#956 Do RNN and LSTM have Long Memory?

Jingyu Zhao [1], Feiqing Huang [1], Jia Lv [2], Yanjie Duan [2], Zhen Qin [2], Guodong Li [1], Guangjian Tian [2]

[1] Department of Statistics and Actuarial Science, The University of Hong Kong, Hong Kong, China

[2] Huawei Noah's Ark Lab, Hong Kong, China

ICML, Jul 2020

Overview

- Pros and Cons of Long short-term Memory (LSTM)

+

- Countless applications
- Numerically proven effectiveness on synthetic tasks
  e.g., $y_{T+1} = y_1$

−

- Markovian updates: states at time t only depend on the states at time $t-1$
- Statistical tests show that LSTM cannot
  (i) produce long memory output given white noise as input
  (ii) produce short memory residual given long memory input

Overview

- The term *Long Memory* in …

Deep Learning	**Statistics**
• Not well-defined yet	• Well-defined for stationary stochastic processes • No exogenous inputs
• Short memory has a synonym “vanishing gradients” from the algorithmic / training aspect	• From the modeling perspective e.g. fractional ARIMA (ARFIMA)
• Datasets: language, music, etc.	• Datasets: records in finance, dendrochronology, hydrology, etc.

Overview

- Our contributions
    - 1. Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.

      $\Rightarrow$ RNN and LSTM do not have long memory most of the time!

exogenous inputs: $y_0, y_1, \cdots, y_T$; $x_0, x_1, \cdots, x_T$ $\xrightarrow{\text{predict}}$ y_{T+1}

no exogenous inputs: $y_0, y_1, \cdots, y_T$ $\xrightarrow{\text{predict}}$ y_{T+1}

Overview

- Our contributions
  - 1. Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.
  - 2. We propose a new definition of long memory recurrent networks, allowing exogenous inputs.
    - We want the correlation between the target y_t and the input x_{t-k} to decay slowly as $k \to \infty$.

Overview

- Our contributions

  - 1. Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.

  - 2. We propose a new definition of long memory recurrent networks, allowing exogenous inputs.

  - 3. We explore theory-guided applications: MRNN and MLSTM.
    - A long memory filter is added to RNN at the input or LSTM at the cell states, to pass distant information to current hidden units.

Overview

- Our contributions
  - 1. Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.
  - 2. We propose a new definition of long memory recurrent networks, allowing exogenous inputs.
  - 3. We explore theory-guided applications: MRNN and MLSTM
  - 4. We conduct numerical studies to illustrate the advantages of proposed models.
    - They can be used alone or merge into current network structures.

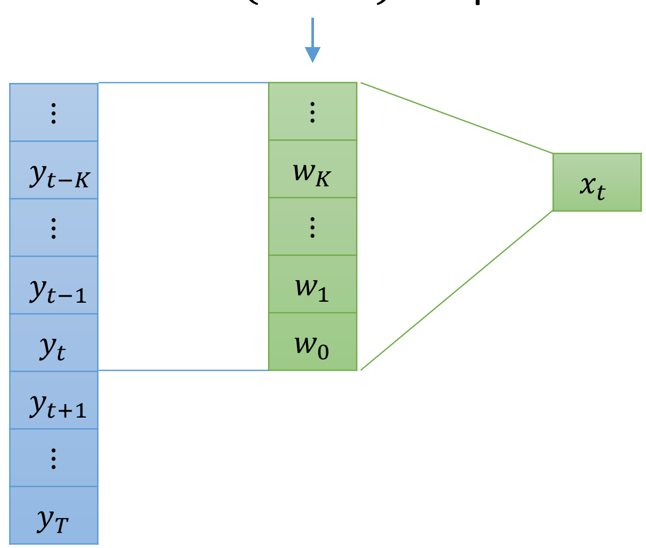
Introduction

- Statistical long memory models
  - A fractionally integrated processes $\{y_t\}$ is defined as

$$(1-B)^d y_t = x_t \iff y_t = (1-B)^{-d} x_t$$

If $x_t \sim \mathrm{ARMA}$, $y_t \sim$ fractionally integrated $\mathrm{ARMA} = \mathrm{ARFIMA}$

