymbol{y}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \text{sigm}(\boldsymbol{W}_{if}\boldsymbol{y}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \text{sigm}(\boldsymbol{W}_{io}\boldsymbol{y}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{ic}\boldsymbol{y}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \qquad \boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$$

- GRU

$$\boldsymbol{r}_t = \text{sigm}(\boldsymbol{W}_{ir}\boldsymbol{y}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \text{sigm}(\boldsymbol{W}_{iz}\boldsymbol{y}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1})) \qquad \boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t$$

$\mathcal{H}$ activation function

PreRNN

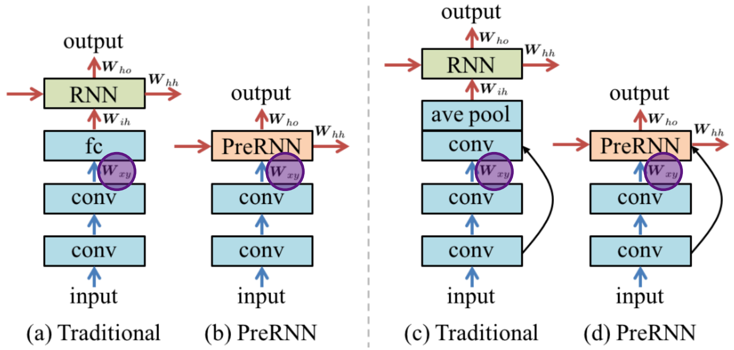
Transformation for VRNN

- A feedforward layer in CNNs

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{W}_{xy} \circ \boldsymbol{x})$$

- PreVRNN

$$\boldsymbol{y}_t = \begin{cases} \mathcal{H}(\boldsymbol{W}_{xy}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{y}_{t-1}) & \text{a } \texttt{fc} \text{ layer} \\ \mathcal{H}(\mathcal{P}(\mathcal{B}(\boldsymbol{W}_{xy} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) + \boldsymbol{W}_{hh}\boldsymbol{y}_{t-1}) & \text{a } \texttt{conv} \text{ layer} \end{cases}$$



A schematic overview of the traditional RNN and the proposed PreRNN.

$\mathcal{H}$ activation function
$\mathcal{B}$ batch normalization
$\mathcal{P}$ pooling

PreRNN

Transformation for LSTM

- A feedforward layer in CNNs

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{W}_{xy} \circ \boldsymbol{x})$$

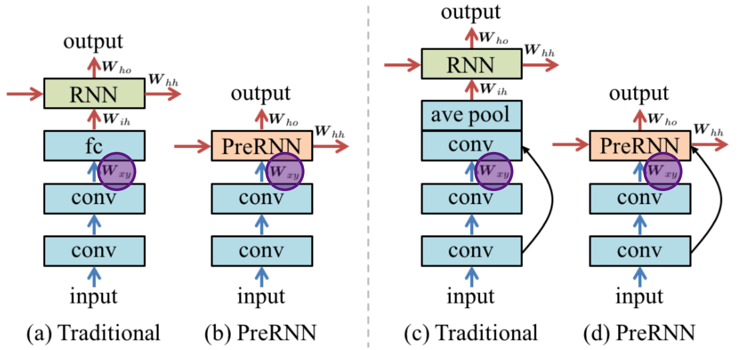
- Gate-dependent input-to-hidden state

$$\boldsymbol{u}_t(g) = \begin{cases} \boldsymbol{W}^p_{ig}\boldsymbol{x}_t & \text{a } \texttt{fc} \text{ layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}^p_{ig} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) & \text{a } \texttt{conv} \text{ layer} \end{cases}$$

- PreLSTM

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{u}_t(i) + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{u}_t(f) + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{u}_t(o) + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{u}_t(c) + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$



A schematic overview of the traditional RNN and the proposed PreRNN.

$\mathcal{H}$ activation function

$\mathcal{B}$ batch normalization

$\mathcal{P}$ pooling

PreRNN

Transformation for GRU

- A feedforward layer in CNNs

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{W}_{xy} \circ \boldsymbol{x})$$

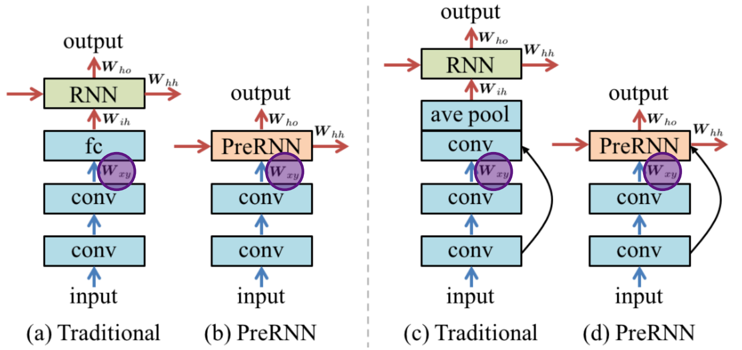
- Gate-dependent input-to-hidden state

$$\boldsymbol{u}_t(g) = \begin{cases} \boldsymbol{W}^p_{ig}\boldsymbol{x}_t & \text{a } \texttt{fc} \text{ layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}^p_{ig} * \boldsymbol{x}_t) + \gamma_t) & \text{a } \texttt{conv} \text{ layer} \end{cases}$$

- PreGRU

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{u}_t(r) + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{u}_t(z) + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{u}_t(h) + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$



A schematic overview of the traditional RNN and the proposed PreRNN.

