Deep Learning, Dynamics and Control

II: Deep Learning for Sequence Modelling

Qianxiao Li
Department of Mathematics
Institute for Functional Intelligent Materials
blog.nus.edu.sg/qianxiaoli

Dynamical systems and Semi-algebraic geometry
Interactions with Optimization and Deep Learning
Dalat, Vietnam
17–21 Jul 2023

NUS
National University of Singapore

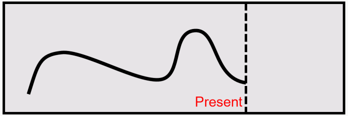
Overview

1. Recurrent Neural Networks
2. Temporal Convolutional Networks
3. Encoder-Decoder Networks

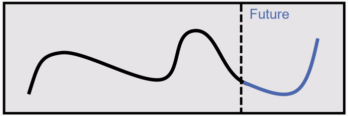
Sequence Modelling Applications

Time-Series/Dynamics Prediction

Historical Data



Prediction



Machine Translation

Language 1

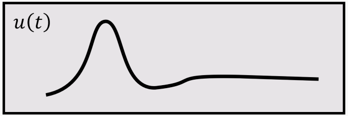
Google Translate is a multilingual neural machine translation service

Language 2

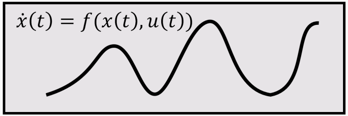
谷歌翻译是一种多语言神经机器翻译服务

Control Systems

Control

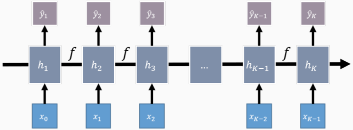


Response

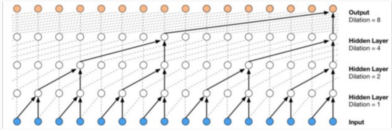


Machine Learning Architectures for Sequence Modelling

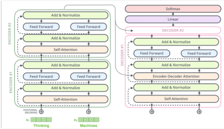
RNN



WaveNet (CNN)

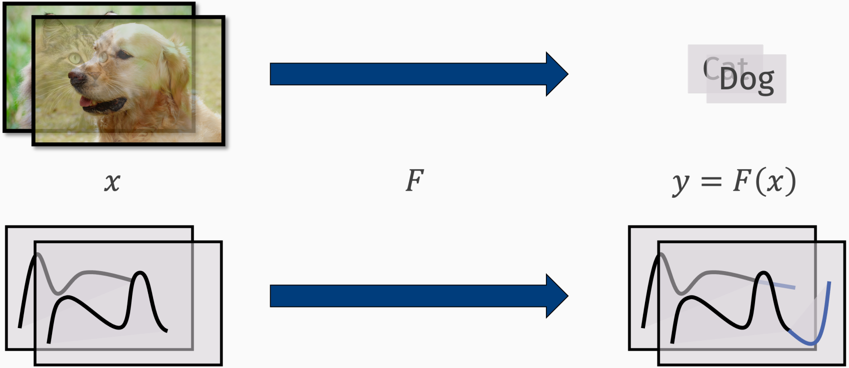


Transformers

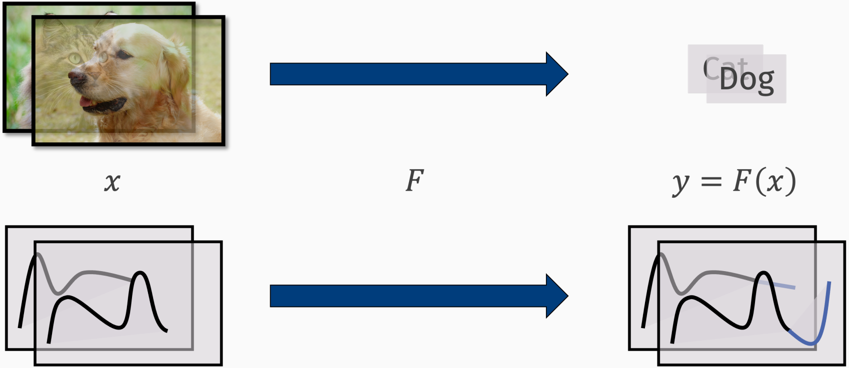


General question: How are they different? When should we use which?

Supervised Learning



Supervised Learning



Goal: Learn/approximate target F

Static setting

(input)	$x \in \mathcal{X} = \mathbb{R}^d$
(output)	$y \in \mathcal{Y} = \mathbb{R}^n$
(target)	$y = F(x)$

Modelling Static vs Dynamic Relationships

Static setting

(input) $x \in \mathcal{X} = \mathbb{R}^d$

(output) $y \in \mathcal{Y} = \mathbb{R}^n$

(target) $y = F(x)$

Dynamic setting

(input) $\boldsymbol{x} = \{x_t \in \mathbb{R}^d\} \in \mathcal{X}$

(output) $\boldsymbol{y} = \{y_t \in \mathbb{R}^n\} \in \mathcal{Y}$

(target) $y_t = H_t(\boldsymbol{x}) \quad \forall \quad t$

Modelling Static vs Dynamic Relationships

Static setting

(input)	$x \in \mathcal{X} = \mathbb{R}^d$
(output)	$y \in \mathcal{Y} = \mathbb{R}^n$
(target)	$y = F(x)$

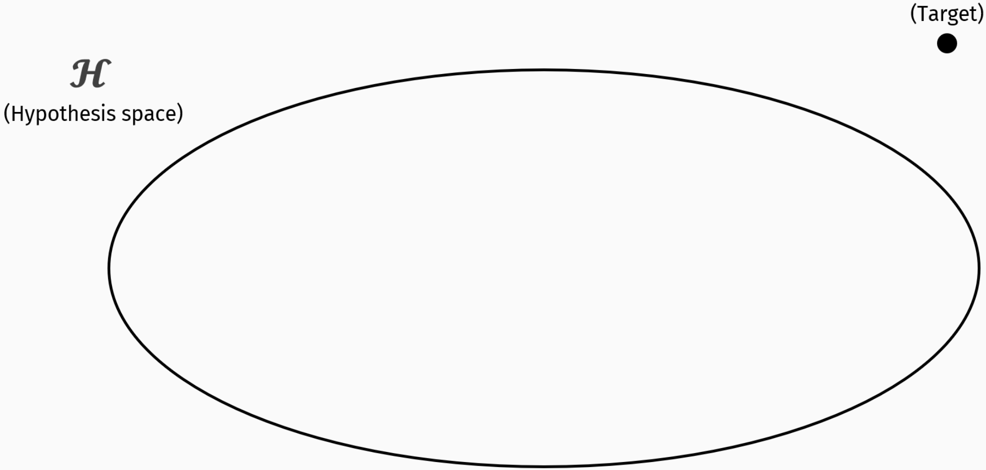
Dynamic setting

(input)	$\boldsymbol{x} = \{x_t \in \mathbb{R}^d\} \in \mathcal{X}$
(output)	$\boldsymbol{y} = \{y_t \in \mathbb{R}^n\} \in \mathcal{Y}$
(target)	$y_t = H_t(\boldsymbol{x}) \quad \forall \quad t$

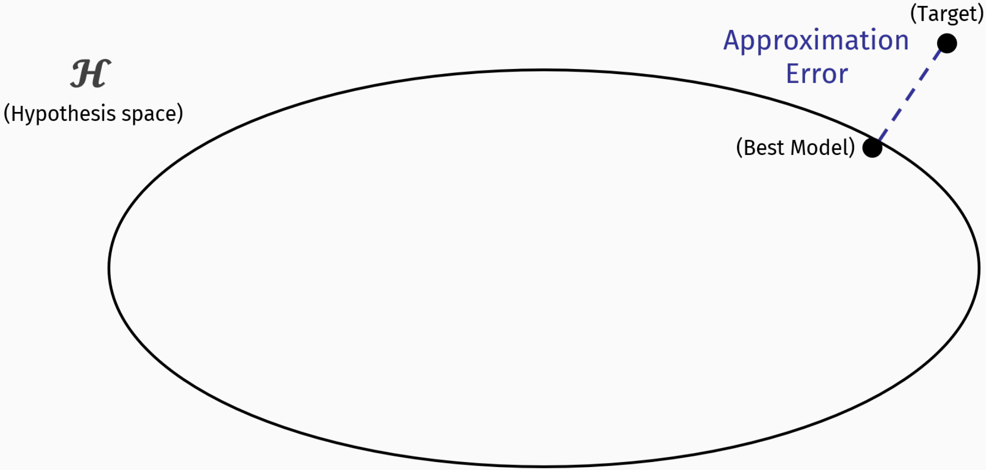
Goal of supervised learning

- Static: learn/approximate the target F
- Dynamic: learn/approximate the target $\boldsymbol{H} = \{H_t\}$