$(1-B)^d$ represents an infinitely long filter



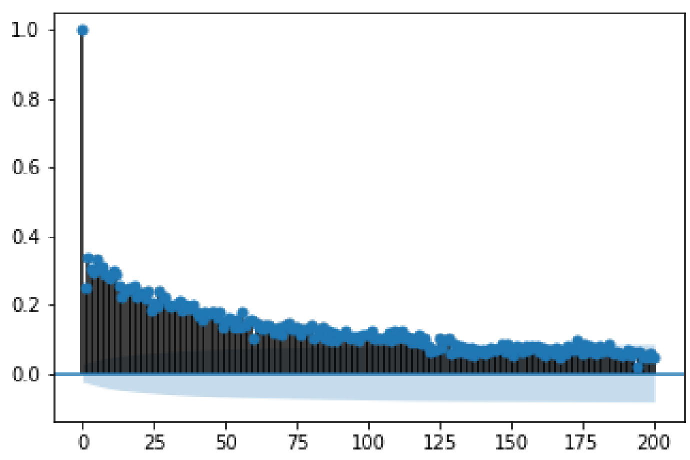
	⋮
w_K	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	⋮
w_2	$= \frac{-d(1-d)}{2!}$
w_1	$= -d$
w_0	$= 1$

filter weights fully determined by d

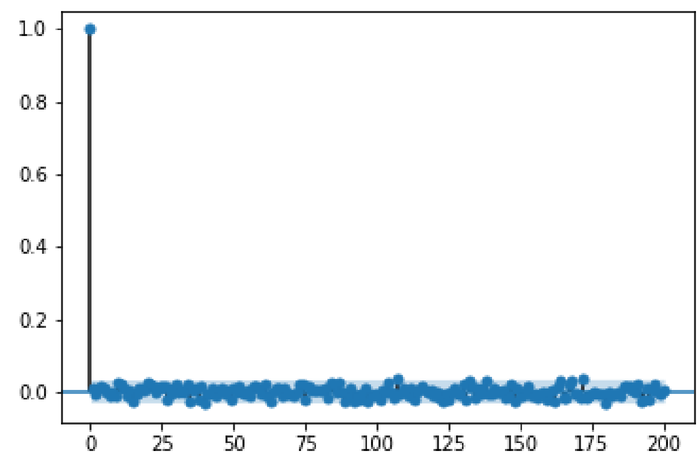
Introduction

- Long memory datasets
  - A statistical but visual check is to look at the sample plot of the autocorrelation function $\rho(k) = \mathrm{Corr}(X_t, X_{t-k})$,
    i.e., sample ACF plot

ACF Plot of Long Memory Time Series

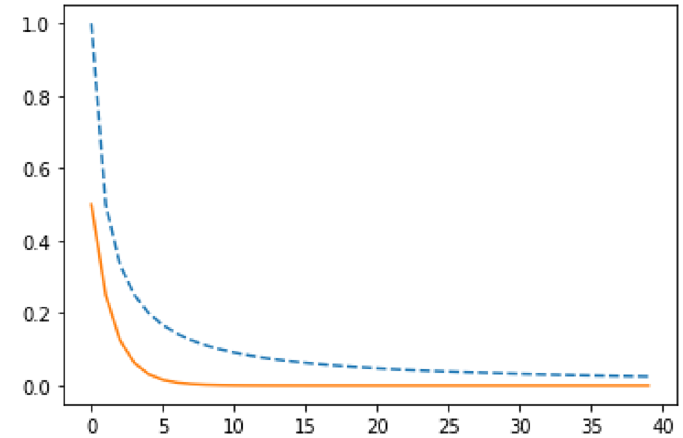


ACF Plot of Short Memory Time Series



Memory Properties of Recurrent Networks

- The statistical definition of Long Memory
  - For a second-order stationary univariate process $\{X_t\}$, it has
    (a) long memory, or (b) short memory if
    (a) $\sum_{k=-\infty}^{\infty} \rho(k) = \infty$, or (b) $\sum_{k=-\infty}^{\infty} \rho(k) = C < \infty$
  - E.g. polynomial decay (blue dashed line)
    $\rho(k) \sim |k|^{-1}, \sum_{k=-\infty}^{\infty} \rho(k) = \infty$
  - E.g. exponential decay (orange line)
    $\rho(k) \sim 2^{-|k|}, \sum_{k=-\infty}^{\infty} \rho(k) = C$



Memory Properties of Recurrent Networks

- Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.