$\mathcal{H}$ activation function
$\mathcal{B}$ batch normalization
$\mathcal{P}$ pooling

PreRNN

Comparison to Traditional RNN

- VRNN => PreVRNN

$$\boldsymbol{h}_t = \mathcal{H}(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{y}_t = \begin{cases} \mathcal{H}(\boldsymbol{W}_{xy}\boldsymbol{x}_t + \boldsymbol{W}_{hh}\boldsymbol{y}_{t-1}) & \text{a fc layer} \\ \mathcal{H}(\mathcal{P}(\mathcal{B}(\boldsymbol{W}_{xy} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) + \boldsymbol{W}_{hh}\boldsymbol{y}_{t-1}) & \text{a conv layer} \end{cases}$$

- LSTM => PreLSTM

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{W}_{ii}\boldsymbol{y}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \quad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{W}_{if}\boldsymbol{y}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{W}_{io}\boldsymbol{y}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \quad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{W}_{ic}\boldsymbol{y}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{u}_t(i) + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \quad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{u}_t(f) + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{u}_t(o) + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \quad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{u}_t(c) + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

- GRU => PreGRU

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{W}_{ir}\boldsymbol{y}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \quad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{W}_{iz}\boldsymbol{y}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{W}_{ih}\boldsymbol{y}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{u}_t(r) + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \quad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{u}_t(z) + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{u}_t(h) + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$

$\mathcal{H}$ activation function
$\mathcal{B}$ batch normalization
$\mathcal{P}$ pooling

PreRNN-SIH

Transformation for LSTM

- A feedforward layer in CNNs

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{W}_{xy} \circ \boldsymbol{x})$$

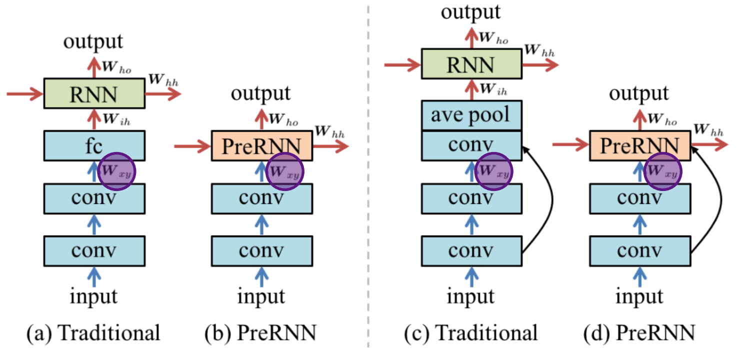
- Single input-to-hidden (SIH) state

$$\boldsymbol{v}_t = \begin{cases} \boldsymbol{W}_{xy}\boldsymbol{x}_t & \text{a fc layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}_{xy} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) & \text{a conv layer} \end{cases}$$

- PreLSTM-SIH

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{v}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$



A schematic overview of the traditional RNN and the proposed PreRNN.

$\mathcal{H}$ activation function
$\mathcal{B}$ batch normalization
$\mathcal{P}$ pooling

PreRNN-SIH

Transformation for GRU

- A feedforward layer in CNNs

$$\boldsymbol{y} = \mathcal{H}(\boldsymbol{W}_{xy} \circ \boldsymbol{x})$$

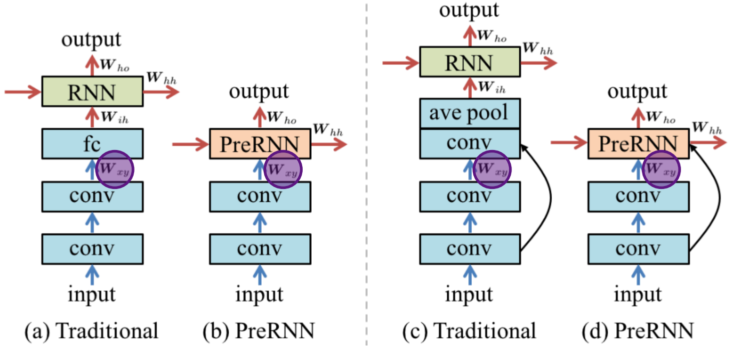
- Single input-to-hidden (SIH) state

$$\boldsymbol{v}_t = \begin{cases} \boldsymbol{W}_{xy}\boldsymbol{x}_t & \text{a } \texttt{fc} \text{ layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}_{xy} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) & \text{a } \texttt{conv} \text{ layer} \end{cases}$$

- PreGRU-SIH

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{v}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$



A schematic overview of the traditional RNN and the proposed PreRNN.

$\mathcal{H}$ activation function

$\mathcal{B}$ batch normalization

$\mathcal{P}$ pooling

PreRNN-SIH

Comparison to PreRNN

- (Gate-dependent => Single) input-to-hidden state

$$\boldsymbol{u}_t(g) = \begin{cases} \boldsymbol{W}^p_{ig}\boldsymbol{x}_t & \text{a fc layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}^p_{ig} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) & \text{a conv layer} \end{cases}$$

$$\boldsymbol{v}_t = \begin{cases} \boldsymbol{W}_{xy}\boldsymbol{x}_t & \text{a fc layer} \\ \mathcal{P}(\mathcal{B}(\boldsymbol{W}_{xy} * \boldsymbol{x}_t) + \boldsymbol{\gamma}_t) & \text{a conv layer} \end{cases}$$

- PreLSTM => PreLSTM-SIH

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{u}_t(i) + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{u}_t(f) + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{u}_t(o) + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{u}_t(c) + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{i}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{f}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1})$$

$$\boldsymbol{o}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1}) \qquad \tilde{\boldsymbol{c}}_t = \tanh(\boldsymbol{v}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1})$$

