Journal of Machine Learning **ISSN**: 2790-2048(e), 2790-203X(p)

A Brief Survey on the Approximation Theory for Sequence Modelling

Haotian Jiang [1], Qianxiao Li * [1,2], Zhong Li [3], and Shida Wang [1]

[1]Department of Mathematics, National University of Singapore, Singapore.
[2]Institute for Functional Intelligent Materials, National University of Singapore, Singapore.
[3]Microsoft Research Asia

Abstract. We survey current developments in the approximation theory of sequence modelling in machine learning. Particular emphasis is placed on classifying existing results for various model architectures through the lens of classical approximation paradigms, and the insights one can gain from these results. We also outline some future research directions towards building a theory of sequence modelling.

Keywords:
Approximation theory,
Sequence modelling,
Machine learning,
Deep learning,
Dynamics.

Article Info.:
Volume: 2
Number: 1
Pages: 1- 30
Date: March/2023
doi.org/10.4208/jml.221221

Article History:
Received: 21/12/2022
Accepted: 04/03/2023

Handler:
Weinan E

1 Introduction

The modelling of relationships between sequences is an important task that enables a wide array of applications, including classical time-series prediction problems in finance [78], and modern machine learning problems in natural language processing [4]. Another class of engineering applications involving sequential relationships are control systems, which study the dependence of a dynamical trajectory on an input control sequence [12]. In general, sequence-to-sequence relationships can be very complex. For example, when the index set for the sequences is infinite, one can understand these relationships as mappings between infinite-dimensional spaces. Thus, traditional modelling techniques are limited in their efficacy, especially when there is little prior knowledge on the system of interest. To address these difficulties, an increasingly popular method to model sequence relationships is to leverage machine learning.

To date, a large variety of machine learning paradigms have been proposed to model sequential relationships. One of the earliest attempts is the class of neural networks called recurrent neural networks (RNN) [70], and their variants [17,39]. Besides the RNN family of models, many other alternatives have also been explored. These include convolutional based models [84], encoder-decoder based models [17] attention based models [4], and their combinations. For example, the powerful transformer architecture [85] combines encoder-decoder and attention architectures.

Despite the rapid developments in the practical domain, the theoretical foundation of data-driven sequence modelling is still in its nascent stages. For example, the most

*Corresponding author. qianxiao@nus.edu.sg.

basic question of how the aforementioned architectures are different, and how practitioners should select the model architecture based on their applications, is largely unknown and relies on trial and error. Thus, an important direction of theoretical research is to understand the essential properties, and most importantly, distinctions between different sequence modelling paradigms.

The present survey aims to provide an overview of the theoretical research on sequence modelling in the specific direction of approximation theory [22, 59]. In a nutshell, approximation theory is the study of how a complex relationship (say a function) can be broken down as a combination of simpler, more readily computable building blocks. A theoretical foundation of sequence modelling requires the understanding of how and when a sequential relationship can be approximated by simpler components realized as various neural network architectures. The theory of sequence modelling is an active area of research that spans decades of work both in machine learning and the study of nonlinear dynamics. Thus, the purpose of this survey is not to give an exhaustive summary of all relevant results in the literature, but rather to highlight some interesting insights for approximation theory gained from existing works. Moreover, we discuss some classes of open questions that are of significance in order to progress the understanding of the approximation theory for sequences.

The survey is organized as follows. In Section 2, we introduce the mathematical problem of approximation, including the key questions one is interested in answering. In particular, we highlight the new aspects of sequence approximation as compared to classical paradigms of function approximation. In Section 3, we discuss approximation results on recurrent neural networks, where much more is known compared with other architectures. In Section 4, we consider the approximation theory of other model architectures, including those of convolutional, encoder-decoder and attention types. In Section 5, we summarize the known results and motivate some future directions of interest.

Notation. Let us introduce some notational conventions. Throughout this paper, we use lower-case letters to denote scalars and vectors. Boldface letters are reserved for sequences, e.g. $\boldsymbol{x} = \{x(t) : t \in \mathcal{T}\}$. As in the previous formula, script letters such as $\mathcal{T}$ are used to represent sets of scalar or vectors, both finite and infinite-dimensional. Capital letters are used to denote mappings between vector spaces. Correspondingly, a bold-faced capital letter is a sequence of such mappings. Sometimes, we wish to refer to a portion of a sequence. Let $\mathcal{S} \subset \mathcal{T}$, then $\boldsymbol{x}_{\mathcal{S}} := \{x(t) : t \in \mathcal{S}\}$. We use $|\cdot|$ to denote the Euclidean norm, and reserve $\|\cdot\|$, possibly with subscripts, for norms over function (e.g. sequence) spaces. We use $\dot{x}(t)$ to denote the derivative of $t \mapsto x(t)$. Higher derivatives of order $r \geq 0$ are written as $x^{(r)}(t)$. Throughout this survey, we reserve the letters m, n, d, i, j to represent integers. Sequence indices, equivalently referred to as time indices, are written as t or s.

2 Sequence modelling as an approximation problem

We begin by formalizing the broad mathematical problem of approximation and some of the key questions one may be interested in. We then discuss how one may formulate sequence modelling in the setting of approximation theory.

2.1 The problem of approximation

Let us introduce the basic problem of approximation for functions on vector spaces. Let $\mathcal{X}$ and $\mathcal{Y}$ be normed vector spaces. We consider a family of target functions, or simply targets, which is a subset $\mathcal{C}$ of all mappings $\mathcal{X} \to \mathcal{Y}$, i.e. $\mathcal{C} \subset \mathcal{Y}^{\mathcal{X}}$. In the learning theory literature, one sometimes calls $\mathcal{C}$ a concept space. These are the relationships we wish to learn, or approximate, by some simpler candidate functions. Let us denote this set of candidates by $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$. In learning theory, this is often called a hypothesis space. The problem of approximation concerns how well can functions in $\mathcal{H}$ resolve functions in $\mathcal{C}$.

In broad terms, we may classify results on approximation theory into three types: universal approximation results (density-type), approximation rate estimates (Jackson-type), and inverse approximation results (Bernstein-type). Let us discuss each in turn.

Universal approximation results (density-type). Universal approximation theorems are the most basic approximation results. We say that $\mathcal{H}$ is an universal approximator for $\mathcal{C}$ if for every $H \in \mathcal{C}$ and $\epsilon > 0$, there exists $\widehat{H} \in \mathcal{H}$ such that $\|H - \widehat{H}\| \leq \epsilon$. In other words, $\mathcal{H}$ is dense in $\mathcal{C}$ in the topology generated by $\|\cdot\|$. The choice of the norm depends on applications. We illustrate this with the following example.

We consider approximating scalar functions by trigonometric polynomials. Here, we set $\mathcal{X} = [0, 2\pi]$ and $\mathcal{Y} = \mathbb{R}$. The target space is $\mathcal{C} = C^{\alpha}_{\text{per}}([0, 2\pi])$, the set of α-times continuously differentiable, periodic functions on $[0, 2\pi]$. The hypothesis space is

$$\mathcal{H} = \cup_{m \in \mathbb{N}_+} \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R}, m \in \mathbb{N}_+ \right\}. \tag{2.1}$$

As a direct consequence of the Stone-Weierstrass theorem, $\mathcal{H} \subset \mathcal{C}$ is dense in $\mathcal{C}$ with respect to the norm $\|H\| = \sup_{x \in [0,2\pi]} |H(x)|$ [2, p. 32].