  - Target sequence: $\{y_t\}$
  - General hidden states: $\{s_t\}$
  - Random error: $\{\varepsilon_t\}$
  - Transition function: $\mathcal{M}$

  - A recurrent network with Markovian updates is written as
$$\begin{pmatrix} y_t \\ s_t \end{pmatrix} = \mathcal{M}(y_{t-1}, s_{t-1}) + \begin{pmatrix} \varepsilon_t \\ 0 \end{pmatrix}$$
  - RNN and LSTM belongs to recurrent networks with Markovian updates!

Memory Properties of Recurrent Networks

- Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.
    - The sufficient conditions are met most of the time! (see Corollary 1 & 2 in the paper)

Table 1. Restrictions on weights such that the RNN process is geometrically ergodic.

Output function g	Activation function σ: identity or ReLU	Activation function σ: sigmoid or tanh
identity	$\lvert w_{zh}w_{hh}\rvert \leq a$, $\lvert w_{zh}w_{hy}\rvert \leq a$, $\lvert w_{hh}\rvert \leq a$, $\lvert w_{hy}\rvert \leq a$	No
sigmoid	$\lvert w_{hh}\rvert \leq a$, $\lvert w_{hy}\rvert \leq a$	No
softmax	$\lvert w_{hh}\rvert \leq a$, $\lvert w_{hy}\rvert \leq a$	No

Memory Properties of Recurrent Networks

- Assuming no exogenous inputs, we prove sufficient conditions for a recurrent network with Markovian updates to have short memory.
  - The sufficient conditions are met most of the time! (see Corollary 1 & 2 in the paper)

Table 4. Application of Theorem 1 to specific LSTMs.

		Activation function σ	
		ReLU or identity	sigmoid or tanh
Output function g	identity	$\lvert w_{oh}\rvert + \lvert w_{ih}\rvert + \lvert w_{zh}w_{oh}\rvert \le a$, $\lvert w_{oy}\rvert + \lvert w_{iy}\rvert + \lvert w_{zh}w_{oy}\rvert \le a$, $\lvert w_{fh}v + w_{fy}u + b_f\rvert \le a$	No
	sigmoid	$\lvert w_{oh}\rvert + \lvert w_{ih}\rvert \le a$, $\lvert w_{oy}\rvert + \lvert w_{iy}\rvert \le a$, $\lvert w_{fh}v + w_{fy}u + b_f\rvert \le a$	$\lvert\sigma(w_{fh} + w_{fy} + b_f)\rvert \le a$
	softmax	$\lvert w_{oh}\rvert + \lvert w_{ih}\rvert \le a$, $\lvert w_{oy}\rvert + \lvert w_{iy}\rvert \le a$, $\lvert w_{fh}v + w_{fy}u + b_f\rvert \le a$	$\lvert\sigma(w_{fh} + w_{fy} + b_f)\rvert \le a$

Long Memory Recurrent Networks

- We propose a new definition of long memory recurrent networks, allowing exogenous inputs.

  - Suppose we manage to write the target sequence $\{y_t\}$ as a linear function of the network inputs $\{x_t\}$,

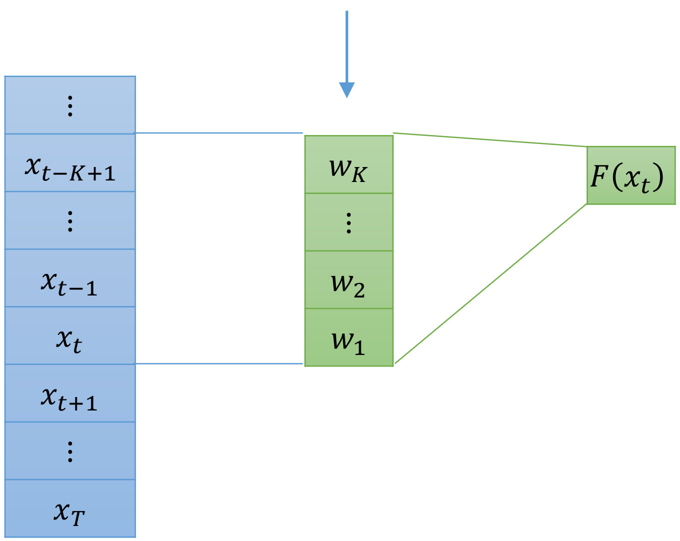
$$y_t = \sum_{k=0}^{\infty} A_k x_{t-k} + \varepsilon_t$$

  - A neural network has long memory if elements of A_k decay slowly as $k \to \infty$.
  - This definition is closely connected to its statistics counterpart.
  - Possible extensions to nonlinear networks are discussed in the paper.

Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - Long-term information cannot be stably stored in the hidden states of a recurrent network with Markovian updates.
  - A long memory filter is added to RNN at the input or LSTM at the cell states, to pass distant information to current hidden units.

truncated $(1-B)^d$ as the long memory filter



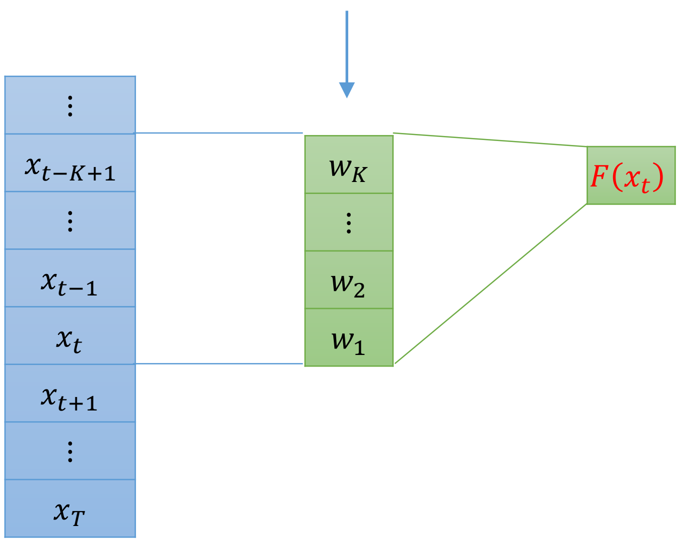
$\boldsymbol{w_K}$	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	$\vdots$
$\boldsymbol{w_2}$	$= \frac{-d(1-d)}{2!}$
$\boldsymbol{w_1}$	$= -d$

K filter weights fully determined by d

Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented RNN, x_t and $F(x_t)$ are parallel inputs
    - Normal hidden units: $h_t = \tanh(W_h[h_{t-1},\ x_t] + b_h)$
    - Long memory hidden: $m_t = \tanh(W_m[m_{t-1},\ F(x_t)] + b_m)$
    - Output: $z_t = g(W_z[h_t, m_t] + b_z)$

truncated $(1-B)^d$ as the long memory filter



$\boldsymbol{w_K}$	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	$\vdots$
$\boldsymbol{w_2}$	$= \frac{-d(1-d)}{2!}$
$\boldsymbol{w_1}$	$= -d$

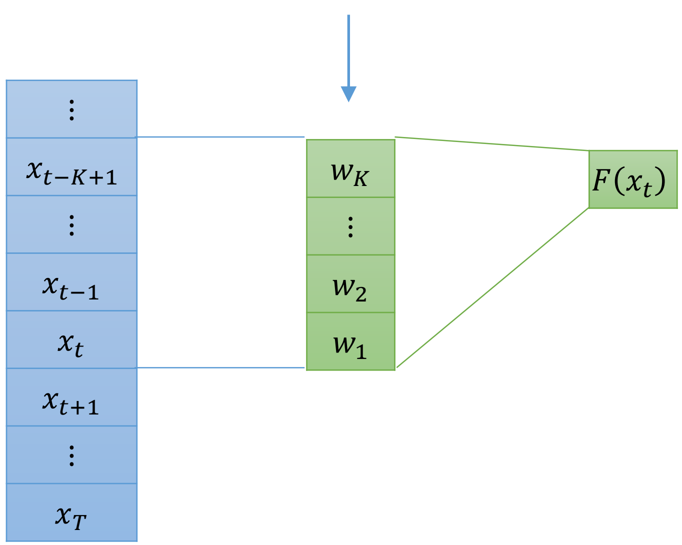
K filter weights fully determined by d

Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented RNN, the memory parameter d can be time-varying (MRNN) or constant through time (MRNNF):

$$d_t = 0.5\,\sigma(W_d[d_{t-1}, h_{t-1}, m_{t-1}, x_t] + b_d) \in (0, 0.5)$$

truncated $(1-B)^d$ as the long memory filter



w_K	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	⋮
w_2	$= \frac{-d(1-d)}{2!}$
w_1	$= -d$