- PreGRU => PreGRU-SIH

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{u}_t(r) + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{u}_t(z) + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{u}_t(h) + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$

$$\boldsymbol{r}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hr}\boldsymbol{h}_{t-1}) \qquad \boldsymbol{z}_t = \mathrm{sigm}(\boldsymbol{v}_t + \boldsymbol{W}_{hz}\boldsymbol{h}_{t-1})$$

$$\tilde{\boldsymbol{h}}_t = \tanh(\boldsymbol{v}_t + \boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}))$$

$\mathcal{H}$ activation function
$\mathcal{B}$ batch normalization
$\mathcal{P}$ pooling

Applications

Diversity

Applications	Sequences	CNNs	Datasets	Objectives
Sequential Face Alignment	Color	VGG16 [38]	300VW [7]	ℓ_2
Hand Gesture Recognition	Color & Depth	C3D [43]	NVGesture [28]	CTC [15]
Action Recognition	Color & Flow	ResNet50 [20]	UCF101 [39]	NLL

Summary of the diverse experiments in terms of applications, video types, pre-trained backbone CNNs, benchmark datasets, and objective functions.

Applications

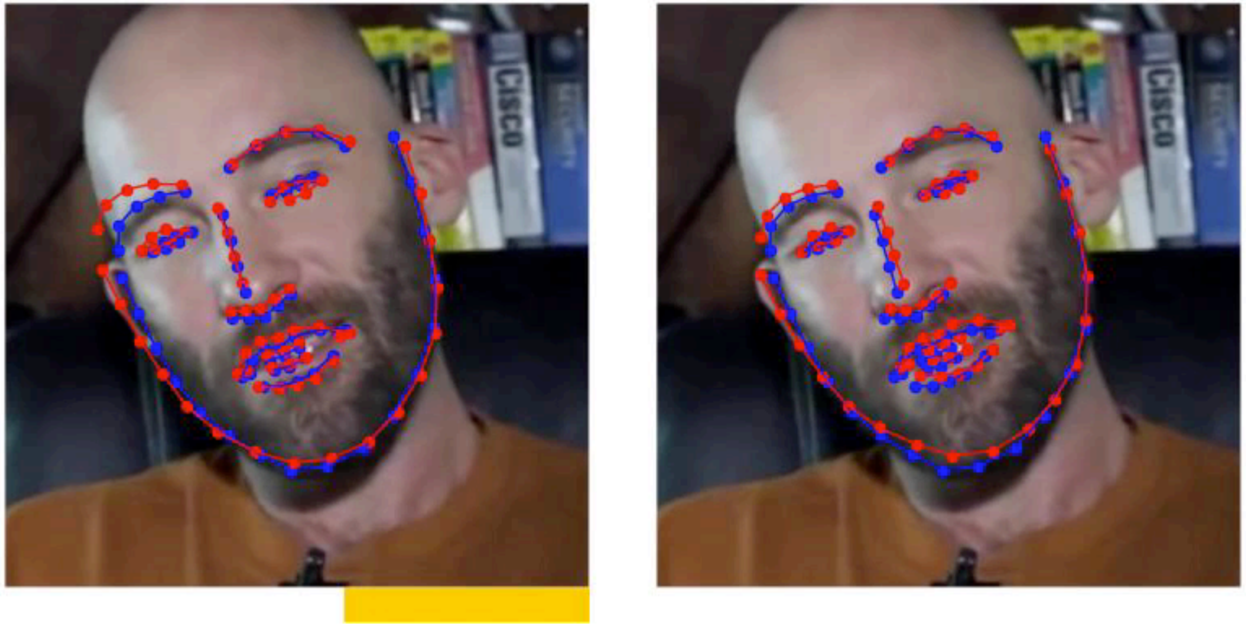
Face Alignment

Applications	Sequences	CNNs	Datasets	Objectives
Sequential Face Alignment	Color	VGG16 [38]	300VW [7]	ℓ_2
Hand Gesture Recognition	Color & Depth	C3D [43]	NVGesture [28]	CTC [15]
Action Recognition	Color & Flow	ResNet50 [20]	UCF101 [39]	NLL

Summary of the diverse experiments in terms of applications, video types, pre-trained backbone CNNs, benchmark datasets, and objective functions.

Applications

Face Alignment



Examples of detected facial landmarks on the 300VW dataset by traditional GRU (left) and PreGRU (right).

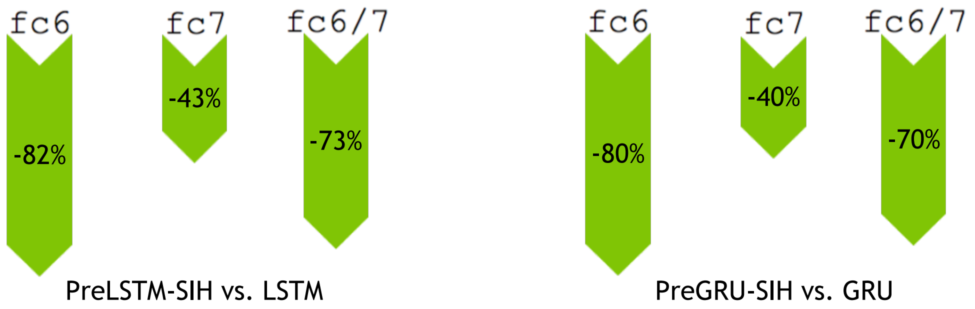
Blue dots: ground truth
Red dots: detected landmarks
Green bar: PreGRU with larger error
Yellow bar: traditional GRU with larger error
Bar length: error scale