Approximation rate estimates (Jackson-type). Universal approximation (density) ensures that our hypothesis space $\mathcal{H}$ is in a sense "big enough", so that we can use it to approximate a reasonably large variety of target functions. However, such results do not tell us precisely what types of functions in $\mathcal{C}$ are "easy" (or "hard") to approximate using $\mathcal{H}$. In other words, two hypothesis spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ may both be dense in $\mathcal{C}$ but can be naturally adapted to approximate functions of different types.

To resolve this, we may ask a finer question on the rate of approximation. Fix a hypothesis space $\mathcal{H}$. Let $\{\mathcal{H}^m : m \in \mathbb{N}_+\}$ be a collection of subsets of $\mathcal{H}$ such that $\mathcal{H}^m \subset \mathcal{H}^{m+1}$ and $\cup_{m \in \mathbb{N}_+} \mathcal{H}^m = \mathcal{H}$. Here, m is a measure of complexity of the approximation candidates, and $\mathcal{H}^m$ is the subset of hypotheses with complexity at most m. This is also called the approximation budget. Then, the approximation rate estimate is an inequality of the form

$$\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq C_{\mathcal{H}}(H, m). \tag{2.2}$$

Eq. (2.2) tells us the best possible approximation error one can hope to obtain under approximation budget m. Note that $\mathcal{H}$ is dense if and only if $\lim_{m \to \infty} C_{\mathcal{H}}(H, m) = 0$ for every

$H \in \mathcal{C}$. The speed at which $C_{\mathcal{H}}(H, m)$ decays as m increases is the approximation rate, and its dependence on H measures the complexity of a particular target H under the current approximation scheme $\mathcal{H}$.

Returning to the example in Eq. (2.1), the hypothesis space with budget m is

$$\mathcal{H}^m = \left\{ \widehat{H}(x) = \sum_{i=0}^{m-1} a_i \cos(ix) + b_i \sin(ix) : a_i, b_i \in \mathbb{R} \right\}. \tag{2.3}$$

The classical Jackson's theorem [2, p. 187] gives a rate estimate of the form

$$\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq \frac{c_\alpha \max_{0 \leq r \leq \alpha} \|H^{(r)}\|}{m^\alpha}, \tag{2.4}$$

where c_α is a constant depending only on α. Observe that the rate of decay of the approximation error is $m^{-\alpha}$ and the complexity of a target function under the trigonometric polynomial approximation scheme is its norm associated with the Sobolev space $\mathcal{W}^{\alpha,\infty} := \{H : \max_{0 \leq r \leq \alpha} \|H^{(r)}\| < \infty\}$. The key insight here is that a function H is easy to approximate using trigonometric polynomials if it has small gradient (Sobolev) norm. We will hereafter refer to estimates in the form of Eq. (2.2) as Jackson-type results.

Inverse approximation results (Bernstein-type). Jackson-type results tell us that if a target function H possesses some property related to $\{\mathcal{H}^m\}$, (e.g. smoothness, small gradient norm), then it is in fact easy to approximate with $\{\mathcal{H}^m\}$. Inverse approximation results are converse statements. It identifies properties that H ought to possess if one starts with the assumption that it can be well-approximated (in a sense to be made precise in each case) by $\{\mathcal{H}^m\}$.

In the case of trigonometric polynomial approximation, the following inverse approximation result is due to Bernstein [2, p. 206]. Fix some periodic $H : [0, 2\pi] \to \mathbb{R}$, and suppose that there exists a constant $c > 0$, $\delta > 0$ and $\alpha \in \mathbb{N}_+$ so that for every $m \in \mathbb{N}_+$, one has

$$\inf_{\widehat{H} \in \mathcal{H}^m} \|H - \widehat{H}\| \leq \frac{c}{m^{\alpha+\delta}}. \tag{2.5}$$

Then, H is α-times continuously differentiable and its α-th derivative is δ-Hölder continuous. Intuitively, this result says that if a function H can be approximated with a rate in Eq. (2.4), then it must be in $C^\alpha_{\text{per}}([0, 2\pi])$. Combined with Jackson's result, one gains a complete characterization of the type of functions - namely smooth functions, and their associated Sobolev spaces - that can be efficiently approximated with trigonometric polynomials. We will hereafter refer to these inverse approximation theorems as Bernstein-type results.

2.2 Sequence modelling as an approximation problem

Now, we introduce the problem of sequence approximation, which can be regarded as a particular class of approximation problems as introduced in Section 2.1. The key difference with classical approximation theories is that the input spaces $\mathcal{X}$ and the output spaces $\mathcal{Y}$ are now spaces of sequences, and may be infinite-dimensional.

We consider an input sequence indexed by a completely ordered index set $\mathcal{T}$

$$\boldsymbol{x} = \{x(t) : t \in \mathcal{T}\}. \tag{2.6}$$

There are two main choices of the index set $\mathcal{T}$. For discrete sequences (e.g. sequences of word embeddings), $\mathcal{T}$ is (a subset of) $\mathbb{Z}$. For continuous sequences (e.g. measurements of a continuous-time control system), $\mathcal{T}$ is (a subset of) $\mathbb{R}$. The input space is a collection $\mathcal{X}$ of such sequences. Correspondingly, the output space $\mathcal{Y}$ is another collection of sequences.

Each input sequence $\boldsymbol{x} \in \mathcal{X}$ corresponds to an output sequence $\boldsymbol{y}$ with

$$y(t) = H_t(\boldsymbol{x}), \quad t \in \mathcal{T}. \tag{2.7}$$

That is, the sequence $\boldsymbol{H} = \{H(t) \equiv H_t : t \in \mathcal{T}\}$ is our target. In this case, the target is in general an infinite-dimensional operator mapping $\mathcal{X} \to \mathcal{Y}$, and for each t, H_t is a functional on $\mathcal{X}$. We will hereafter refer to operators of this type as functional sequences.

Now, we seek to approximate $\boldsymbol{H}$ by candidates from a hypothesis space $\mathcal{H}$. The latter may be recurrent neural networks, convolutional networks or other types of models. In each case, one first identifies appropriate target spaces $\mathcal{C}$ for which $\mathcal{H}$ is dense. Then, one seeks Jackson-type and Bernstein-type results that characterize the types of sequence relationships that can be efficiently approximated by each hypothesis space.