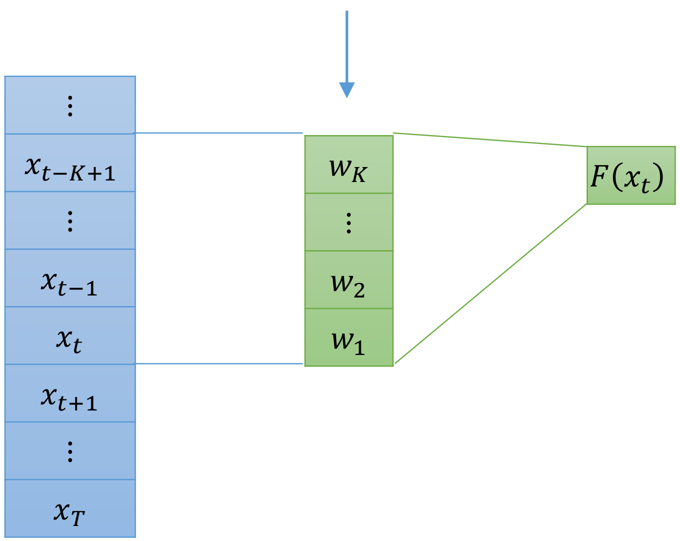
K filter weights fully determined by d

Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented RNN, the memory parameter d can be time-varying (MRNN) or constant through time (MRNNF):

$$d \quad = 0.5\,\sigma(\qquad\qquad b_d) \in (0, 0.5)$$

truncated $(1-B)^d$ as the long memory filter



w_K	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	⋮
w_2	$= \frac{-d(1-d)}{2!}$
w_1	$= -d$

K filter weights fully determined by d

Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In LSTM, the update of cell states can be viewed as a random coefficient vector AR(1) model

$$\text{(LSTM) } c_t = f_t\, c_{t-1} + i_t\, \widetilde{c_t} \quad \leftrightarrow \quad c_t = A_t c_{t-1} + \varepsilon_t \text{ (RC-VAR(1))}$$

$$\text{(LSTM) } c_t - f_t\, c_{t-1} = i_t\, \widetilde{c_t} \quad \leftrightarrow \quad c_t - A_t c_{t-1} = \varepsilon_t \text{ (RC-VAR(1))}$$

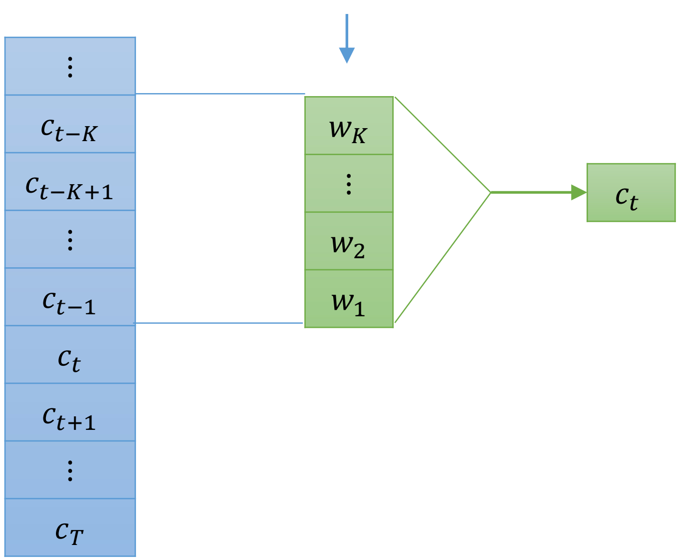
Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented LSTM, long memory filter is applied to the cell states, generalizing the RC-VAR(1) form

$$\text{(MLSTM)}\ c_t - d\, c_{t-1} - \cdots = (1-B)^d\, c_t = i_t\, \tilde{c}_t$$

$$\text{(LSTM)}\ c_t - f_t\, c_{t-1} = i_t\, \tilde{c}_t$$

truncated $(1-B)^d$ as the long memory filter



w_K	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	⋮
w_2	$= \frac{-d(1-d)}{2!}$
w_1	$= -d$