Applications

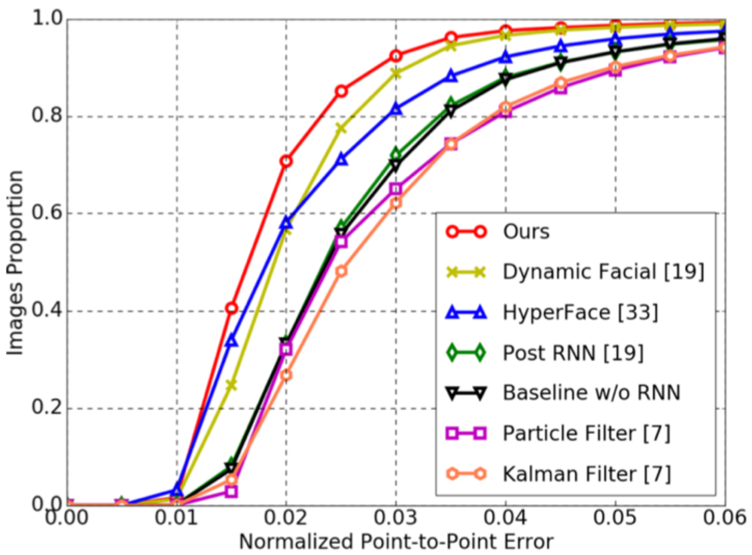
Face Alignment

	Traditional		PreRNN			PreRNN-SIH		
	1 layer	2 layers	fc6	fc7	fc6/7	fc6	fc7	fc6/7
VRNN	0.704	0.716	0.757	0.742	**0.763**	-	-	-
LSTM	0.718	0.671	**0.769**	0.754	0.746	0.743	0.746	0.719
GRU	0.722	0.698	**0.772**	0.755	0.761	0.768	0.748	0.762

AUC of traditional RNNs and our proposed PreRNN(-SIH) on 300VW.



Ratios of reduced recurrent parameters by PreRNN-SIH.



Comparison of our approach with the state-of-the-art methods.

Applications

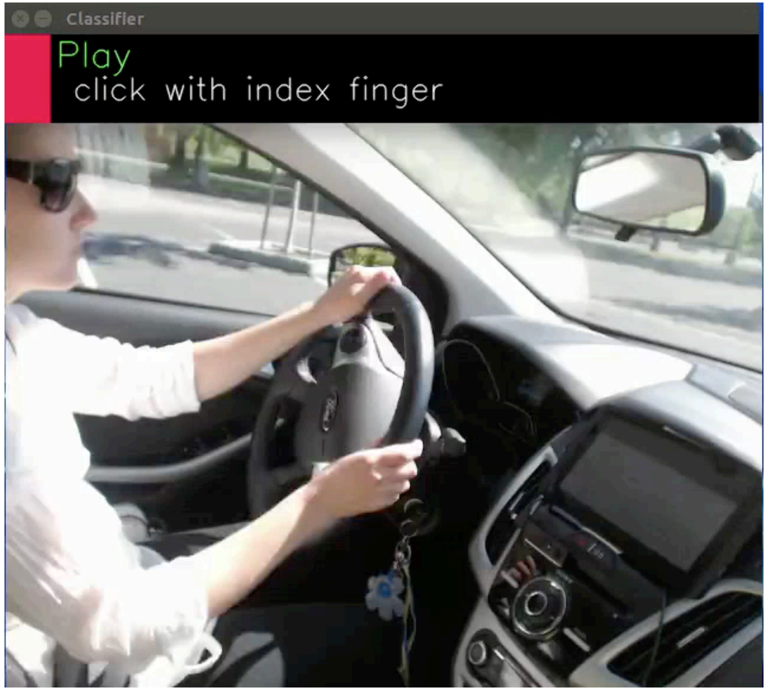
Hand Gesture Recognition

Applications	Sequences	CNNs	Datasets	Objectives
Sequential Face Alignment	Color	VGG16 [38]	300VW [7]	ℓ_2
Hand Gesture Recognition	Color & Depth	C3D [43]	NVGesture [28]	CTC [15]
Action Recognition	Color & Flow	ResNet50 [20]	UCF101 [39]	NLL

Summary of the diverse experiments in terms of applications, video types, pre-trained backbone CNNs, benchmark datasets, and objective functions.

Applications

Hand Gesture Recognition



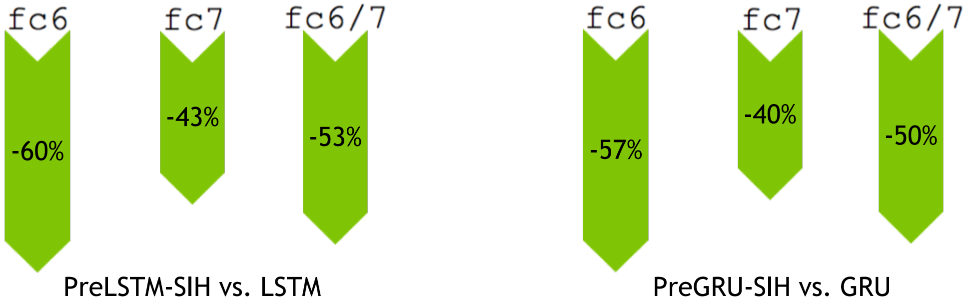
PreVRNN based hand gesture recognition system for in-car media player control.

Applications

Hand Gesture Recognition

	Traditional		PreRNN			PreRNN-SIH		
	1 layer	2 layers	fc6	fc7	fc6/7	fc6	fc7	fc6/7
VRNN	83.3%	80.8%	81.9%	82.9%	**84.4%**	-	-	-
LSTM	81.3%	81.3%	81.7%	81.9%	82.7%	80.0%	81.7%	**84.2%**
GRU	81.9%	82.5%	82.1%	81.0%	83.1%	**84.4%**	79.8%	83.8%

Classification accuracy of traditional RNNs and our proposed PreRNN(-SIH) on NVGesture.