From the viewpoint of classical approximation theory, one novel aspect of sequence approximation is that the input and output spaces are infinite-dimensional, provided that the index set $\mathcal{T}$ is infinite. In fact, many interesting aspects of sequence modelling, such as those associated with memory, precisely result from an unbounded index set, e.g. $\mathcal{T} = \mathbb{R}$ or $\mathbb{Z}$. We note that while sequence modelling is in effect an infinite-dimensional approximation problem, it should be contrasted with generic operator learning problems [9,15,20,47,61,65,75]. Here, a sequence is not a generic function but one with domain being a completely ordered index set. Therefore, the sequences and their corresponding vector spaces contain temporal structure that should be highlighted in the approximation results. It is for this reason that we do not call $\boldsymbol{H}$ operators, but functional sequences, to highlight the presence of its sequential structure. We close this part with a final remark. There are many applications where the output is not a sequence but rather just a finite dimensional vector. Examples include sequence regression [87] and sentiment analysis [77]. The present formulation includes these cases by writing $y \equiv y(\infty) = H_\infty(\boldsymbol{x})$ as the input-output functional relationship.

In next sections, we give a brief but structured overview of the approximation results for sequence modelling, paying particular attention to the theoretical insights and their consequences on practical architecture design.

3 Recurrent neural networks

Recurrent neural networks (RNN) are one of the earliest model architectures proposed for modelling sequential relationships [70]. The key idea is the introduction of a hidden dynamical system that captures the memory patterns in the sequences. We begin by introducing the RNN architecture and its corresponding hypothesis space.

3.1 Recurrent neural network hypothesis space

We first consider modelling a sequential relationship on the index set $\mathcal{T} = \mathbb{Z}$. Suppose at each time, the input sequence $x(t) \in \mathbb{R}^d$ is a vector. Without much loss in generality, we can consider the output sequence as a scalar sequence, i.e. $y(t) \in \mathbb{R}$. For vector-valued output sequences, one may consider each output dimension separately to deduce corresponding results. The (one-layer) recurrent neural network parametrizes the relationship between an input sequence $\boldsymbol{x}$ and an output sequence $\boldsymbol{y}$ as the following discrete dynamical system:[1]

$$\begin{aligned} h(t+1) &= \sigma\big(Wh(t) + Ux(t) + b\big), \\ y(t) &= c^\top h(t), \end{aligned} \qquad t \in \mathbb{Z}. \tag{3.1}$$

Here, $\boldsymbol{h}$ is a hidden state sequence, with each $h(t) \in \mathbb{R}^m$. Thus, the trainable parameters are $W \in \mathbb{R}^{m\times m}$, $U \in \mathbb{R}^{m\times d}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^m$. Conventionally, we impose a zero initial condition on $\boldsymbol{h}$, i.e. if the input sequence first becomes non-zero at a particular t_0 then $h(t_0) = 0$. For theoretical treatments, we can also take $t_0 = -\infty$ to handle inputs of unbounded support. The function σ is an activation function, which is a scalar function acting element-wise. In typical RNNs, σ is taken as the hyperbolic tangent function (tanh), but many other choices are possible.

Observe that Eq. (3.1) defines a functional sequence $\widehat{\boldsymbol{H}}$, with $y(t) = \widehat{H}_t(\boldsymbol{x}) = c^\top h(t)$, and $h(t)$ satisfies the dynamics in Eq. (3.1). Formally, we can write the RNN hypothesis space as

$$\begin{aligned} \mathcal{H}_{\mathrm{RNN}} &= \bigcup_{m\in\mathbb{N}_+} \mathcal{H}^m_{\mathrm{RNN}}, \\ \mathcal{H}^m_{\mathrm{RNN}} &= \left\{ \begin{array}{l} \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = c^\top h(t), \boldsymbol{h} \text{ follows Eq. (3.1) with} \\ W \in \mathbb{R}^{m\times m},\ U \in \mathbb{R}^{m\times d},\ b \in \mathbb{R}^m,\ c \in \mathbb{R}^m \end{array} \right\}. \end{aligned} \tag{3.2}$$

The approximation budget here is m, which is the width of the RNN, or the dimension of the hidden state sequence $\boldsymbol{h}$. Approximation theory of RNN investigates the ability of $\mathcal{H}_{\mathrm{RNN}}$ and $\{\mathcal{H}^m_{\mathrm{RNN}}\}$ to approximate appropriate target functional sequences.

It is often convenient to consider a continuous variant of the RNN, i.e. $\mathcal{T} = \mathbb{R}$. In this case, the RNN hidden state equation is now continuous in t, and it can be viewed as a time index. The only change is that we replace the difference equation (3.1) by the differential equation

$$\begin{aligned} \dot{h}(t) &= \sigma\big(Wh(t) + Ux(t) + b\big), \\ y(t) &= c^\top h(t), \end{aligned} \qquad t \in \mathbb{R}. \tag{3.3}$$

Besides theoretical advantages, some practical applications (e.g. irregularly-sampled time series) require a continuous-index model. The corresponding hypothesis space is analogous to Eq. (3.2) with Eq. (3.3) in place of Eq. (3.1).

A remark is in order on the choice of time-index for sequence approximation. Generally, $\mathcal{T}$ can be discrete or continuous, and bounded or unbounded, leading to four different

[1] There are notational variants in the literature, e.g. sometimes the index for the input is $t-1$ instead of t. Such minor variations do not affect approximation results.

settings. In addition, in each setting there is a choice of the norm that measures the approximation error. In the simplest case where $\mathcal{T}$ is bounded and discrete, the approximation problem is finite-dimensional. Beyond this setting, the choice of norm generally matters. For density-type results, the choice of discrete vs continuous $\mathcal{T}$ is usually not important, since they can be bridged by a discretization argument in one way and taking limits in the other. The distinction between bounded and unbounded $\mathcal{T}$ is however significant, and the latter generally requires more stringent conditions and is also more important for analyzing memory behavior that occurs at asymptotic regimes of $\mathcal{T}$. On the other hand, for Jackson/Bernstein-type theorems, there is a difference between discrete and continuous $\mathcal{T}$. Typically, approximation error estimates for a discrete $\mathcal{T}$ are grid-dependent, and do not readily translate to a uniform error estimate over all discrete grid partitions. In this sense, uniform-in-t estimates for the continuous case are stronger results, as they imply approximation rates for any grid using a discretization argument, given some regularity conditions on the sequences to allow one to estimate the discretization error.

3.2 Density-type results

As with most machine learning models, density-type results are the most basic and thus prevalent. Such results are minimal guarantees for the general applicability of a machine learning model. At the same time, the most theoretically interesting part about these results is the identification of appropriate target spaces $\mathcal{C}$ in which a particular $\mathcal{H}$ is dense.