The Problem of Approximation



The Problem of Approximation



Example: Approximation by Trigonometric Polynomials

Consider

- $\mathcal{C} = C^{\alpha}_{\text{per}}([0, 2\pi], \mathbb{R})$ (Periodic C^{α} functions)
- $\mathcal{H} = \cup_{m \in \mathbb{N}_+} \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R} \right\}.$

Example: Approximation by Trigonometric Polynomials

Consider

- $\mathcal{C} = C^\alpha_{\text{per}}([0, 2\pi], \mathbb{R})$ (Periodic C^α functions)
- $\mathcal{H} = \cup_{m \in \mathbb{N}_+} \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R} \right\}.$

Then, the Stone-Weierstrass theorem implies density

> For any $H \in \mathcal{C}$ and $\epsilon > 0$, there exists $\widehat{H} \in \mathcal{H}$ with $\|H - \widehat{H}\| \leq \epsilon$.

Example: Approximation by Trigonometric Polynomials

We can also ask a finer question: rate of approximation

N. I. Achieser, Theory of Approximation. Courier Corporation, 5, 2013

Example: Approximation by Trigonometric Polynomials

We can also ask a finer question: rate of approximation

Given an approximation budget m, consider

$$\mathcal{H}^m = \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R}, m \geq 1 \right\}.$$

What is the best possible approximation error given budget m?

N. I. Achieser, Theory of Approximation. Courier Corporation, 5, 2013

Example: Approximation by Trigonometric Polynomials

We can also ask a finer question: rate of approximation

Given an approximation budget m, consider

$$\mathcal{H}^m = \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R}, m \geq 1 \right\}.$$

What is the best possible approximation error given budget m?

Jackson proved the following estimate

$$\boxed{\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq \frac{C_\alpha \max_{0 \leq r \leq \alpha} \|H^{(r)}\|}{m^\alpha},}$$

N. I. Achieser, Theory of Approximation. Courier Corporation, 5, 2013

Example: Approximation by Trigonometric Polynomials

We can also ask the reverse question: suppose H can be efficiently approximated (e.g. rate $m^{-\alpha}$) by $\mathcal{H}^m$. What can we say about H?

N. I. Achieser, Theory of Approximation. Courier Corporation, 5, 2013

Example: Approximation by Trigonometric Polynomials

We can also ask the reverse question: suppose H can be efficiently approximated (e.g. rate $m^{-\alpha}$) by $\mathcal{H}^m$. What can we say about H?

Bernstein proved the following result

$$\boxed{\inf_{\widehat{H}\in\mathcal{H}^m} \|H - \widehat{H}\| \leq \frac{c}{m^\alpha}, \forall m \geq 1 \quad \Longrightarrow \quad H \in \mathcal{C} = C^\alpha_{\text{per}}([0, 2\pi])}$$

N. I. Achieser, Theory of Approximation. Courier Corporation, 5, 2013

Insight on trigonometric polynomial approximation

Efficient approximation $\iff$ Smoothness (small gradient norm)

Three Types of Results

Given a hypothesis space $\mathcal{H}$ and a target space $\mathcal{C}$, we seek three types of results

Density-type

For all $H \in \mathcal{C}$

$$\inf_{\widehat{H} \in \mathcal{H}} \|H - \widehat{H}\| = 0$$

Jackson-type

For all $H \in \mathcal{C}$

$$\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq C(H, m)$$

Bernstein-type

If for all $m \geq 1$

$$\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq C(H, m),$$

then $H \in \mathcal{C}$

Consider an input sequence

$$\boldsymbol{x} = \{x(t) : t \in \mathcal{T}\}, \quad x(t) \in \mathbb{R}^d \qquad \text{(Index set } \mathcal{T} \subset \mathbb{R} \text{ or } \mathcal{T} \subset \mathbb{Z}\text{)}$$

Sequence Modelling as an Approximation Problem

Consider an input sequence

$$\boldsymbol{x} = \{x(t) : t \in \mathcal{T}\}, \quad x(t) \in \mathbb{R}^d \qquad \text{(Index set } \mathcal{T} \subset \mathbb{R} \text{ or } \mathcal{T} \subset \mathbb{Z}\text{)}$$

and corresponding output sequence $\boldsymbol{y} = \{y(t) \in \mathbb{R} : t \in \mathcal{T}\}$ related by

$$y(t) = H_t(\boldsymbol{x}), \qquad t \in \mathcal{T}$$

Sequence Modelling as an Approximation Problem

Consider an input sequence

$$\boldsymbol{x} = \{x(t) : t \in \mathcal{T}\}, \quad x(t) \in \mathbb{R}^d \qquad \text{(Index set } \mathcal{T} \subset \mathbb{R} \text{ or } \mathcal{T} \subset \mathbb{Z}\text{)}$$

and corresponding output sequence $\boldsymbol{y} = \{y(t) \in \mathbb{R} : t \in \mathcal{T}\}$ related by

$$y(t) = H_t(\boldsymbol{x}), \qquad t \in \mathcal{T}$$

The approximation target is the functional sequence

$$\boldsymbol{H} = \{H(t) \equiv H_t : t \in \mathcal{T}\}$$

Our goal is to derive Density-type, Jackson-type and Bernstein-type results for

- $\mathcal{C} \to$ suitable classes of functional sequences

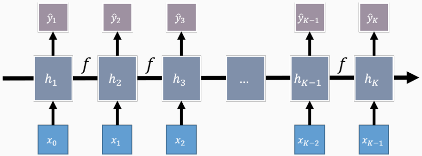
$$\boldsymbol{H} = \{H(t) \equiv H_t : t \in \mathcal{T}\}$$

- $\mathcal{H} \to$ RNNs, CNNs/WaveNets, Encoder-Decoders, Transformers

Recurrent Neural Networks

The recurrent neural network (RNN) architecture

$$\begin{aligned} h(t+1) &= \sigma(Wh(t) + Ux(t) + b) \\ \hat{y}(t) &= c^\top h(t) \quad t \in \mathbb{Z} \end{aligned} \tag{1}$$

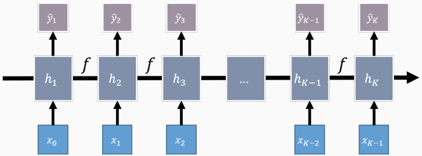


D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, 6088 1986

The Recurrent Neural Network Hypothesis Space

The recurrent neural network (RNN) architecture

$$
\begin{aligned}
h(t+1) &= \sigma(Wh(t) + Ux(t) + b) \\
\widehat{y}(t) &= c^\top h(t) \quad t \in \mathbb{Z}
\end{aligned}
\tag{1}
$$



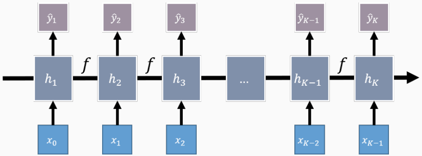
This gives rise to the RNN hypothesis space

$$
\mathcal{H}_{\text{RNN}} = \bigcup_{m \geq 1} \mathcal{H}^m_{\text{RNN}} \qquad \mathcal{H}^m_{\text{RNN}} = \left\{ \begin{aligned} \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = c^\top h(t), \boldsymbol{h} \text{ follows } Eq.\ (1) \text{ with} \\ W \in \mathbb{R}^{m\times m}, U \in \mathbb{R}^{m\times d}, b \in \mathbb{R}^m, c \in \mathbb{R}^m \end{aligned} \right\}
$$