Ratios of reduced recurrent parameters by PreRNN-SIH.

Method	Modality	Accuracy
C3D [43]	Color	69.3%
R3DCNN [28]	Color	74.1%
Ours	Color	**76.5%**
SNV [49]	Depth	70.7%
C3D [43]	Depth	78.8%
R3DCNN [28]	Depth	80.3%
Ours	Depth	**84.4%**
Two-Stream [37]	Color + Flow	65.6%
iDT [45]	Color + Flow	73.4%
R3DCNN [28]	Five Modalities	83.8%
Baseline (w/o RNN)	Color + Depth	81.0%
Ours	Color + Depth	**85.0%**

Comparison of our approach with the state-of-the-art methods.

Applications

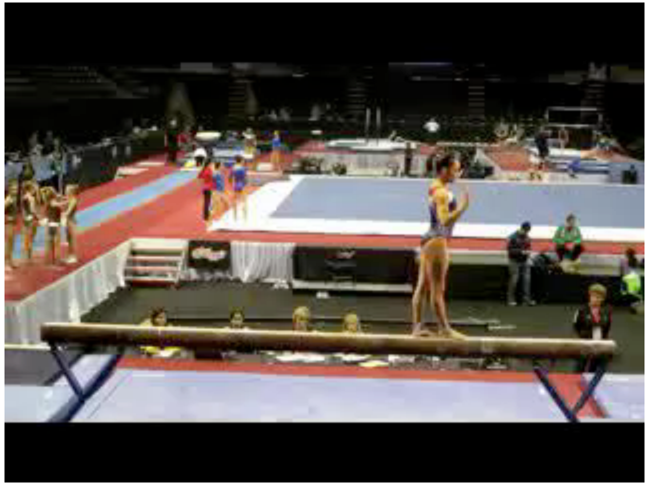
Action Recognition

Applications	Sequences	CNNs	Datasets	Objectives
Sequential Face Alignment	Color	VGG16 [38]	300VW [7]	ℓ_2
Hand Gesture Recognition	Color & Depth	C3D [43]	NVGesture [28]	CTC [15]
Action Recognition	Color & Flow	ResNet50 [20]	UCF101 [39]	NLL

Summary of the diverse experiments in terms of applications, video types, pre-trained backbone CNNs, benchmark datasets, and objective functions.

Applications

Action Recognition



PreGRU => balance beam
Traditional GRU => floor gymnastics

PreGRU => skijet
Traditional GRU => kayaking

Examples of misclassified videos by traditional GRU, but corrected by PreGRU.

Applications

Action Recognition

	Traditional			PreRNN			PreRNN-SIH		
	Color	Flow	Comb	Color	Flow	Comb	Color	Flow	Comb
VRNN	82.9%	83.6%	91.6%	83.8%	84.6%	**92.7%**	-	-	-
LSTM	83.4%	84.0%	92.5%	85.3%	84.8%	93.2%	85.0%	84.6%	**93.5%**
GRU	83.6%	83.8%	92.2%	84.3%	85.2%	**93.7%**	84.9%	84.7%	93.3%

Classification accuracy of traditional RNNs and our proposed PreRNN(-SIH) on UCF101.

-16%

PreLSTM-SIH vs. LSTM

-14%

PreGRU-SIH vs. GRU

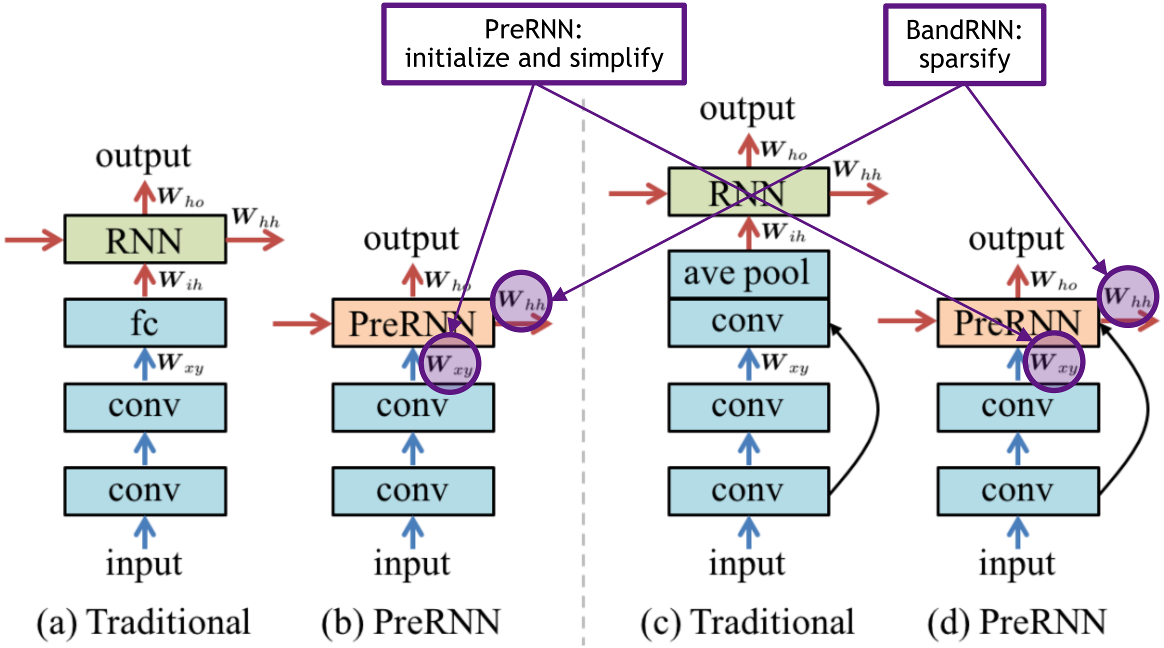
Ratios of reduced recurrent parameters by PreRNN-SIH.