K filter weights fully determined by d

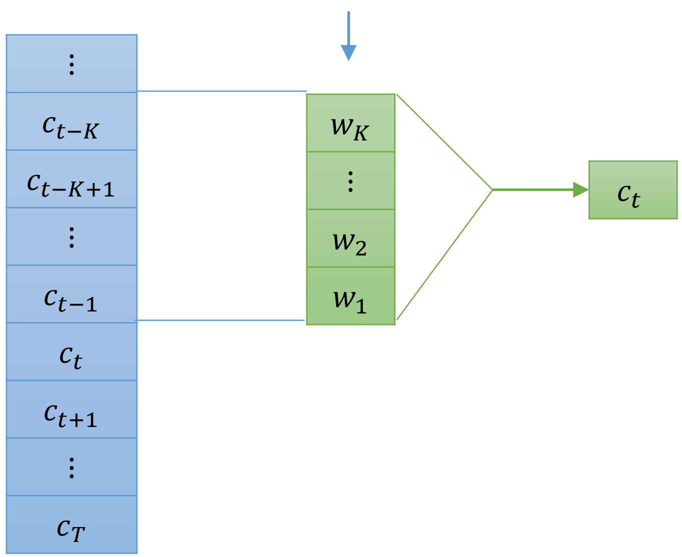
Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented LSTM, the memory parameter d can be time-varying (MLSTM) or constant through time (MLSTMF):

(MLSTM) $(1-B)^{d}\, c_t = i_t\, \tilde{c}_t, d_t = 0.5\, \sigma(W_d[d_{t-1}, h_{t-1}, x_t] + b_d)$

(LSTM) $c_t - f_t\, c_{t-1} = i_t\, \tilde{c}_t,\ f_t = \sigma\big(W_d[h_{t-1}, x_t] + b_f\big)$

truncated $(1-B)^{d}$ as the long memory filter



$\boldsymbol{w_K}$	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	⋮
$\boldsymbol{w_2}$	$= \frac{-d(1-d)}{2!}$
$\boldsymbol{w_1}$	$= -d$

K filter weights fully determined by d

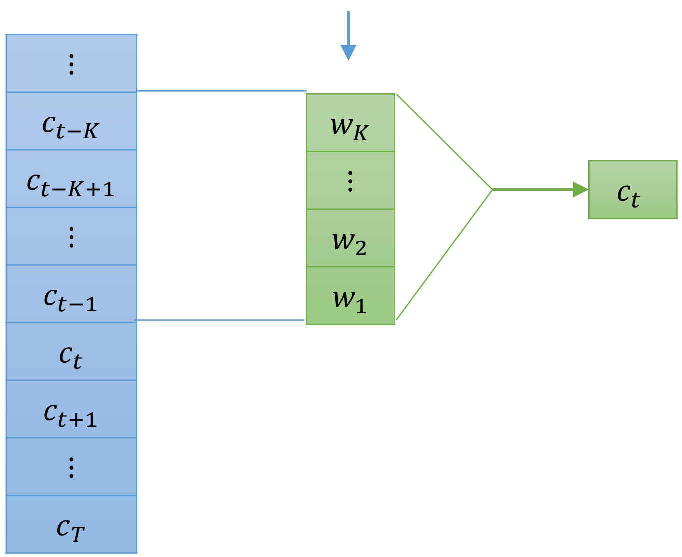
Long Memory Recurrent Networks

- We explore theory-guided applications: MRNN and MLSTM.
  - In Memory-augmented LSTM, the memory parameter d can be time-varying (MLSTM) or constant through time (MLSTMF):

$$\text{(MLSTM)}\ (1-B)^d\, c_t = i_t\, \tilde{c}_t,\ d\ = 0.5\ \sigma(\qquad\qquad\qquad b_d)$$

$$\text{(LSTM)}\ c_t - f_t\, c_{t-1} = i_t\, \tilde{c}_t,\ f_t = \sigma\big(W_d[h_{t-1}, x_t] + b_f\big)$$

truncated $(1-B)^d$ as the long memory filter



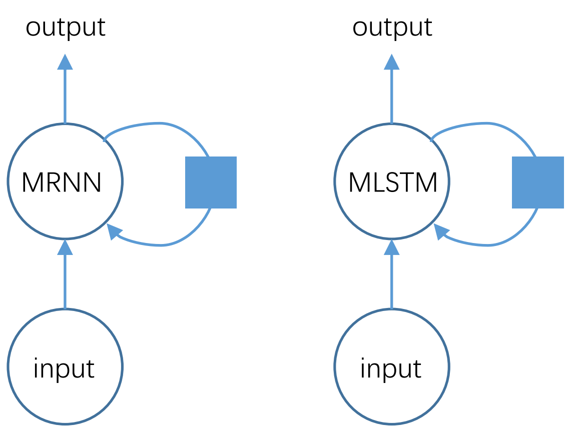
w_K	$= \prod_{j=0}^{K-1} \frac{j-d}{j+1} = \frac{-d(1-d)\cdots(K-2-d)}{(K-1)!}$
	$\vdots$
w_2	$= \frac{-d(1-d)}{2!}$
w_1	$= -d$