Hidden dynamic functionals. By observing the RNN structure, it is natural to consider target functionals that are themselves defined via observations of a hidden dynamical system that has a compatible structure. For instance, in continuous time index case one can consider

$$\boldsymbol{x} \mapsto \boldsymbol{H}(\boldsymbol{x}) = \boldsymbol{y} \quad \text{with} \quad \begin{aligned} \dot{h}(t) &= f\big(h(t), x(t)\big), \quad h(t) \in \mathbb{R}^n, \\ y(t) &= g\big(h(t)\big), \qquad\quad h(-\infty) = 0, \end{aligned} \tag{3.4}$$

where $f : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ and $g : \mathbb{R}^n \to \mathbb{R}$. We may assume that f is Lipschitz and g is continuous so that $\boldsymbol{H}$ is well-behaved. The function g is called a readout map. Since the functions f, g parameterize a functional sequence via a hidden dynamical system, we call them hidden dynamic functional sequences, or $\mathcal{C}_{HD}$ for short. The discrete or bounded index cases are defined similarly. In the non-linear dynamics literature, Eq. (3.4) is often called a non-linear time-invariant system and the corresponding functional sequence $\boldsymbol{H}$ is referred to as a time-invariant filter. The term time-invariant (strictly, equivariant) highlights that $\boldsymbol{H}$ commutes with time-shifts. To see this, denote by $\boldsymbol{S}_\tau$ the shift operator $\boldsymbol{S}_\tau(\boldsymbol{x})(t) = x(t-\tau)$, then $\boldsymbol{H}$ satisfies $\boldsymbol{H} \circ \boldsymbol{S}_\tau = \boldsymbol{S}_\tau \circ \boldsymbol{H}$. However, in this survey we refrain from calling them time-invariant filters, because there may exist functional sequences that commute with time shifts, but are not readily written in the form Eq. (3.4), e.g. the shift functional sequence $H_t(\boldsymbol{x}) = \boldsymbol{S}_\tau(\boldsymbol{x})(t) = x(t-\tau)$.

Density-type results for $\mathcal{C}_{HD}$ are also called universal simulation, since it requires the approximate simulation of a dynamics driven by f and a readout map defined by g by

a RNN. Earlier results on hidden dynamic functional sequences focus on a bounded index set (see the survey of [73] and references therein, and also [18,54]). In these works, the main technique is to appeal to the universal approximation theory of feed-forward networks (e.g. [21]). The simple observation is that the right hand side of RNNs are feature maps of a fully connected network. Thus, by increasing m one can construct an approximation of f as

$$(h, x) \mapsto f(h, x) \approx (h_1, x) \mapsto \sigma\big(W(h_1, h_2)^\top + Ux + b\big), \tag{3.5}$$

where $h_1 \in \mathbb{R}^n$ and $h_2 \in \mathbb{R}^{m-n}$. The readout map g can be handled likewise. A similar approach is developed in [72] in the discrete-time setting, and [29] for simulating dynamics without inputs. Since the results concern a compact time interval, to approximate dynamics it is enough to approximate f. This is in general not true for the unbounded case, as the approximation error can be magnified by the dynamics.

To handle unbounded $\mathcal{T}$ (e.g. $\mathcal{T} = \mathbb{R}$), one strategy is to introduce some decay properties to the targets. One such property is the fading memory property (FMP) [10]. Let $\boldsymbol{x_1}, \boldsymbol{x_2}$ be bounded sequences indexed by $\mathbb{R}$, and let $\boldsymbol{H}$ be a sequence of causal, shift-equivariant (also called time-homogeneous) functionals. Here, causal means $H_t(\boldsymbol{x}) = H_t(\boldsymbol{x}_{(-\infty,t]})$ for all t. We say that $\boldsymbol{H}$ has the FMP if there is a monotonically decreasing function $\boldsymbol{w} : \mathbb{R}_+ \to (0, 1]$ such that for any $\epsilon > 0$ there exists $\delta > 0$ with

$$|H_t(\boldsymbol{x_1}) - H_t(\boldsymbol{x_2})| < \epsilon \text{ whenever } \sup_{s\in(-\infty,t]} |x_1(s) - x_2(s)|w(t - s) < \delta. \tag{3.6}$$

Intuitively, this says that two inputs sequences that differ more and more in their history ($t \to -\infty$) still produce similar outputs at the present. This is in effect requiring the memory of $\boldsymbol{H}$ to decay. Note that due to time-equivariance, it is enough to check this for just one t, say $t = 0$. Then, we can define a weighted norm on the space of semi-infinite sequences on $(-\infty, 0]$ by

$$\|\boldsymbol{x}\|_{\boldsymbol{w}} = \sup_{s\in(-\infty,0]} |x(s)w(-s)|. \tag{3.7}$$

Consequently, the FMP (Eq. (3.6)) is simply a continuity requirement of H_0 with respect to $\|\cdot\|_{\boldsymbol{w}}$. We denote by $\mathcal{C}_{\text{FMP}}$ the set of causal, shift-equivariant functional sequences satisfying the FMP. The FMP allows one to prove density on unbounded $\mathcal{T}$, e.g. in [33,35]. Indeed, the FMP property allows one to approximate $\boldsymbol{H} \in \mathcal{C}_{\text{FMP}}$ by a truncated version on a bounded interval. Then, approximation results can be deduced using methodologies for the bounded case. Note that the FMP is defined for general functional sequences, and is not limited to the form of hidden dynamic functional sequences. Thus, this idea can also be used to prove density for general functionals on unbounded index sets.

In the specific setting of hidden dynamic functional sequences, another technique for handling unbounded index sets was proposed in [37]. Here, the authors consider dynamics driven by f that satisfy a property called uniformly asymptotically incrementally stable. This roughly says that the flow maps of $\dot{h} = f(h, x)$ are uniformly continuous, uniformly in $\boldsymbol{x}$, and that $h(t)$ is independent of initial condition at large t. One can understand this as again a memory decay condition, as any initial condition on h is forgotten in the large time limit. This allows one to localize the approximation of f and g to a compact set,

which then allows one to appeal to standard approximation results from the feed-forward networks.

General functional sequences. Now, we turn to more general functional sequences. Since the RNN architecture (Eq. (3.2)) is causal and shift-equivariant, we should restrict our attention to target spaces satisfying the same properties. However, we no longer assume that these functional sequences admit a representation in the form of Eq. (3.4). For density-type results, this distinction is not important. This is because it is known that $\mathcal{C}_{HD}$ is dense in $\mathcal{C}_{\text{FMP}}$ in the norm $\|\boldsymbol{H}\| = \sup_{t\in\mathbb{R},x\in\mathcal{K}} |H_t(\boldsymbol{x})|$, where $\mathcal{K}$ is a bounded equicontinuous set in $C(\mathbb{R})$ [11, Theorem 2]; see also [36]. The idea relies on approximation of FMP functionals by a Volterra series [86]. The density can also be established without appealing to the Volterra series [35, Theorem 8]. Therefore, density-type results on $\mathcal{C}_{HD}$ can be passed onto $\mathcal{C}_{\text{FMP}}$, provided the norms are compatible. In the RNN case, this program is carried out in [34, 35]. However, we will see later that for Jackson-type results, the choice of target spaces is important: the rate of approximations generally depends on such choices.

It is also possible to construct a RNN approximation in $\mathcal{C}_{\text{FMP}}$ directly, without the need to use $\mathcal{C}_{HD}$ as an intermediate. For example, in [33] the authors first use the FMP to reduce the approximation problem to one over a finite, bounded index set, and then appeal to the density of fully connected neural network to obtain approximation. It remains then to construct a (large) RNN to represent the fully connected network. A similar result for stochastic inputs is proved in [32].