Method	Accuracy
Dynamic Image Nets [3]	76.9%
Long-Term Recurrent ConvNet [10]	82.9%
Composite LSTM Model [40]	84.3%
C3D [43]	85.2%
iDT [45]	86.4%
Two-Stream ConvNet [37]	88.0%
Multilayer Multimodal Fusion [48]	91.6%
Long-Term ConvNets [44]	91.7%
Two-Stream Fusion [14]	92.5%
Spatiotemporal ResNets [12]	93.4%
Inflated 3D ConvNets [4]	93.4%
Temporal Segment Networks [46]	**94.2%**
Spatiotemporal Multiplier Nets [13]	**94.2%**
Baseline (w/o RNN)	91.7%
Ours	**94.3%**

Comparison of our approach with the state-of-the-art methods.

PreRNN+BandRNN

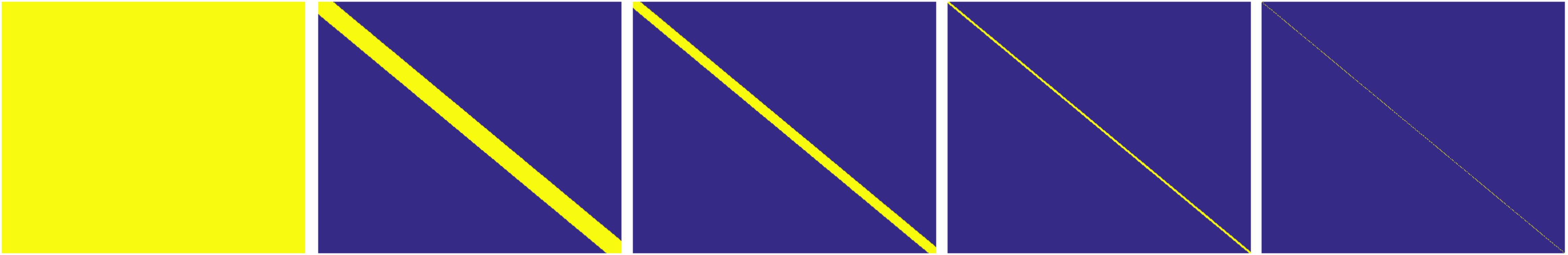
Overview



A schematic overview of the traditional RNN and the proposed PreRNN+BandRNN.

BandRNN

Sparsify Hidden-to-Hidden Weight Matrix



full (2048x2048) | 201 (10%) | 101 (5%) | 21 (1%) | diag (1)

PreRNN+BandRNN

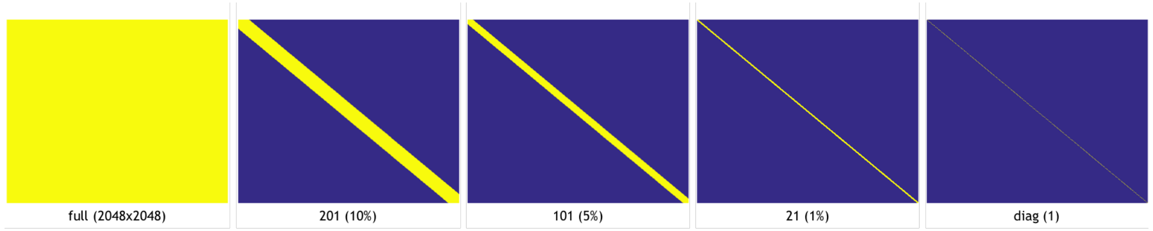
Action Recognition

PreVRNN	Sparsity (H2H)
91.9%	diag
92.3%	1%
92.0%	5%
92.2%	10%
92.7%	full
Traditional VRNN: 91.6%	
Baseline (w/o RNN): 91.2%	

PreLSTM	PreLSTM-SIH	Sparsity (H2H)
92.7%	92.6%	diag
92.8%	93.0%	1%
92.9%	92.7%	5%
93.2%	92.9%	10%
93.2%	93.5%	full
Traditional LSTM: 92.5%		
Baseline (w/o RNN): 91.2%		

PreGRU	PreGRU-SIH	Sparsity (H2H)
92.8%	92.5%	diag
92.8%	92.5%	1%
92.8%	92.3%	5%
92.9%	92.6%	10%
93.7%	93.3%	full
Traditional GRU: 92.2%		
Baseline (w/o RNN): 91.2%		

Classification accuracy of PreRNN(-SIH) with various sparsity of hidden-to-hidden weight matrices on UCF101.