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, 6088 1986

The recurrent neural network (RNN) architecture

$$\begin{aligned} h(t+1) &= \sigma(Wh(t) + Ux(t) + b) \\ \widehat{y}(t) &= c^\top h(t) \quad t \in \mathbb{Z} \end{aligned} \tag{1}$$



This gives rise to the RNN hypothesis space

$$\mathcal{H}_{\text{RNN}} = \bigcup_{m\geq 1} \mathcal{H}^m_{\text{RNN}} \qquad \mathcal{H}^m_{\text{RNN}} = \left\{ \begin{matrix} \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = c^\top h(t), \boldsymbol{h} \text{ follows } Eq.\ (1) \text{ with} \\ W \in \mathbb{R}^{m\times m}, U \in \mathbb{R}^{m\times d}, b \in \mathbb{R}^m, c \in \mathbb{R}^m \end{matrix} \right\}$$

Continuous time index variant

$$h(t+1) = \sigma(Wh(t) + Ux(t) + b) \qquad \to \qquad \dot{h}(t) = \sigma(Wh(t) + Ux(t) + b),$$

D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, no. 6088, 6088 1986

Density-type Results for RNN

Early results focus on target functional sequences that are themselves generated by hidden dynamical systems

$$C = \left\{ \boldsymbol{x} \mapsto \boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{y} \quad \text{with} \quad \begin{array}{ll} \dot{h}(t) = f(h(t), x(t)), & h(t) \in \mathbb{R}^n \\ y(t) = g(h(t)), & h(-\infty) = 0 \end{array} \right\}$$

E. Sontag, "Neural Nets As Systems Models And Controllers," in Proc. Seventh Yale Workshop on Adaptive and Learning Systems, 1992

T. Chow and Xiao-Dong Li, "Modeling of continuous time dynamical systems with input by recurrent neural networks," IEEE Trans. Circuits Syst. I, vol. 47, no. 4, 2000

A. M. Schäfer and H. G. Zimmermann, "Recurrent Neural Networks Are Universal Approximators," in Artificial Neural Networks – ICANN 2006, S. D. Kollias, A. Stafylopatis, W. Duch, and E. Oja, Eds., Berlin, Heidelberg: Springer, 2006

Density-type Results for RNN

Early results focus on target functional sequences that are themselves generated by hidden dynamical systems

$$C = \left\{ \boldsymbol{x} \mapsto \boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{y} \quad \text{with} \quad \begin{aligned} \dot{h}(t) &= f(h(t), x(t)), & h(t) &\in \mathbb{R}^n \\ y(t) &= g(h(t)), & h(-\infty) &= 0 \end{aligned} \right\}$$

Density on bounded intervals $t \in [0, T]$ follows from the density of fully connected networks

$$(h, x) \mapsto f(h, x) \approx (h_1, x) \mapsto \sigma(W(h_1, h_2)^\top + Ux + b)$$

E. Sontag, "Neural Nets As Systems Models And Controllers," in Proc. Seventh Yale Workshop on Adaptive and Learning Systems, 1992

T. Chow and Xiao-Dong Li, "Modeling of continuous time dynamical systems with input by recurrent neural networks," IEEE Trans. Circuits Syst. I, vol. 47, no. 4, 2000

A. M. Schäfer and H. G. Zimmermann, "Recurrent Neural Networks Are Universal Approximators," in Artificial Neural Networks – ICANN 2006, S. D. Kollias, A. Stafylopatis, W. Duch, and E. Oja, Eds., Berlin, Heidelberg: Springer, 2006

Density-type Results for RNN on Unbounded Index Sets

To handle unbounded sets, one usually resorts to some localization argument

---

L. Grigoryeva and J.-P. Ortega, "Echo state networks are universal," Neural Networks, vol. 108, 2018

J. Hanson and M. Raginsky, "Universal Simulation of Stable Dynamical Systems by Recurrent Neural Nets," Proceedings of the 2nd Conference on Learning for Dynamics and Control, vol. 120, 8, 2020

Density-type Results for RNN on Unbounded Index Sets

To handle unbounded sets, one usually resorts to some localization argument

- Fading memory property. For some decreasing $\boldsymbol{w} : \mathbb{R}_+ \to \mathbb{R}$, assume

$$|H_t(\boldsymbol{x}_1) - H_t(\boldsymbol{x}_2)| < \epsilon \text{ whenever } \sup_{s \in (-\infty, t]} |x_1(s) - x_2(s)| w(t-s) < \delta.$$

L. Grigoryeva and J.-P. Ortega, "Echo state networks are universal," Neural Networks, vol. 108, 2018

J. Hanson and M. Raginsky, "Universal Simulation of Stable Dynamical Systems by Recurrent Neural Nets," Proceedings of the 2nd Conference on Learning for Dynamics and Control, vol. 120, 8, 2020

Density-type Results for RNN on Unbounded Index Sets

To handle unbounded sets, one usually resorts to some localization argument

- Fading memory property. For some decreasing $\boldsymbol{w} : \mathbb{R}_+ \to \mathbb{R}$, assume

$$|H_t(\boldsymbol{x}_1) - H_t(\boldsymbol{x}_2)| < \epsilon \text{ whenever } \sup_{s \in (-\infty, t]} |x_1(s) - x_2(s)| w(t-s) < \delta.$$

- Uniformly asymptotically incrementally stable (for $f - g$ type).

L. Grigoryeva and J.-P. Ortega, "Echo state networks are universal," Neural Networks, vol. 108, 2018

J. Hanson and M. Raginsky, "Universal Simulation of Stable Dynamical Systems by Recurrent Neural Nets," Proceedings of the 2nd Conference on Learning for Dynamics and Control, vol. 120, 8, 2020

Jackson-type Results for RNN

Results in the non-linear setting:

- Time truncation (fading memory) $+$ Barron-type assumption on truncated functional
- Time truncation (stability) $+$ Barron-type assumption on f, g

These give Jackson-type estimates of order $m^{-1/2}$

L. Gonon, L. Grigoryeva, and J.-P. Ortega, “Approximation Bounds for Random Neural Networks and Reservoir Systems,” 16, 2021

J. Hanson and M. Raginsky, “Universal Simulation of Stable Dynamical Systems by Recurrent Neural Nets,” Proceedings of the 2nd Conference on Learning for Dynamics and Control, vol. 120, 8, 2020

Empirically, it is found RNN performs poorly when modelling "long-term memory"

A precise investigation of this requires operating directly on unbounded index set and quantifying memory effects

The Linear RNN Hypothesis Space

We analyze the linear case where $\sigma(z) = z$, we have the dynamics

$$
\begin{aligned}
\hat{y}(t) &= c^\top h(t), \\
\dot{h}(t) &= Wh(t) + Ux(t).
\end{aligned}
\quad \text{where} \quad
\begin{aligned}
&h(t) \in \mathbb{R}^m && \text{(hidden state)} \\
&W \in \mathbb{R}^{m \times m} && \text{(Recurrent Kernel)} \\
&U \in \mathbb{R}^{m \times d} && \text{(Input Kernel)} \\
&c \in \mathbb{R}^m && \text{(Output layer weights)}
\end{aligned}
$$

The Linear RNN Hypothesis Space

We analyze the linear case where $\sigma(z) = z$, we have the dynamics

$$\hat{y}(t) = c^\top h(t), \qquad \dot{h}(t) = Wh(t) + Ux(t).$$

where

$$\begin{aligned}
&h(t) \in \mathbb{R}^m && \text{(hidden state)}\\
&W \in \mathbb{R}^{m\times m} && \text{(Recurrent Kernel)}\\
&U \in \mathbb{R}^{m\times d} && \text{(Input Kernel)}\\
&c \in \mathbb{R}^m && \text{(Output layer weights)}
\end{aligned}$$