Many of the aforementioned density-results stem from the reservoir computing literature, where researchers are interested in studying systems such as the RNN, but with the internal weights (W, U, b in Eq. (3.3)) being random variables. This random version of the RNN is called an echo-state network (ESN). From the machine learning viewpoint, one can understand ESNs as an analogue of random feature models corresponding to RNNs. These models have the nice property that the hypothesis space is linear and training these networks is a convex problem, since only c needs to be trained. Previously mentioned results show existence of (W, U, b) and c to approximate each $\boldsymbol{H}$, but do not address the approximation of classes of $\boldsymbol{H}$ by choosing only c and using a common random realization of (W, U, b). The latter approximation problem is studied in [31], where a density result with some explicit error estimates is obtained. Here, the primary idea is to constrain target functionals to a subset of $\mathcal{C}_{\text{FMP}}$ whose Fourier transform has finite third moment. This builds on the idea of [6,7] where functions of this type (but with finite first and second moments) were shown to be approximated by feed-forward neural networks without suffering the curse of dimensionality. This is to be contrasted with a related line of work [25–27], which introduces a probabilistic definition of Barron-type functions via an expectation in place of a moment condition on its Fourier transform. In both cases, it is known that such functions can be approximated by randomly sampling neural network weights according to a distribution to achieve approximation. This is used in [31] to prove density for ESNs with random weights. We note that in general Barron function approximations, the random weight distributions depend on the target functions to be approximated, whereas in [31] the distribution of the reservoir weights is fixed as uniform. This comes with the cost of stronger regularity conditions, as we will discuss later.

3.3 Jackson-type results

Compared to density-type results, there are fewer Jackson-type results for RNNs. In the aforementioned work of [31], a quantitative error estimate can be obtained by a time-truncation argument in the discrete time index setting. Let $\boldsymbol{H}|_T$ denote the restriction of $\boldsymbol{H}$ to sequences of length $T+1$. Then, for each t we can identify $H|_T(t)$ with a function $H_{T,t}: \mathbb{R}^{d\times(T+1)} \to \mathbb{R}$. If one imposes additional regularity conditions by requiring $H_{T,t} \in \mathcal{W}^{k,2}$ for each t, then one can deduce an error estimate of the form

$$\inf_{\widehat{\boldsymbol{H}}\in\mathcal{H}^m_{\text{ESN}}} \mathbb{E}\left[\|\boldsymbol{H}-\widehat{\boldsymbol{H}}\|^2\right]^{1/2} \leq c_1 \frac{\|\boldsymbol{H}|_T\|_{W^{k,2}}}{m^{1/\alpha}} + c_2 \sum_{i=T+1}^{\infty} w(-i), \tag{3.8}$$

where $\alpha > 2$ and w is the weighting function used in the definition of $\mathcal{C}_{\text{FMP}}$. In particular, if we consider approximation on a bounded index set the last term vanishes, and we obtain an approximate Monte-Carlo rate $1/\sqrt{m}$. However, a caveat is that the smoothness requirement k for this estimate to hold increases linearly with dT, i.e. it becomes increasingly stringent on larger time intervals or input dimensions. In other words, this estimate is more useful for bounded index sets and low input dimensions.

In the setting of hidden dynamic functional sequences, a similar estimate is proved in [37] for unbounded index sets. The key assumption of uniformly asymptotically incrementally stable dynamics (c.f. the discussion in Section 3.2) is combined with the additional assumption that f, g are Barron-type functions. Then, one can obtain a Monte-Carlo error rate that decays as $1/\sqrt{m}$. The argument is a combination of the localization argument outlined previously for the density result, and the application of the results of [6,7] on the localized compact domain.

A general property of these results is the reliance on time truncation, thus the rate estimates do not explicitly account for the behavior on large time intervals. Jackson-type error estimates that directly operates on unbounded time domains are proved in the linear RNN case ($\sigma(z) = z$ and $b = 0$ in (3.3)) [55,56]. Let us call these hypothesis spaces $\mathcal{H}_{\text{L-RNN}}$ and $\{\mathcal{H}^m_{\text{L-RNN}}\}$. Observe that each $\widehat{\boldsymbol{H}} \in \mathcal{H}^m_{\text{L-RNN}}$ has the form

$$\widehat{H}_t(\boldsymbol{x}) = \int_0^\infty c^\top e^{Ws} U x(t-s) ds, \quad c \in \mathbb{R}^m, \quad W \in \mathbb{R}^{m\times m}, \quad U \in \mathbb{R}^{m\times d}. \tag{3.9}$$

Here, the input space considered is $\mathcal{X} = C_0(\mathbb{R}, \mathbb{R}^d)$, the space of continuous vector-valued sequences vanishing at infinity. We will also assume that W is Hurwitz (i.e. it is non-singular with eigenvalues having negative real parts), so that the dynamics is stable. In this case, one can check that each $\widehat{\boldsymbol{H}}$ is linear, continuous in the uniform norm and shift-equivariant (time-homogeneous). In addition, it is regular in the sense that if $\boldsymbol{x}_n(t) \to 0$ for almost every t then $\boldsymbol{H}(\boldsymbol{x}_n) \to 0$. It turns out that these conditions are sufficient conditions for functionals in $\mathcal{C}$ to be uniformly approximated by linear RNNs [56]. The idea is straightforward: one first shows that any linear functional sequence $\boldsymbol{H}$ satisfying these conditions admits a common Riesz representation

$$H_t(\boldsymbol{x}) = \int_{-\infty}^t \rho(t-s)^\top x(s) ds = \int_0^\infty \rho(s)^\top x(t-s) ds. \tag{3.10}$$

In other words, $\boldsymbol{H}$ and $\boldsymbol{\rho} \in L^1$ can be identified. Note that the application of Riesz representation is valid since $C_0(\mathbb{R}, \mathbb{R}^d)$ is taken as the input sequence space. In broader settings, e.g. $C(\mathbb{R}, \mathbb{R}^d)$ where input sequences need not decay at infinity, more assumptions is required for the existence of this representation. For example, [10, Theorem 5] shows that if $\mathcal{X} = C(\mathbb{R}, \mathbb{R})$, $\boldsymbol{H}$ admits the form (3.10) if and only if $\boldsymbol{H}$ has fading memory, in addition to the aforementioned assumptions. Now, comparing Eq. (3.9) and Eq. (3.10), linear RNN approximation of these functionals boils down to

$$|H_t(\boldsymbol{x}) - \widehat{H}_t(\boldsymbol{x})| \leq \|\boldsymbol{x}\|_{L^\infty} \|\boldsymbol{\rho} - \widehat{\boldsymbol{\rho}}\|_{L^1}, \tag{3.11}$$

where $\widehat{\rho}(s) = [c^\top e^{Ws} U]^\top$. Therefore, we may deduce approximation properties of targets by linear RNNs by approximation of functions in L^1 by exponential sums of the form $[c^\top e^{Ws} U]^\top$. The density of such exponential sums can be derived using the Müntz–Szász theorem [59].