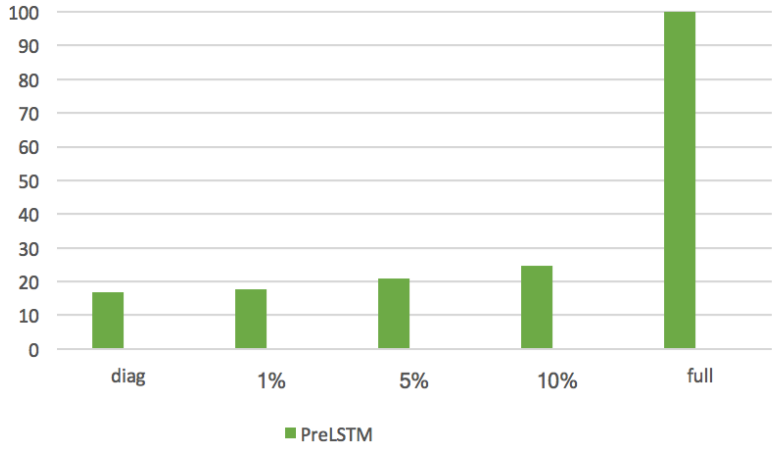


PreRNN+BandRNN

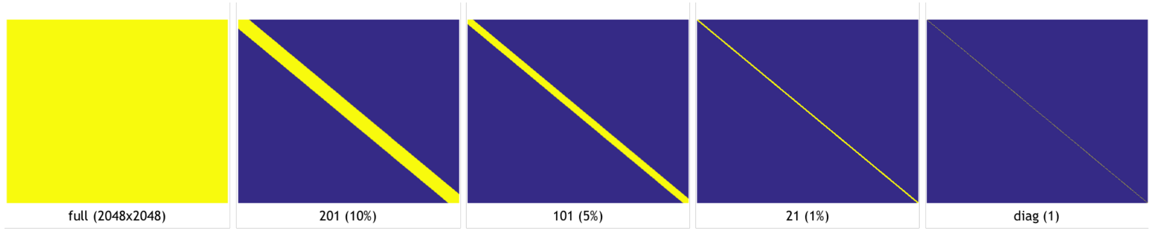
Action Recognition

PreLSTM	PreLSTM-SIH	Sparsity (H2H)
92.7%	92.6%	diag
92.8%	93.0%	1%
92.9%	92.7%	5%
93.2%	92.9%	10%
93.2%	93.5%	full
Traditional LSTM: 92.5%		
Baseline (w/o RNN): 91.2%		

Classification accuracy of PreLSTM(-SIH) with various sparsity of hidden-to-hidden weight matrices on UCF101.



Ratios (%) of recurrent parameters of PreLSTM(-SIH) to traditional LSTM with various sparsity of hidden-to-hidden weight matrices.

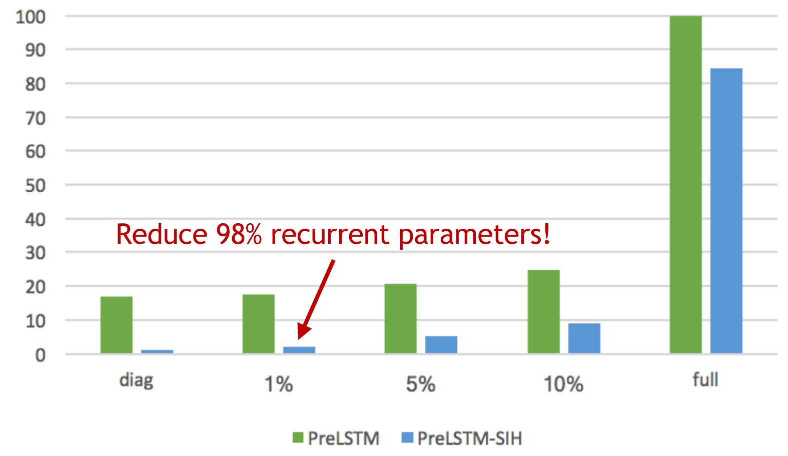


PreRNN+BandRNN

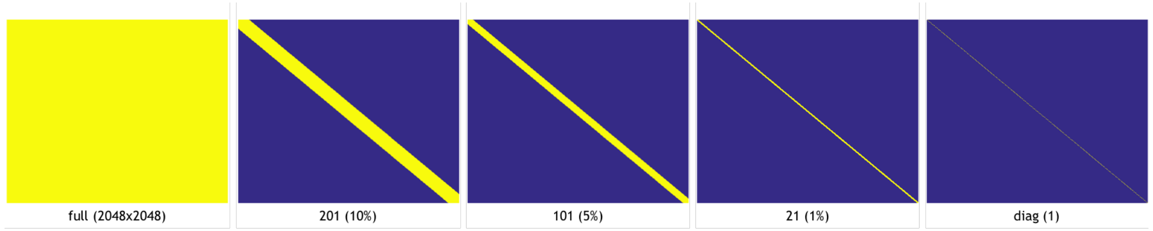
Action Recognition

PreLSTM	PreLSTM-SIH	Sparsity (H2H)
92.7%	92.6%	diag
92.8%	93.0%	1%
92.9%	92.7%	5%
93.2%	92.9%	10%
93.2%	93.5%	full
Traditional LSTM: 92.5%		
Baseline (w/o RNN): 91.2%		

Classification accuracy of PreLSTM(-SIH) with various sparsity of hidden-to-hidden weight matrices on UCF101.



Ratios (%) of recurrent parameters of PreLSTM(-SIH) to traditional LSTM with various sparsity of hidden-to-hidden weight matrices.