This gives rise to the (stable) linear RNN hypothesis space

$$\mathcal{H}_{\text{L-RNN}} = \cup_{m\geq 1} \underbrace{\left\{ \{\hat{H}_t(\mathbf{x}) = \int_0^\infty c^\top e^{Ws} Ux(t-s)ds\}, W \in \mathcal{W}_m, U \in \mathbb{R}^{m\times d}, c \in \mathbb{R}^m \right\}}_{\mathcal{H}^m_{\text{L-RNN}}}$$

$$\mathcal{W}_m = \{W \in \mathbb{R}^{m\times m} : \text{eigenvalues of } W \text{ have negative real parts (Hurwitz)}\}$$

Density of L-RNN on Unbounded Index Domains

L-RNN functional sequences: $\widehat{H}_t(\boldsymbol{x}) = \int_0^\infty c^\top e^{Ws} U x(t-s) ds$

Z. Li, J. Han, W. E, and Q. Li, "On the Curse of Memory in Recurrent Neural Networks: Approximation and Optimization Analysis,", presented at the International Conference on Learning Representations, 18, 2021

Density of L-RNN on Unbounded Index Domains

L-RNN functional sequences: $\widehat{H}_t(\boldsymbol{x}) = \int_0^\infty c^\top e^{Ws} U x(t-s) ds$

Notice that:

- Each $\hat{H}_t$ is a continuous, linear, causal functional
- The functional sequence $\widehat{\boldsymbol{H}}$ is shift-equivariant (time-homogeneous)

$$\boldsymbol{H} \circ \boldsymbol{S}_\tau = \boldsymbol{S}_\tau \circ \boldsymbol{H}, \qquad \boldsymbol{S}_\tau(\boldsymbol{x})(t) = x(t-\tau)$$

Z. Li, J. Han, W. E, and Q. Li, "On the Curse of Memory in Recurrent Neural Networks: Approximation and Optimization Analysis,", presented at the International Conference on Learning Representations, 18, 2021

Density of L-RNN on Unbounded Index Domains

L-RNN functional sequences: $\widehat{H}_t(\boldsymbol{x}) = \int_0^\infty c^\top e^{Ws} U x(t-s) ds$

Notice that:

- Each $\hat{H}_t$ is a continuous, linear, causal functional
- The functional sequence $\widehat{\boldsymbol{H}}$ is shift-equivariant (time-homogeneous)

$$\boldsymbol{H} \circ \boldsymbol{S}_\tau = \boldsymbol{S}_\tau \circ \boldsymbol{H}, \qquad \boldsymbol{S}_\tau(\boldsymbol{x})(t) = x(t-\tau)$$

It turns out that $\mathcal{H}_{\text{L-RNN}}$ is dense in any $\mathcal{C}$ satisfying the same properties!

Z. Li, J. Han, W. E, and Q. Li, "On the Curse of Memory in Recurrent Neural Networks: Approximation and Optimization Analysis,", presented at the International Conference on Learning Representations, 18, 2021

Density of L-RNN on Unbounded Index Domains

L-RNN functional sequences: $\widehat{H}_t(\boldsymbol{x}) = \int_0^\infty c^\top e^{Ws} U x(t-s) ds$

Notice that:

- Each $\hat{H}_t$ is a continuous, linear, causal functional
- The functional sequence $\widehat{\boldsymbol{H}}$ is shift-equivariant (time-homogeneous)

$$\boldsymbol{H} \circ \boldsymbol{S}_\tau = \boldsymbol{S}_\tau \circ \boldsymbol{H}, \qquad \boldsymbol{S}_\tau(\boldsymbol{x})(t) = x(t-\tau)$$

It turns out that $\mathcal{H}_{\text{L-RNN}}$ is dense in any $\mathcal{C}$ satisfying the same properties!

Main idea: Prove a general Riesz representation for $\boldsymbol{H} \in \mathcal{C}$

$$H_t(\boldsymbol{x}) = \int_0^\infty \rho(s)^\top x(t-s) ds \qquad \left[\text{Approximate } \rho(s) \text{ by } [c^\top e^{Ws} U]^\top \right]$$

Z. Li, J. Han, W. E, and Q. Li, "On the Curse of Memory in Recurrent Neural Networks: Approximation and Optimization Analysis,", presented at the International Conference on Learning Representations, 18, 2021

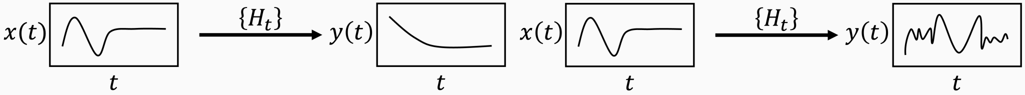
Smoothness and Memory

Approximation rates depend on appropriate complexity measures

Smoothness and Memory

Approximation rates depend on appropriate complexity measures

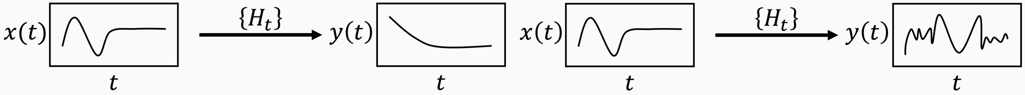
Key concepts: smoothness and memory



Smoothness and Memory

Approximation rates depend on appropriate complexity measures

Key concepts: smoothness and memory



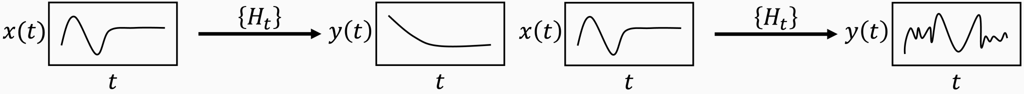
Define

- e_i, $i = 1, \ldots, d$ as the standard basis vectors in $\mathbb{R}^d$
- $\boldsymbol{e}_i$ as the constant signal $e_{i,t} = e_i \mathbb{1}_{\{t \geq 0\}}$

Smoothness and Memory

Approximation rates depend on appropriate complexity measures

Key concepts: smoothness and memory



Define

- e_i, $i = 1, \ldots, d$ as the standard basis vectors in $\mathbb{R}^d$
- $\boldsymbol{e}_i$ as the constant signal $e_{i,t} = e_i \mathbb{1}_{\{t \geq 0\}}$

Given a functional sequence $\boldsymbol{H}$,

- Denote the output of constant signal $y_i(t) := H_t(\boldsymbol{e}_i)$
- smoothness is measured by the smoothness of $t \mapsto y_i(t)$
- memory is measured by the decay rate of the $t \mapsto y_i^{(k)}(t)$

Jackson-type Result for L-RNN

We assume that the memory decays exponentially

$$e^{\beta t} H_t^{(r)}(\boldsymbol{e}_i) = o(1), \qquad t \to \infty, \qquad i = 1, \ldots, d, \quad 1 \leq r \leq \alpha + 1$$

Then, we have a Jackson-type estimate

$$\boxed{\inf_{\widehat{H} \in \mathcal{H}^m_{\text{L-RNN}}} \|H - \widehat{H}\| \leq \frac{c_\alpha d \gamma}{\beta m^\alpha}, \qquad \gamma = \sup_{t \geq 0} \max_{i=1,\ldots,d} \max_{r=1,\ldots,\alpha+1} \frac{|e^{\beta t} H_t^{(r)}(\boldsymbol{e}_i)|}{\beta^r}}$$

Z. Li, J. Han, W. E, and Q. Li, "On the Curse of Memory in Recurrent Neural Networks: Approximation and Optimization Analysis,", presented at the International Conference on Learning Representations, 18, 2021

Rate estimate

$$\inf_{\widehat{H}\in\mathcal{H}^{m}_{\text{L-RNN}}} \|H - \widehat{H}\| \leq \frac{c_\alpha d\gamma}{\beta m^\alpha},$$

Curse of Memory in Approximation

Rate estimate

$$\inf_{\widehat{H}\in\mathcal{H}^{m}_{\text{L-RNN}}} \|H-\widehat{H}\| \leq \frac{c_\alpha d\gamma}{\beta m^\alpha},$$

Observations

- The smoothness dependence (α) is familiar

Curse of Memory in Approximation

Rate estimate

$$\inf_{\widehat{H}\in\mathcal{H}^m_{\text{L-RNN}}} \|H - \widehat{H}\| \leq \frac{c_\alpha d\gamma}{\beta m^\alpha},$$