Similarly, [56] further use this idea to prove a Jackson-type result for the error estimate. Here enters the crucial property of memory decay. There exists a vast literature on possible notions of memory decay for functional sequences, see e.g. [10, 32] and references therein. In the linear case, the following simple definition suffices. Let $\boldsymbol{e}_i = e_i \mathbf{1}_{t \geq 0}$, $i = 1, \ldots, d$ with e_i the unit vector in the i-th axis direction. We consider targets $\boldsymbol{H}$ such that there exist $\alpha \in \mathbb{Z}_+$, $\beta > 0$ such that

$$e^{\beta t} H_t^{(r)}(\boldsymbol{e_i}) = o(1), \quad t \to \infty, \quad i = 1, \ldots, d, \quad 1 \leq r \leq \alpha + 1. \tag{3.12}$$

Intuitively, these functionals forget input history at a rate of at least $e^{-\beta t}$. Thus, we may also understand them possessing an exponentially decaying memory. The main result in [55] is a Jackson-type error estimate

$$\inf_{\widehat{\boldsymbol{H}} \in \mathcal{H}^m_{\text{L-RNN}}} \|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \leq \frac{c_\alpha d \gamma}{\beta m^\alpha}, \quad \gamma = \sup_{t \geq 0} \max_{i=1,\ldots,d} \max_{r=1,\ldots,\alpha+1} \frac{\left| e^{\beta t} H_t^{(r)}(\boldsymbol{e}_i) \right|}{\beta^r}, \tag{3.13}$$

where $\|\boldsymbol{H}\| = \sup_t \sup_{\|\boldsymbol{x}\|_{L^\infty} \leq 1} |H_t(\boldsymbol{x})|$. Comparing with Eq. (2.4), we see that one obtains a similar rate characterized by the smoothness parameter α. The new phenomena is the assumption of exponential decaying memory in Eq. (3.12). The key insight here is as follows. If we assume, in addition to the usual smoothness requirements, that the memory of targets decay like an exponential (Eq. (3.12)), then we can efficiently approximate them using linear RNNs.

We remark here that this result demonstrates the importance of considering more general functional sequences than $\mathcal{C}_{HD}$ in establishing Jackson-type results. Assume instead that one considers hidden dynamic functional sequences with both f, g as linear functions, i.e.

$$f(h, x) = W_* h + B_* x, \quad g(h) = c_*^\top h, \quad W_* \in \mathbb{R}^{n \times n}, \quad U_* \in \mathbb{R}^{n \times d}, \quad c_* \in \mathbb{R}^n. \tag{3.14}$$

Then, the rate estimate becomes trivial: If $m \geq n$, then the approximation error is 0 and we have perfect representation. However, in practice it is generally not possible to know the

precise mechanism for the generation of the sequence data, and a theory should handle general functional sequences. From the Riesz representation (3.10) of general linear, causal and shift-equivariant target functional sequences, $\boldsymbol{\rho}$ can be any L^1 function, and may not in the form of an exponential sum. In this case, the approximation rate estimate becomes non-trivial.

We close the discussion by discussing the so-called curse of memory phenomenon identified in the above analysis observed in [55]. The density type results, including the linear RNN case, do not require the targets to have an exponentially decaying memory in the sense of Eq. (3.12). However, the rate estimate in Eq. (3.13) does have this requirement. The natural question is therefore, what if one has a slower memory decay rate? For example, we may replace Eq. (3.12) by

$$H_t^{(r)}(\boldsymbol{e_i}) \sim e^{-\beta t} \quad \longrightarrow \quad H_t^{(r)}(\boldsymbol{e_i}) \sim t^{-(r+\omega)}, \quad \omega > 0. \tag{3.15}$$

Then, a truncation argument in [56] shows that to obtain an approximation error of ϵ, a size of the RNN may need to grow exponentially, as $m \sim \epsilon^{-1/\omega}$. While this is not a lower bound for the optimal approximation error, it suggests that in sequence approximation, one may observe a very similar issue with approximating ordinary functions in high dimensions. There, it is known that the approximation budget required to achieve a prescribed approximation error grows like an exponential function of the dimension of the function domain. This is known as the curse of dimensionality. Here, the results suggests that in sequence approximation problems using RNNs, there lies a curse of memory. In particular, it affirms the empirical observations that RNNs usually perform well when memory in the system is small, but suffer in its performance for approximating long-term memory [8]. The result in Eq. (3.13) confirms the first part of the observation. The second part can be further demonstrated by optimization analysis [56] and also a Bernstein-type result, as we discuss next.

3.4 Bernstein-type results

Recall that Bernstein-type results deduce properties of targets assuming that they can be efficiently approximated by a hypothesis space. Known Bernstein-type results for RNNs are currently limited to linear functional sequences. With the same set-up as the Jackson-type theorem, [56] proves a Bernstein-type result, which we now describe.

Let us assume that we have a target functional sequence $\boldsymbol{H}$ such that it (and its derivatives in time) can be uniformly approximated by a sequence of linear RNNs. That is, we assume that there is a sequence $\widehat{\boldsymbol{H}}_m \in \mathcal{H}^m_{\text{L-RNN}}$ such that $\|\boldsymbol{H} - \widehat{\boldsymbol{H}}_m\| \to 0$ and that

$$\sup_{t\geq 0} \left| H_t^{(k)}(\boldsymbol{e}_i) - \widehat{H}_{m,t}^{(k)}(\boldsymbol{e}_i) \right| \to 0, \quad k = 1, \ldots, \alpha + 1. \tag{3.16}$$

Then, under additional technical conditions, there must exist a $\beta > 0$ such that

$$e^{\beta t} H_t^{(r)}(\boldsymbol{e_i}) = o(1), \quad t \to \infty, \quad i = 1, \ldots, d, \quad 1 \leq r \leq \alpha + 1. \tag{3.17}$$

In other words, a target can be effectively approximated by linear RNNs only if it has exponentially decaying memory. This is in a sense a partial converse to the Jackson-type

result in Eq. (3.13). Together, it shows that, at least in the linear setting, effective RNN approximation occurs if and only if the target functional sequence has an exponentially decaying memory pattern. Bernstein-type results can assist in designing architectures for sequence modelling: if a model aims to model a sequential relationship whose memory pattern does not decay like an exponential, then it is necessary to go beyond the RNN setting due to the limitations posed by the inverse approximation result.

At the end of Section 4.1, we discuss an example given in [42] where the target functional sequence does not have an exponentially decaying memory, and alternative architectures such as dilated convolutions are shown to be more effective than RNNs.

We end the section on RNNs approximation by discussing some of its variants. In the practical literature, a number of generalizations of the simple RNN hypothesis space (Eq. (3.2)) have been proposed. Examples include the long-short term memory (LSTM) network [39] and gated recurrent units (GRU) [16]. Density-type results for these networks can be directly deduced since they often include the classical RNN as a special case by a proper choice of its trainable parameters. In some cases (e.g. normalized RNNs in [72], and deep variants with fixed width in [41]), additional analysis is required to establish density. However, rate estimates of Jackson-type or inverse theorems of Bernstein-type (different from classical RNNs) are generally not known for these more complex structures, and is an interesting direction of future work.