PreRNN+BandRNN for Video Understanding

Summary

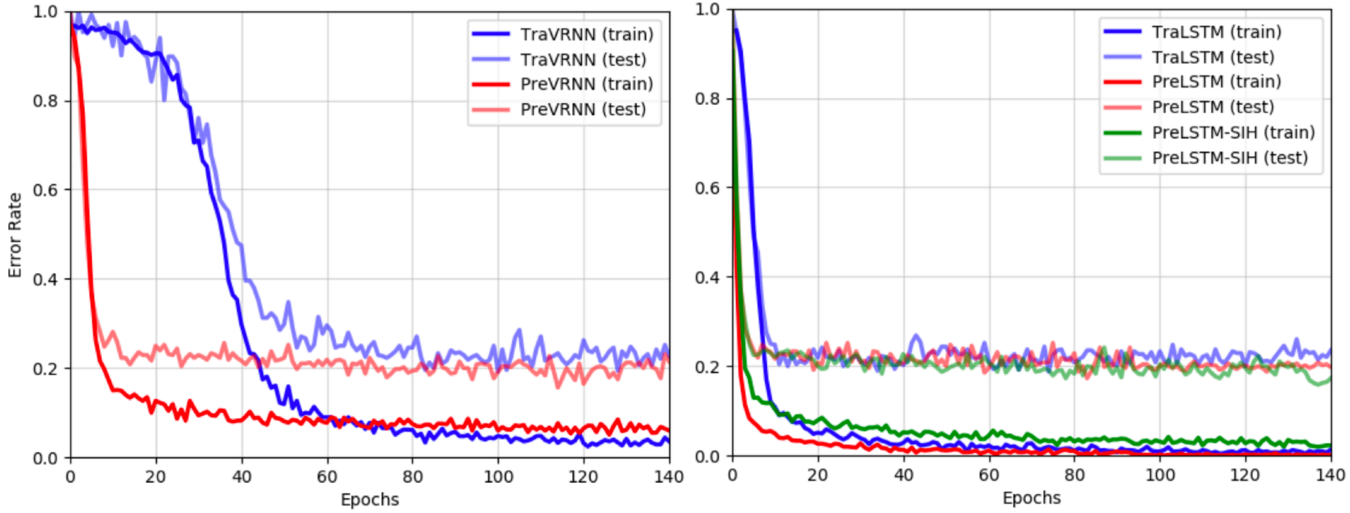
- PreRNN: better leverage strong generalization of pre-trained CNNs
- PreRNN-SIH: simplify input-to-hidden states and largely reduce input-to-hidden recurrent parameters
- BandRNN: sparsify hidden-to-hidden weight matrices and further significantly reduce hidden-to-hidden recurrent parameters
- PreRNN+BandRNN: simple and effective, produce better or comparable results to traditional RNNs, while only introduce super lightweight recurrent parameters

Majority of this work can be found at:
X. Yang, P. Molchanov, J. Kautz. Making Convolutional Networks Recurrent for Visual Sequence Learning. CVPR, 2018.

NVIDIA

Convergence

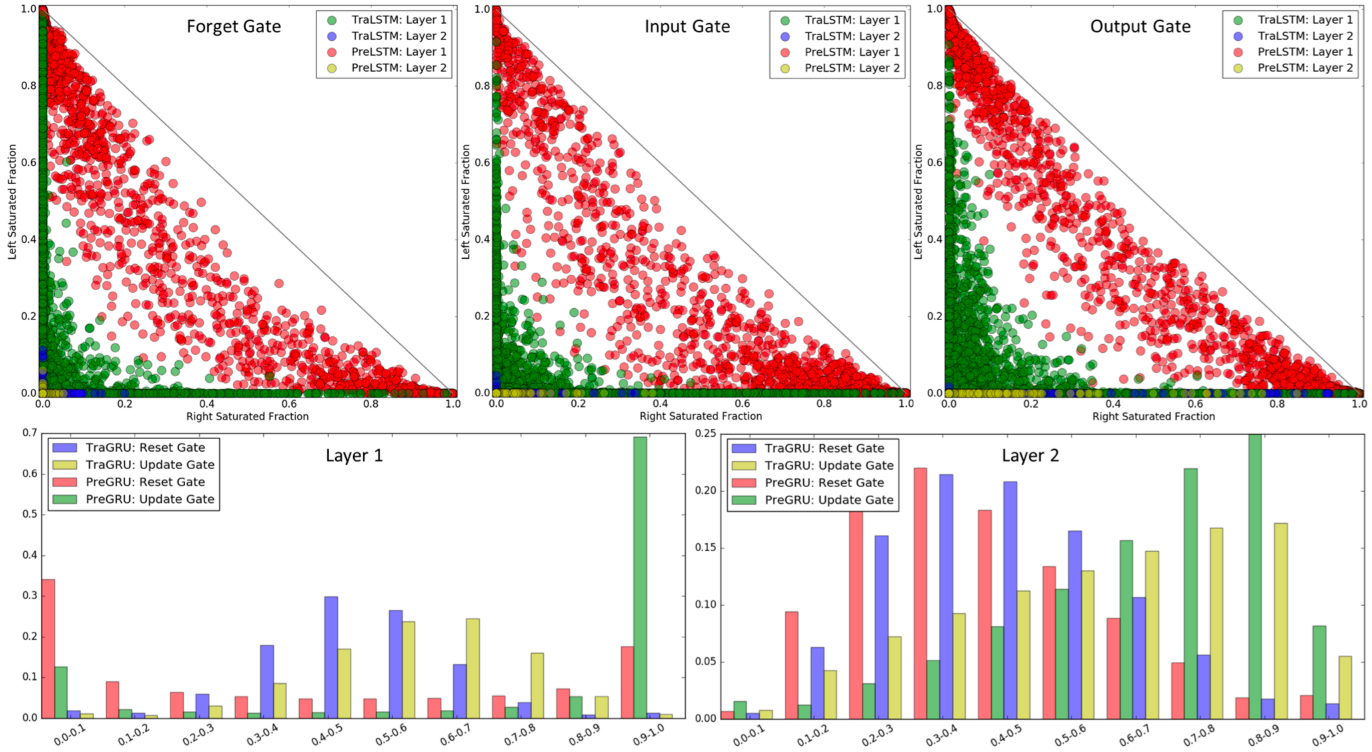
PreRNN Converges Faster



Comparison of the training processes between the traditional RNN and our proposed PreRNN(-SIH) for VRNN (left) and LSTM (right).

Understanding RNNs

Internal Mechanism of Traditional RNN and PreRNN



Examples of the gate activation distribution for LSTM and GRU. Top: saturation plots of the fraction of times that each gate unit is left or right saturated for LSTM. Bottom: activation histograms over 10 bins for GRU.