Observations

- The smoothness dependence (α) is familiar
- The memory dependence (β, γ) is new: we need

$$y_i^{(r)}(t) \equiv H_t^{(r)}(\boldsymbol{e}_i) \sim e^{-\beta t}, \qquad \beta > 0, 1 \leq r \leq \alpha + 1$$

Curse of Memory in Approximation

Rate estimate

$$\inf_{\widehat{H} \in \mathcal{H}^m_{\text{L-RNN}}} \|H - \widehat{H}\| \leq \frac{c_\alpha d \gamma}{\beta m^\alpha},$$

Observations

- The smoothness dependence (α) is familiar
- The memory dependence (β, γ) is new: we need

$$y_i^{(r)}(t) \equiv H_t^{(r)}(\boldsymbol{e}_i) \sim e^{-\beta t}, \qquad \beta > 0, 1 \leq r \leq \alpha + 1$$

- There is no curse of dimensionality due to linearity

Curse of Memory in Approximation

Rate estimate

$$\inf_{\widehat{H}\in\mathcal{H}^m_{\text{L-RNN}}} \|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \leq \frac{c_\alpha d\gamma}{\beta m^\alpha},$$

Observations

- The smoothness dependence (α) is familiar
- The memory dependence (β, γ) is new: we need

$$y_i^{(r)}(t) \equiv H_t^{(r)}(\boldsymbol{e}_i) \sim e^{-\beta t}, \qquad \beta > 0, 1 \leq r \leq \alpha + 1$$

- There is no curse of dimensionality due to linearity
- However, hidden in these results is a curse of memory: if we replace

$$H_t^{(r)}(\boldsymbol{e}_i) \sim e^{-\beta t} \quad \longrightarrow \quad H_t^{(r)}(\boldsymbol{e}_i) \sim t^{-(r+\omega)} \quad (\omega > 0),$$

  then, the sufficient number of neurons to achieve approximation error of ϵ grows like $m \sim \epsilon^{-1/\omega}$

Bernstein-type Result for L-RNN

We can further derive a Bernstein-type result

Z. Li, J. Han, W. E, and Q. Li, "Approximation and Optimization Theory for Linear Continuous-Time Recurrent Neural Networks," Journal of Machine Learning Research, vol. 23, no. 42, 2022

Bernstein-type Result for L-RNN

We can further derive a Bernstein-type result

Assuming $\boldsymbol{H}$ can be efficiently approximated by $\{\mathcal{H}^m_{\text{L-RNN}}\}$, i.e. there exists $\widehat{\boldsymbol{H}}_m \in \mathcal{H}^m_{\text{L-RNN}}$ with $\|\boldsymbol{H} - \widehat{\boldsymbol{H}}_m\| \to 0$ and

$$\sup_{t\geq 0} |H_t^{(k)}(\boldsymbol{e}_i) - \widehat{H}_{m,t}^{(k)}(\boldsymbol{e}_i)| \to 0, \qquad k = 1, \ldots, \alpha + 1.$$

Z. Li, J. Han, W. E, and Q. Li, "Approximation and Optimization Theory for Linear Continuous-Time Recurrent Neural Networks," Journal of Machine Learning Research, vol. 23, no. 42, 2022

Bernstein-type Result for L-RNN

We can further derive a Bernstein-type result

Assuming $\boldsymbol{H}$ can be efficiently approximated by $\{\mathcal{H}^m_{\text{L-RNN}}\}$, i.e. there exists $\widehat{\boldsymbol{H}}_m \in \mathcal{H}^m_{\text{L-RNN}}$ with $\|\boldsymbol{H} - \widehat{\boldsymbol{H}}_m\| \to 0$ and

$$\sup_{t \geq 0} |H^{(k)}_t(\boldsymbol{e}_i) - \widehat{H}^{(k)}_{m,t}(\boldsymbol{e}_i)| \to 0, \qquad k = 1, \ldots, \alpha + 1.$$

Then, under technical conditions, there must exist a $\beta > 0$ with

$$\boxed{e^{\beta t} H^{(r)}_t(\boldsymbol{e}_i) = o(1), \qquad t \to \infty, \qquad i = 1, \ldots, d, \quad 1 \leq r \leq \alpha + 1.}$$

That is, the memory must decay exponentially!

Z. Li, J. Han, W. E, and Q. Li, "Approximation and Optimization Theory for Linear Continuous-Time Recurrent Neural Networks," Journal of Machine Learning Research, vol. 23, no. 42, 2022

Insight on RNN approximation

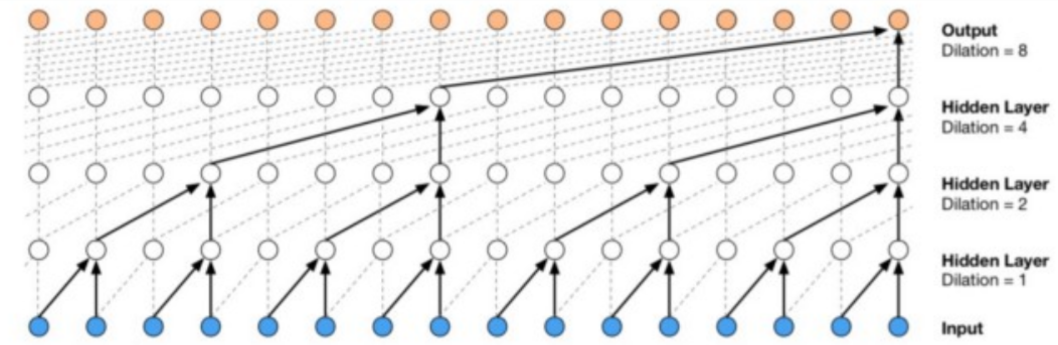
Efficient approximation $\iff$ Exponentially decaying memory

Temporal Convolutional Networks

Convolutional Architectures

A popular alternative to recurrent architectures is convolutional based architectures for sequence modelling

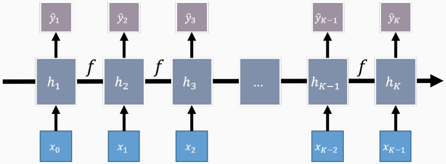
Example: WaveNet



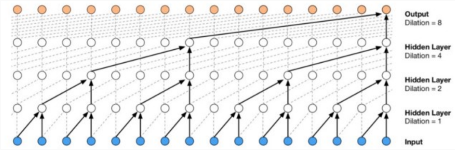
A. van den Oord et al., "Wavenet: A generative model for raw audio," Proc. 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9), 2016

Convolutional vs Recurrent Architectures

In practice, there are empirical works demonstrating the superiority of either, depending on application



v.s.



Is one really better than the other?

When should we use convolutional and recurrent architectures?

S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018

The Dilated Convolutional Architecture

For discrete sequences, define the dilated convolution operation

$$(\boldsymbol{u} *_l \boldsymbol{v})(t) = \sum_{s \geq 0} u(s)^\top v(t - ls), \qquad l \in \mathbb{Z}_+.$$

The Dilated Convolutional Architecture

For discrete sequences, define the dilated convolution operation

$$(u *_l v)(t) = \sum_{s \geq 0} u(s)^\top v(t - ls), \qquad l \in \mathbb{Z}_+.$$

The temporal CNN architecture is

$$\begin{aligned} h_{0,i} &= x_i \\ h_{k+1,i} &= \sigma\left(\sum_{j=1}^{M_k} w_{kji} *_{d_k} h_{k,j}\right) \\ \hat{y} &= h_{K,1} \end{aligned}$$

$h_{k,i}$	hidden state at layer k, channel i
w_{kji}	size $\mathbb{R}^l$ convolutional filters $l = 2$
K	# layers
M_k	# channels at layer k
d_k	dilation rate at layer k

Density Results for CNN

Density results for CNN are mostly studied for image applications

- Results without shift-equivariance[1,2] cannot be adapted to the temporal setting
- Results with shift equivariance[3,4] can be, but not straightforward for unbounded index sets

[1] K. Oono and T. Suzuki, "Approximation and non-parametric estimation of ResNet-type convolutional neural networks," in Proceedings of the 36th International Conference on Machine Learning, PMLR, May 24, 2019, pp. 4922–4931.