K filter weights fully determined by d

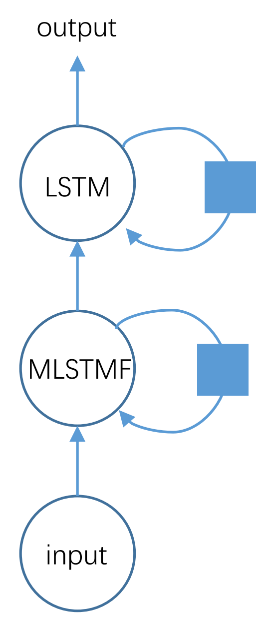
Experiments

- We conduct numerical studies to illustrate the advantages of proposed models.
  - They can be used alone or merge into current network structures!

e.g. proposed cell structure replacing the hidden units in RNN/LSTM



e.g. a two layer network with one layer of MLSTM cell + one layer of LSTM cell

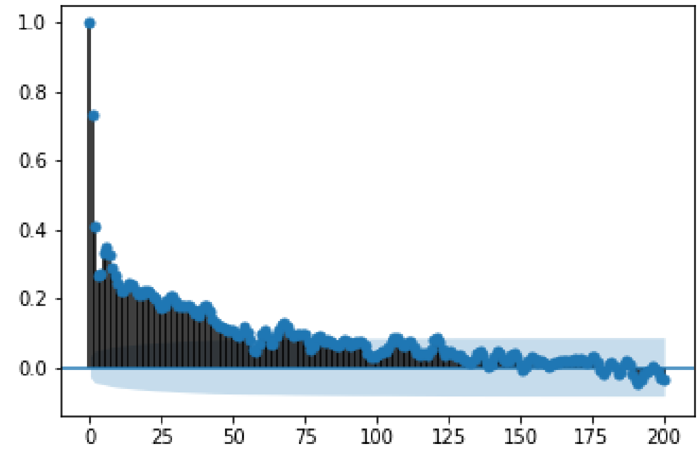


Experiments

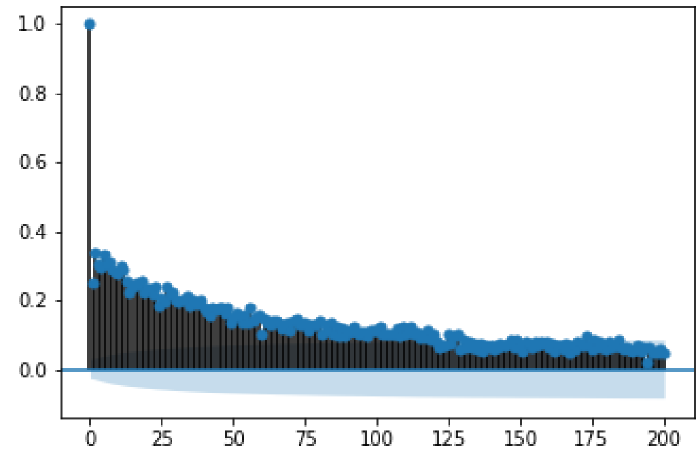
- Datasets

Time Series Forecasting

- Synthetic series
  - ARFIMA sequence



- Real data
  - DJI financial returns
  - Traffic volume
  - Tree ring measures



- Source:
  - Yahoo Finance
  - UCI machine learning repository
  - R package: tsdl

Experiments

- Datasets

Paper Reviews Classification

- Spanish paper reviews
- Evaluated by a five-point scale:
  - -2, -1, 0, 1, 2
  - Source:
    - UCI machine learning repository

{
"evaluation": "1",
"text": "- El artículo aborda un problema contingente y muy relevante, e incluye tanto un diagnóstico nacional de uso de buenas prácticas como una solución (buenas prácticas concretas). - El lenguaje es adecuado. - El artículo se siente como la concatenación de tres artículos diferentes: (1) resultados de una encuesta, (2) buenas prácticas de seguridad, (3) incorporación de buenas prácticas. - El orden de las secciones sería mejor si refleja este orden (la versión revisada es #2, #1, #3). - El artículo no tiene validación de ningún tipo, ni siquiera por evaluación de expertos.",
...
},

Experiments

- Experiment highlights

Time Series Forecasting

Table 2. Overall performance in terms of RMSE. Average RMSE and the standard deviation (in brackets) are reported. The best result is highlighted in **bold**.