4 Other architectures

Let us now expand our discussion to models beyond the RNN model family. Many of these architectures are proposed or popularized in fairly recent years. A partial but important motivations for developing these alternative model architectures is precisely the limitations with respect to memory we have discussed in Section 3. Very often in practical applications, we want to model sequence relationships having long and irregular memory patterns. For example, in machine translation tasks, an output word at the end of the sentence in one language may depend on the very first word in the corresponding sentence in another language. Moreover, the number of words in the original and translated sentences are often not the same. For these reasons, a variety of alternative models to the RNN have been proposed. Each of them are competitive in different domains of application. The subsequent discussions will highlight a number of such examples.

However, to concretely understand the gains of using alternative architectures to RNN, it is necessary to develop some theoretical understanding of their comparison. For example, can an alternative architecture such as a convolutional-based architecture overcome the curse of memory related to RNNs? This often requires the developments of Jackson-type estimates in similar approximation settings, which tells us precisely which functional sequences are easy to approximate under a particular hypothesis space corresponding to a model architecture of interest.

4.1 Convolution-based architectures

We begin with results for convolutional-based architectures. While convolutional neural networks (CNN) were originally developed for computer vision applications [50], tempo-

ral versions of the CNNs have been shown to be effective in many sequence modelling tasks [5]. Since convolution operations are easier to describe using a discrete index set, we shall assume throughout this subsection that $\mathcal{T} = \mathbb{Z}$.

The basic building block of temporal CNNs is the causal dilated convolution operation

$$(\boldsymbol{u} *_l \boldsymbol{v})(t) = \sum_{s \geq 0} u(s)^\top v(t - ls), \quad l \in \mathbb{Z}_+. \tag{4.1}$$

Note that the summation is taken over $s \geq 0$ to ensure causality, meaning that the outcome at time t depends only on the past information. When $l = 1$, this is the usual convolution. Dilations $l \geq 2$ result in larger receptive fields with the same number of parameters, and are hence useful in processing long sequences. For example, successful temporal CNN architectures, including the WaveNet [84] and the TCN [51], contain stacks of dilated convolutions with increasing dilation rates.

We can write a general dilated temporal CNN model with K layers and M channels at each layer as

$$\begin{aligned}
&\boldsymbol{h}_{0,i} = \boldsymbol{x}_i, \\
&\boldsymbol{h}_{k+1,i} = \sigma\left(\sum_{j=1}^{M_k} \boldsymbol{w}_{kji} *_{d_k} \boldsymbol{h}_{k,j}\right), \quad i = 1, \ldots, M_{k+1}, \quad k = 0, \ldots, K-1, \\
&\widehat{\boldsymbol{y}} = \boldsymbol{h}_{K,1},
\end{aligned} \tag{4.2}$$

where $M_0 = d$ is the input dimension, $M_K = 1$ is the output dimension. $M_k = M$ is the number of channels at layer k for $k = 1, \ldots, K-1$.

Here, $\boldsymbol{x}_i$ is the scalar sequence corresponding to the i-th element of the vector sequence $\boldsymbol{x}$, and $\boldsymbol{w}_{kji}$ is the convolutional filter at layer k, mapping from channel j at layer k to channel i at layer $k+1$. A common choice for the dilation rate in applications is $d_k = 2^K$, so we adopt this choice for the subsequent exposition. Furthermore, for establishing approximation results it is sufficient to assume that the support of each filter $\boldsymbol{w}_{kji}$ is 2, since convolutional filters of large sizes include this case. This gives rise to the temporal CNN hypothesis space

$$\mathcal{H}_{\text{CNN}} = \bigcup_{K,M} \mathcal{H}_{\text{CNN}}^{(K,M)} = \bigcup_{K,M} \Big\{ \boldsymbol{x} \mapsto \widehat{\boldsymbol{y}} \text{ in Eq. (4.2)} \Big\}. \tag{4.3}$$

Density-type results have been studied for general CNNs mostly for two-dimensional image applications, and some of them can be adapted to the one-dimensional, causal case here. For brevity, we will not give an exhaustive list of this literature. We mention however that most existing results are not directly applicable to the sequence modelling case due to the shift-equivariant requirement. For example, the works of [66, 67, 88] consider approximating general functions, and shift-equivariance is violated at the boundaries. Density results for fully convolutional cases [52, 58, 69, 80] are more relevant for the present application. Nevertheless, due to the nature of image data having finite supports, none of these results consider an unbounded index set. However, for sequence approximation, the problem of memory should be studied precisely on unbounded index sets. If we assume

some form of memory decay such as the FMP, then a truncation argument can be used to show that the temporal CNN hypothesis space is dense in sequence spaces (e.g. ℓ^p), as a corollary of these results.

For Jackson-type theorems, the current understanding is again limited to the simple but interesting case of linear temporal CNNs, i.e. $\sigma(z) = z$. This gives the linear temporal CNN hypothesis space

$$\mathcal{H}_{\text{L-CNN}} = \bigcup_{K,M} \mathcal{H}^{(K,M)}_{\text{L-CNN}} = \Big\{ \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = \sum_{s=0}^{\infty} \widehat{\rho}(s)^\top x(t-s) \Big\}, \tag{4.4}$$

where $\widehat{\boldsymbol{\rho}}$ is a finitely-supported sequence determined by the filters $\{\boldsymbol{w}_{kji}\}$

$$\widehat{\boldsymbol{\rho}}_i = \sum_{i_1,\dots,i_{K-1}} \boldsymbol{w}_{K-1,i_{K-1},1} *_{2^{K-1}} \boldsymbol{w}_{K-2,i_{K-2},i_{K-1}} *_{2^{K-2}} \cdots *_2 \boldsymbol{w}_{0,i,i_1}. \tag{4.5}$$

Observe the striking similarity of (4.4) and linear RNN hypothesis space (3.9). The key difference is that in the RNN case, the sequence $\widehat{\boldsymbol{\rho}}$ is an exponential sum with infinite support, whereas in the case of CNNs it is a sum of repeated dilated convolutions resulting in a finite support. This in turn leads to, as investigated in [42], vastly different Jackson-type estimates. In particular, one can identify different approximation spaces that suggests how RNN and CNN approximation differ when modelling sequence relationships.

Concretely, [42] proved the following Jackson-type estimate for linear, causal and shift-equivariant functional sequences $\boldsymbol{H}$:

$$\inf_{\widehat{\boldsymbol{H}} \in \mathcal{H}^{(K,M)}_{\text{L-CNN}}} \|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \leq G\big(KM^{1/K} - K\big) C_1(\boldsymbol{H}) d + C_2(\boldsymbol{H}, K). \tag{4.6}$$

Recall that M is the number of convolution filters at each layer and K is the number of layers. Together, (M, K) control the complexity of the CNN hypothesis space. The function $G : \mathbb{R} \to \mathbb{R}$ is a non-increasing function tending to 0, to be explained later.