[2] D.-X. Zhou, "Universality of deep convolutional neural networks," Applied and computational harmonic analysis, vol. 48, no. 2, pp. 787–794, 2020.

[3] T. Lin, Z. Shen, and Q. Li, "On the Universal Approximation Property of Deep Fully Convolutional Neural Networks," 2022.

[4] D. Yarotsky, "Universal Approximations of Invariant Maps by Neural Networks," Constructive Approximation, vol. 55, no. 1, pp. 407–474, Feb. 1, 2022.

Linear Convolutional Hypothesis Space

To obtain quantitative results on unbound domains, we turn to linear activation case and choose $M_k = M$, $d_k = 2^k$

Linear Convolutional Hypothesis Space

To obtain quantitative results on unbound domains, we turn to linear activation case and choose $M_k = M$, $d_k = 2^k$

We get the linear temporal CNN hypothesis space

$$\mathcal{H}_{\text{L-CNN}} = \bigcup_{K,M} \mathcal{H}_{\text{L-CNN}}^{(K,M)} = \left\{ \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = \sum_{s=0}^{\infty} \widehat{\rho}(s)^\top x(t-s) \right\},$$

where $\widehat{\boldsymbol{\rho}}$ is determined by the convolutional filters $\{\boldsymbol{w}_{kji}\}$:

$$\widehat{\boldsymbol{\rho}}_i = \sum_{i_1,\dots,i_{K-1}} \boldsymbol{w}_{K-1,i_{K-1},1} \ast_{2^{K-1}} \boldsymbol{w}_{K-2,i_{K-2},i_{K-1}} \ast_{2^{K-2}} \cdots \ast_{2} \boldsymbol{w}_{0,i,i_1}.$$

Linear Convolutional Hypothesis Space

To obtain quantitative results on unbound domains, we turn to linear activation case and choose $M_k = M$, $d_k = 2^k$

We get the linear temporal CNN hypothesis space

$$\mathcal{H}_{\text{L-CNN}} = \bigcup_{K,M} \mathcal{H}_{\text{L-CNN}}^{(K,M)} = \left\{ \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = \sum_{s=0}^{\infty} \widehat{\rho}(s)^\top x(t-s) \right\},$$

where $\widehat{\boldsymbol{\rho}}$ is determined by the convolutional filters $\{\boldsymbol{w}_{kji}\}$:

$$\widehat{\boldsymbol{\rho}}_i = \sum_{i_1,\ldots,i_{K-1}} \boldsymbol{w}_{K-1,i_{K-1},1} \ast_{2^{K-1}} \boldsymbol{w}_{K-2,i_{K-2},i_{K-1}} \ast_{2^{K-2}} \cdots \ast_2 \boldsymbol{w}_{0,i,i_1}.$$

Compare this with the L-RNN:

- They are both of the form $H_t(\boldsymbol{x}) = \sum \widehat{\boldsymbol{\rho}}(s)^\top x(t-s)$
- RNN: $\widehat{\boldsymbol{\rho}}$ is an exponential sum (infinite support)
- CNN: $\widehat{\boldsymbol{\rho}}$ is a product-sum of length $l = 2$ filters (finite support)

Density-type Results for Temporal L-CNN

Consider the same assumptions of H being continuous, linear, causal and shift-equivariant

H. Jiang, Z. Li, and Q. Li, "Approximation Theory of Convolutional Architectures for Time Series Modelling," in Proceedings of the 38th International Conference on Machine Learning, PMLR, 1, 2021

Density-type Results for Temporal L-CNN

Consider the same assumptions of $\boldsymbol{H}$ being continuous, linear, causal and shift-equivariant

Temporal CNN approximates

$$H_t(\boldsymbol{x}) = \sum_{s \geq 0} \rho(s)^\top x(t-s) \qquad \text{by} \qquad \widehat{H}_t(\boldsymbol{x}) = \sum_{s \geq 0} \widehat{\rho}(s)^\top x(t-s)$$

with $\widehat{\boldsymbol{\rho}}$ (the Riesz representation of $\widehat{\boldsymbol{H}}$) being a convolution product-sum

H. Jiang, Z. Li, and Q. Li, "Approximation Theory of Convolutional Architectures for Time Series Modelling," in Proceedings of the 38th International Conference on Machine Learning, PMLR, 1, 2021

Density-type Results for Temporal L-CNN

Consider the same assumptions of $\boldsymbol{H}$ being continuous, linear, causal and shift-equivariant

Temporal CNN approximates

$$H_t(\boldsymbol{x}) = \sum_{s \geq 0} \rho(s)^\top x(t-s) \qquad \text{by} \qquad \widehat{H}_t(\boldsymbol{x}) = \sum_{s \geq 0} \widehat{\rho}(s)^\top x(t-s)$$

with $\widehat{\boldsymbol{\rho}}$ (the Riesz representation of $\widehat{\boldsymbol{H}}$) being a convolution product-sum

An immediate consequence is that the temporal CNN hypothesis is dense

H. Jiang, Z. Li, and Q. Li, "Approximation Theory of Convolutional Architectures for Time Series Modelling," in Proceedings of the 38th International Conference on Machine Learning, PMLR, 1, 2021

Principles of CNN Approximation

How does CNN approximation work?

Principles of CNN Approximation

How does CNN approximation work?

First, since a K-layer L-CNN has Riesz representation $\widehat{\boldsymbol{\rho}}$ of support 2^K, the estimate should be of the form

$$\sup_{\|\boldsymbol{x}\|\leq 1} |H_t(\boldsymbol{x}) - \widehat{H}_t(\boldsymbol{x})| \leq \underbrace{\sum_{s=0}^{2^K-1} |\rho(s) - \widehat{\rho}(s)|}_{\text{approx by convs}} + \underbrace{\sum_{s=2^K}^{\infty} |\rho(s)|}_{\text{tail (memory)} \to 0}$$

How does CNN approximation work?

First, since a K-layer L-CNN has Riesz representation $\widehat{\boldsymbol{\rho}}$ of support 2^K, the estimate should be of the form

$$\sup_{\|\boldsymbol{x}\|\leq 1} |H_t(\boldsymbol{x}) - \widehat{H}_t(\boldsymbol{x})| \leq \underbrace{\sum_{s=0}^{2^K-1} |\rho(s) - \widehat{\rho}(s)|}_{\text{approx by convs}} + \underbrace{\sum_{s=2^K}^{\infty} |\rho(s)|}_{\text{tail (memory)} \to 0}$$

Note:

- Second term goes to 0 exponentially in K for any $\boldsymbol{\rho} \in \ell^2$ (does not require exponential decay!)

How does CNN approximation work?

First, since a K-layer L-CNN has Riesz representation $\widehat{\boldsymbol{\rho}}$ of support 2^K, the estimate should be of the form

$$\sup_{\|\boldsymbol{x}\|\leq 1} |H_t(\boldsymbol{x}) - \widehat{H}_t(\boldsymbol{x})| \leq \underbrace{\sum_{s=0}^{2^K-1} |\rho(s) - \widehat{\rho}(s)|}_{\text{approx by convs}} + \underbrace{\sum_{s=2^K}^{\infty} |\rho(s)|}_{\text{tail (memory)} \to 0}$$

Note:

- Second term goes to 0 exponentially in K for any $\boldsymbol{\rho} \in \ell^2$ (does not require exponential decay!)
- So, what does CNN require for efficient approximation?