	ARFIMA	DJI (x100)	Traffic	Tree
RNN	1.1620 (0.1980)	0.2605 (0.0171)	336.44 (10.401)	0.2871 (0.0086)
RNN2	1.1630 (0.1820)	0.2521 (0.0112)	336.32 (10.182)	0.2855 (0.0077)
RWA	1.6840 (0.0050)	0.2689 (0.0095)	346.62 (1.410)	0.3048 (0.0001)
MIST	1.1390 (0.1832)	0.2604 (0.0154)	358.09 (16.270)	0.2883 (0.0091)
MRNNF	1.1010 (0.1000)	**0.2472** (0.0109)	**333.36** (8.453)	0.2822 (0.0048)
MRNN	**1.0880** (0.1140)	0.2487 (0.0105)	333.72 (10.157)	**0.2818** (0.0053)
LSTM	1.1340 (0.1200)	0.2492 (0.0128)	337.60 (8.146)	0.2833 (0.0070)
MLSTMF	1.1580 (0.1660)	0.2540 (0.0139)	337.78 (9.020)	0.2859 (0.0082)
MLSTM	1.1490 (0.1660)	0.2531 (0.0130)	337.83 (9.440)	0.2859 (0.0083)

Paper Reviews Classification

Table 5. Overall performance on Paper Reviews in terms of accuracy, precision, recall and cross-entropy loss (CEloss).

	Accuracy	Precision	Recall	CEloss
RNN	0.2836 (0.0348)	0.1786 (0.0606)	0.2248 (0.0350)	1.5787 (0.0348)
LSTM	0.3021 (0.0468)	0.1724 (0.0697)	0.2274 (0.0332)	1.5752 (0.0189)
MRNNF50	0.3096 (0.0373)	0.1692 (0.0839)	0.2224 (0.0428)	1.5704 (0.0328)
MLSTMF50	**0.3110** (0.0204)	**0.2254** (0.0707)	**0.2594** (0.0262)	**1.4758** (0.0218)

Table 6. Best performance of the models on Paper Reviews.

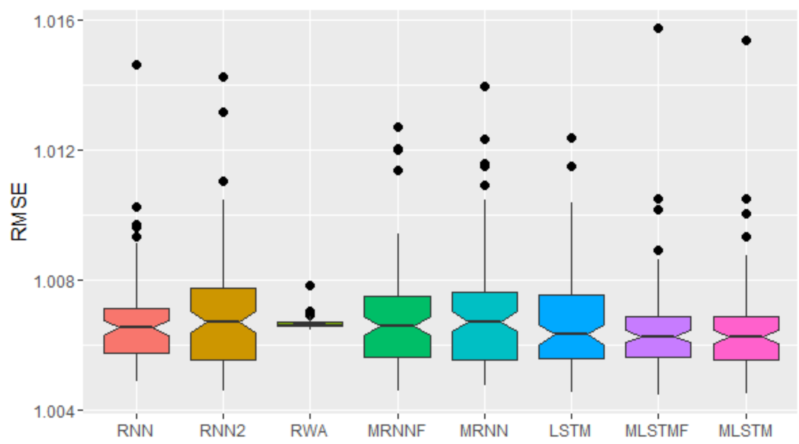
	Accuracy	Precision	Recall	CEloss
RNN	0.3600	0.3951	0.3093	1.5204
LSTM	0.3800	0.4304	0.3225	1.5512
MRNNF50	**0.4000**	0.3992	0.3178	1.5209
MLSTMF50	0.3600	**0.4621**	**0.3596**	**1.4489**

Experiments

- Additional experiments

Performance on short memory dataset

- Synthetic dataset:
  - RNN sequence



Hyperparameter K

- K = 25, 50, 75, 100 tested.
- For MRNN(F), we recommend K = 100
- For MLSTM(F), we recommend K = 25

Thank you for listening!

- Full Paper: https://arxiv.org/abs/2006.03860
- Code Preview: https://github.com/Gladys-Zhao/mRNN-mLSTM

Full paper at arXiv

Code preview at GitHub