A Minimal Example

Let us consider approximating a target functional sequence

$$H_t(\boldsymbol{x}) = r_0 x(t) + r_1 x(t-1) + r_2 x(t-2) + r_3 x(t-3), \qquad r_s \in \mathbb{R}$$

A Minimal Example

Let us consider approximating a target functional sequence

$$H_t(\boldsymbol{x}) = r_0 x(t) + r_1 x(t-1) + r_2 x(t-2) + r_3 x(t-3), \qquad r_s \in \mathbb{R}$$

This $\boldsymbol{H}$ has Riesz representation of support 4

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3)$$

A Minimal Example

Let us consider approximating a target functional sequence

$$H_t(\boldsymbol{x}) = r_0 x(t) + r_1 x(t-1) + r_2 x(t-2) + r_3 x(t-3), \qquad r_s \in \mathbb{R}$$

This $\boldsymbol{H}$ has Riesz representation of support 4

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3)$$

Let us try to approximate it with a temporal CNN of depth $K = 2$ and channel width $M = 1$, which has Riesz representation

$$\widehat{\boldsymbol{\rho}} = (w_{0,0}, w_{0,1}) \ast_2 (w_{1,0}, w_{1,1})$$

A Minimal Example

Therefore, we are seeking the approximation of

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3) \qquad \text{by} \qquad \widehat{\boldsymbol{\rho}} = (w_{0,0}, w_{0,1}) \ast_2 (w_{1,0}, w_{1,1}),$$

A Minimal Example

Therefore, we are seeking the approximation of

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3) \qquad \text{by} \qquad \widehat{\boldsymbol{\rho}} = (w_{0,0}, w_{0,1}) \ast_2 (w_{1,0}, w_{1,1}),$$

which we can rewrite in matrix form as the approximation of

$$\mathsf{T}(\boldsymbol{\rho}) = \begin{pmatrix} r_0 & r_1 \\ r_2 & r_3 \end{pmatrix} \qquad \text{by} \qquad \mathsf{T}(\widehat{\boldsymbol{\rho}}) = \begin{pmatrix} w_{0,0} \\ w_{0,1} \end{pmatrix} \begin{pmatrix} w_{1,0} & w_{1,1} \end{pmatrix}$$

A Minimal Example

Therefore, we are seeking the approximation of

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3) \qquad \text{by} \qquad \widehat{\boldsymbol{\rho}} = (w_{0,0}, w_{0,1}) \ast_2 (w_{1,0}, w_{1,1}),$$

which we can rewrite in matrix form as the approximation of

$$\mathsf{T}(\boldsymbol{\rho}) = \begin{pmatrix} r_0 & r_1 \\ r_2 & r_3 \end{pmatrix} \qquad \text{by} \qquad \mathsf{T}(\widehat{\boldsymbol{\rho}}) = \begin{pmatrix} w_{0,0} \\ w_{0,1} \end{pmatrix} \begin{pmatrix} w_{1,0} & w_{1,1} \end{pmatrix}$$

Then, approximation error is clear:

- If $\mathsf{T}(\boldsymbol{\rho})$ is rank 1, then approximation error is 0
- If $\mathsf{T}(\boldsymbol{\rho})$ is rank 2, then optimal approximation error is its second singular value (Eckart-Young-Mirsky theorem)

The Notion of Tensorization Rank

This argument can be generalized to arbitrary M, K

T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," SIAM Rev., vol. 51, no. 3, 6, 2009

The Notion of Tensorization Rank

This argument can be generalized to arbitrary M, K

For $K \geq 3$ the tensorisation $\mathsf{T}(\cdot)$ of a length-2^K sequence produces a rank-K tensor of linear dimension 2

$$\mathsf{T}(\boldsymbol{\rho}_{[0,2^K]})_{i_1,\ldots,i_K} = \rho_{[0,2^K]}\left(\sum_{j=1}^{K} i_j 2^{j-1}\right), \qquad i_j \in \{0,1\}.$$

T. G. Kolda and B. W. Bader, “Tensor Decompositions and Applications,” SIAM Rev., vol. 51, no. 3, 6, 2009

The Notion of Tensorization Rank

This argument can be generalized to arbitrary M, K

For $K \geq 3$ the tensorisation $\mathsf{T}(\cdot)$ of a length-2^K sequence produces a rank-K tensor of linear dimension 2

$$\mathsf{T}(\boldsymbol{\rho}_{[0,2^K]})_{i_1,\dots,i_K} = \rho_{[0,2^K]}\left(\sum_{j=1}^{K} i_j 2^{j-1}\right), \qquad i_j \in \{0,1\}.$$

This has higher order singular values (HOSV)

$$\sigma_1^{(K)} \geq \sigma_2^{(K)} \geq \cdots \geq \sigma_{2K}^{(K)} \geq 0,$$

whose decay rate (effecrtive tensorisation rank) controls the approximation error by width-M filters

T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," SIAM Rev., vol. 51, no. 3, 6, 2009

Jackson-type Results for L-CNN

This motivates us to define the complexity measure for $\boldsymbol{H}$

$$C^{(G)}(H) = \inf \left\{ c : \Big(\Sigma_{i=s+K}^{2K} \underbrace{(\sigma_i^{(K)})^2}_{\text{tail of singular values of tensorization}}\Big)^{\frac{1}{2}} \leq c \overbrace{G(s)}^{\text{decay rate}}, \ s \geq 0, K \in \mathbb{N}_+ \right\}$$

H. Jiang, Z. Li, and Q. Li, "Approximation Theory of Convolutional Architectures for Time Series Modelling," in Proceedings of the 38th International Conference on Machine Learning, PMLR, 1, 2021

This motivates us to define the complexity measure for $\boldsymbol{H}$

$$C^{(G)}(\boldsymbol{H}) = \inf\left\{ c : (\Sigma_{i=s+K}^{2K} \underbrace{(\sigma_i^{(K)})^2}_{\text{tail of singular values of tensorization}})^{\frac{1}{2}} \le c \overbrace{G(s)}^{\text{decay rate}}, \; s \ge 0, K \in \mathbb{N}_+ \right\}$$

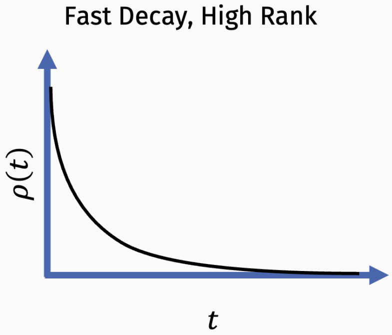
Then, we can show the following Jackson-type estimate

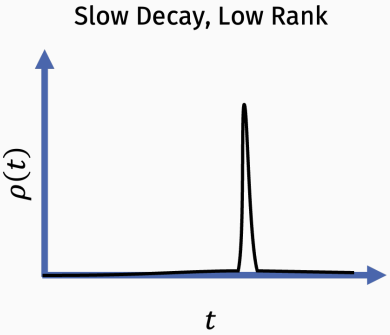
$$\boxed{\inf_{\widehat{\boldsymbol{H}} \in \mathcal{H}_{\text{L-CNN}}^{(K,M)}} \|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \le G(KM^{\frac{1}{K}} - K)C^{(G)}(\boldsymbol{H})d + \|\boldsymbol{\rho}_{[2^K,\infty)}\|_2}$$

H. Jiang, Z. Li, and Q. Li, "Approximation Theory of Convolutional Architectures for Time Series Modelling," in Proceedings of the 38th International Conference on Machine Learning, PMLR, 1, 2021

So, is CNN or RNN better? In general, neither!

Insight:	RNN works well if ρ decays exponentially
	CNN works well if ρ has low rank under tensorization



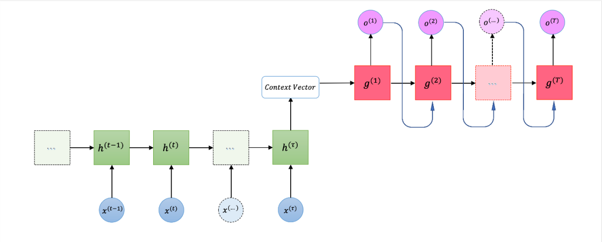


Encoder-Decoder Networks

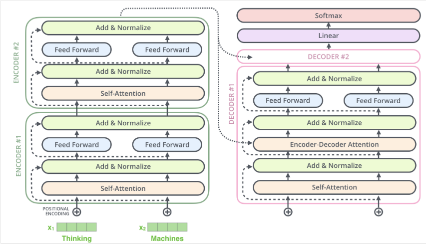
Encoder-Decoder Architectures for Modelling Sequences

Alternative to RNNs and CNNs are encoder-decoder class of architectures

Recurrent Encoder-Decoder

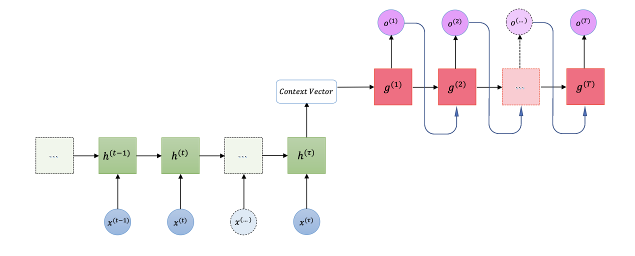


Transformer



How do they compare with RNN and CNN?

The simplest form of encoder-decoder architecture is the recurrent variant



$$
\begin{aligned}
\dot{h}(s) &= \sigma_E(Wh(s) + Ux(s)), \quad & v = Qh_0, \quad & s \leq 0 \\
\dot{g}(t) &= \sigma_D(Vg(t)), & g_0 = Pv, & \\
\widehat{y}(t) &= c^\top g(t), & t \geq 0, &
\end{aligned}
$$

K. Cho et al., "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2, 2014

I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," Advances in neural information processing systems, vol. 27, 2014

The Linear Recurrent Encoder-Decoder Hypothesis Space

In the linear setting, we have the following hypothesis spaces ($d = 1$ for simplicity)

$$\mathcal{H}_{\text{L-REncDec}} = \bigcup_{m,N} \mathcal{H}^{(m,N)}_{\text{L-REncDec}} = \bigcup_{m,N} \left\{ \widehat{\mathbf{H}} : \widehat{H}_t(\mathbf{x}) = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t) \widehat{\phi}_n(s) x(-s) ds \right\},$$

Where

- m is the width of the encoder and decoder RNNs
- N is the size of the context vector

In the linear setting, we have the following hypothesis spaces ($d = 1$ for simplicity)

$$\mathcal{H}_{\text{L-REncDec}} = \bigcup_{m,N} \mathcal{H}^{(m,N)}_{\text{L-REncDec}} = \bigcup_{m,N} \left\{ \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t) \widehat{\phi}_n(s) x(-s) ds \right\},$$

Where

- m is the width of the encoder and decoder RNNs
- N is the size of the context vector

The sequences $\widehat{\boldsymbol{\psi}}_{\boldsymbol{n}}$ and $\widehat{\boldsymbol{\phi}}_{\boldsymbol{n}}$ are in exponential sum forms

$$\widehat{\psi}_n(t) = \left(\sum_{i,j=1}^{m} c_i P_{jn} \left[e^{Vt} \right]_{ij} \right), \qquad \widehat{\phi}_n(t) = \left(\sum_{i,j=1}^{m} u_i Q_{nj} \left[e^{Wt} \right]_{ji} \right).$$

Principles of Encoder-Decoder Approximation

Since the input and ouput sequences no-longer have time correspondence, shift equivariance is lost

Principles of Encoder-Decoder Approximation

Since the input and ouput sequences no-longer have time correspondence, shift equivariance is lost

Without shift-equivariance, the Riesz representation of $\boldsymbol{H}$ is

$$H_t(\boldsymbol{x}) = \int_0^\infty \rho(t,s)x(-s)ds. \qquad \left[\text{Compare: } \widehat{H}_t = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)x(-s)ds\right]$$

Principles of Encoder-Decoder Approximation

Since the input and ouput sequences no-longer have time correspondence, shift equivariance is lost

Without shift-equivariance, the Riesz representation of $\boldsymbol{H}$ is

$$H_t(\boldsymbol{x}) = \int_0^\infty \rho(t,s)x(-s)ds. \qquad \left[\text{Compare: } \widehat{H}_t = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)x(-s)ds\right]$$

Thus, recurrent encoder-decoders approximates

$$\rho(t,s) \qquad \text{by} \qquad \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)$$

Principles of Encoder-Decoder Approximation

Since the input and ouput sequences no-longer have time correspondence, shift equivariance is lost

Without shift-equivariance, the Riesz representation of $\boldsymbol{H}$ is

$$H_t(\boldsymbol{x}) = \int_0^\infty \rho(t,s)x(-s)ds. \qquad \left[\text{Compare: } \widehat{H}_t = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)x(-s)ds\right]$$

Thus, recurrent encoder-decoders approximates

$$\rho(t,s) \qquad \text{by} \qquad \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)$$

That is, a rank-N approximation of a bi-variate function!

Jackson-type Results for L-REncDec

Let us write down the formal SVD of $\rho(t,s)$ as

$$\rho(t,s) = \sum_{n=1}^{\infty} \sigma_n \psi_n(t) \phi_n(s)$$

Z. Li, H. Jiang, and Q. Li, "On the approximation properties of recurrent encoder-decoder architectures,", presented at the International Conference on Learning Representations, 14, 2022

Jackson-type Results for L-REncDec

Let us write down the formal SVD of $\rho(t, s)$ as

$$\rho(t, s) = \sum_{n=1}^{\infty} \sigma_n \psi_n(t) \phi_n(s)$$

Optimal rank-N approximation error depends on decay of singular values

$$C(\boldsymbol{H}, N) \propto \left(\sum_{n=N+1}^{\infty} \sigma_n^2 \right)^{\frac{1}{2}}.$$

Z. Li, H. Jiang, and Q. Li, "On the approximation properties of recurrent encoder-decoder architectures,", presented at the International Conference on Learning Representations, 14, 2022

Jackson-type Results for L-REncDec

Let us write down the formal SVD of $\rho(t,s)$ as

$$\rho(t,s) = \sum_{n=1}^{\infty} \sigma_n \psi_n(t) \phi_n(s)$$

Optimal rank-N approximation error depends on decay of singular values

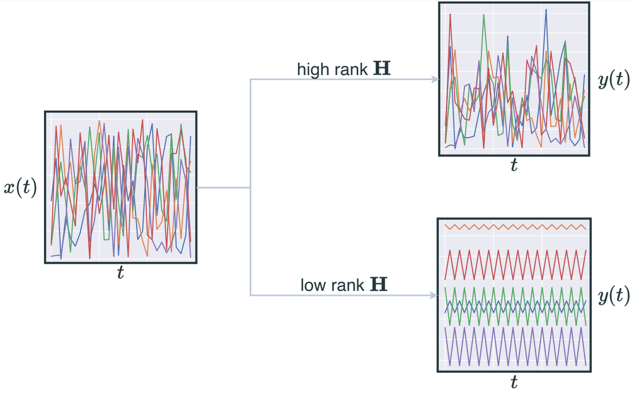
$$C(H,N) \propto \left(\sum_{n=N+1}^{\infty} \sigma_n^2 \right)^{\frac{1}{2}}.$$

We then arrive at a Jackson-type estimate

$$\boxed{\inf_{\widehat{H} \in \mathcal{H}^{m,N}_{\text{L-REncDec}}} \|H - \widehat{H}\| \leq \frac{C_1(\alpha)\gamma}{\beta^2 m^\alpha} + C(H,N),}$$

Z. Li, H. Jiang, and Q. Li, "On the approximation properties of recurrent encoder-decoder architectures,", presented at the International Conference on Learning Representations, 14, 2022

The Notion of Effective Rank under the Temporal Product Structure



Insight: Encoder-decoders are most effective in capturing temporal product structures with low effective rank

Summary

We introduced a basic mathematical setting that allows precise analysis of a variety of architectures including

- RNN, CNN, Recurrent Encoder-Decoder

From the approximation viewpoint

- Can all achieve density in appropriate functional spaces
- Efficient approximation depends on different notions of complexity
  - RNN: Exponential memory decay
  - CNN: Low rank under tensorization
  - Recurrent Encoder-Decoder: Low rank under temporal products

> Need structural compatibility between the model and the target

H. Jiang, Q. Li, Z. Li, and S. Wang, "A Brief Survey on the Approximation Theory for Sequence Modelling," JML, vol. 2, no. 1, 2023

Thank you!