Let us now clarify the form of C_1, C_2. Let $\boldsymbol{\rho}$ be the Riesz representation of $\boldsymbol{H}$,

$$H_t(\boldsymbol{x}) = \sum_{s \geq 0} \rho(s) x(t-s). \tag{4.7}$$

Then, $C_2(\boldsymbol{H}, K) = \|\boldsymbol{\rho}_{[2^K,\infty)}\|_{\ell^2}$ is determined by the rate of decay of the memory of the target functional sequence. In particular, C_2 decays at least exponentially in the depth of the neural network (K), even if the target does not possess memory decay. The term $C_1(\boldsymbol{H})$ is a complexity measure of the target functional sequence, determined by the effective rank of $\boldsymbol{H}$ after a tensorisation transformation. Let us motivate its definition by an example. Set $d = 1$ and suppose the goal is to model a target functional sequence

$$H_t(\boldsymbol{x}) = r_0 x(t) + r_1 x(t-1) + r_2 x(t-2) + r_3 x(t-3), \quad r_s \in \mathbb{R}. \tag{4.8}$$

In this case, the Riesz representation for $\boldsymbol{H}$ has support 4, i.e. $\boldsymbol{\rho} = (r_0, r_1, r_2, r_3)$. A temporal CNN approximates $\boldsymbol{\rho}$ via product-sums in the form of (4.5). Let us take $K = 2$ and $M = 1$. Then, notice that we are seeking the approximation of

$$\boldsymbol{\rho} = (r_0, r_1, r_2, r_3) \quad \text{by} \quad \widehat{\boldsymbol{\rho}} = (w_{0,0}, w_{0,1}) *_2 (w_{1,0}, w_{1,1}), \tag{4.9}$$

which we can rewrite in matrix form as the approximation of

$$\mathbf{T}(\boldsymbol{\rho}) = \begin{pmatrix} r_0 & r_1 \\ r_2 & r_3 \end{pmatrix} \quad \text{by} \quad \mathbf{T}(\widehat{\boldsymbol{\rho}}) = \begin{pmatrix} w_{0,0} \\ w_{0,1} \end{pmatrix} \begin{pmatrix} w_{1,0} & w_{1,1} \end{pmatrix}. \tag{4.10}$$

Then, the approximation error becomes clear. If $\mathbf{T}(\boldsymbol{\rho})$ is rank 1, then it can be represented exactly by the 2-layer CNN with channel size 1. Otherwise, there will be an approximation error, and the optimal approximation error is the second singular value of $\mathbf{T}(\boldsymbol{\rho})$ as a consequence of the Eckart-Young theorem.

This argument can be generalized to any K and M. For $K \geq 3$ the reshaping operation $\mathbf{T}(\cdot)$ acting on a length 2^K sequence produces an order-K tensor of size 2 in each dimension,

$$\mathbf{T}(\boldsymbol{\rho}_{[0,2^K]})_{i_1,\ldots,i_K} = \rho_{[0,2^K]} \left(\sum_{j=1}^{K} i_j 2^{j-1} \right), \quad i_j \in \{0,1\}. \tag{4.11}$$

Then, a temporal CNN approximates this tensor as a sum of rank 1 tensors. The optimal approximation error is hence a consequence of an Eckart-Young type theorem for higher-order singular value decomposition (HOSVD) [46]. This motivates the definition of an approximation space that depends on the tail of the singular value sequence. Let us now make this more precise. We fix a CNN of depth K, and consider the tensorisation $\mathbf{T}(\boldsymbol{\rho}_{[0,2^K]})$. In the theory of HOSVD [46], this tensor has $2K$ singular values

$$\sigma_1^{(K)} \geq \sigma_2^{(K)} \geq \cdots \geq \sigma_{2K}^{(K)} \geq 0, \tag{4.12}$$

the first K of which are equal and redundant. The last K singular values determine the error of low rank approximation of this tensor, much in the same way as ordinary singular value decay rates determine the accuracy of low rank approximation of matrices. Thus, we may consider specifying some decay rate G so that the tail sum of singular values (which corresponds to low rank approximation error) satisfies

$$\left(\sum_{i=s+K}^{2K} |\sigma_i^{(K)}|^2 \right)^{1/2} \leq cG(s) \tag{4.13}$$

with $G(s) \to 0$ as $s \to \infty$. Now, we can build an approximation space by considering target functional sequences whose Riesz representation $\boldsymbol{\rho}$ satisfies the following property: for each K, the singular value tail sum of $\mathbf{T}(\boldsymbol{\rho}_{[0,2^K]})$ has a decay rate of at least G (Eq. (4.13)). Then, the error of low rank approximation of these functional sequences can be described by G. This leads to the definition of a complexity measure in [42] of the form

$$C_1(\boldsymbol{H}) = \inf \left\{ c : \left(\sum_{i=s+K}^{2K} |\sigma_i^{(K)}|^2 \right)^{1/2} \leq cG(s),\ s \geq 0,\ K \geq 1 \right\}, \tag{4.14}$$

and G is a specified rate of decay of the singular values. The Jackson-type rate estimate in Eq. (4.6) then follows from the fact that the maximum rank of a CNN with K layers

and M channels is at least $KM^{1/K}$. The class of functional sequences where C_1 is finite defines an approximation space (with respect to G) of sufficiently regular functional sequences that admits efficient approximation by temporal CNNs. This is analogous to the characterization of classical smoothness spaces by the decay rate of series coefficients, e.g. wavelet coefficients [62]. Here, we can understand $C_1(\boldsymbol{H})$ as a measure of how easy it is to approximate $\boldsymbol{H}$ by tensor product-sums. In particular, it can be shown [42] that if $\boldsymbol{H}$ has a sparse Riesz representation (memory), then it has small $C_1(\boldsymbol{H})$. This supports the empirical observation that temporal CNNs excel in applications such as text-to-speech [84], where such sparsity patterns are expected.

Let us now contrast this insight to that obtained for the RNN, which excel at modelling memory patterns that are exponentially decreasing, but not necessarily sparse. Consider a target functional sequence with Riesz representation

$$\rho(t) = \delta\big(t - 2^{K_0}\big) = \begin{cases} 1, & t = 2^{K_0}, \\ 0, & t \neq 2^{K_0}. \end{cases} \tag{4.15}$$

This corresponds to a shift operation, where the output is the result of shifting the input by 2^{K_0} units. Observe that this target functional sequence is inside the temporal CNN hypothesis space, hence it can be exactly represented by setting $K = K_0$ and $M = 1$. However, when K_0 is large, it becomes increasingly difficult for a power sum $u(t) = c_0 + \sum_{i=1}^{m} c_i \gamma_i^t$ to approximate this function. The form of u here is a simplified discrete analogue of the exponential sum in Eq. (3.9). For any such u, we have the following property due to [28]:

$$m \geq \frac{t}{2 \sup_{s \in [0, 2t+2]} u(s)} |u(t+1) - u(t)|. \tag{4.16}$$

Since $\rho(t)$ has a sudden change at $t = 2^{K_0}$, $u(t)$ need at least $2^{K_0 - 1}$ terms to achieve approximation, making it challenging for a RNN to learn this target. Conversely, there exists targets which are easily approximated (in fact, exactly represented) by $\mathcal{H}_{\text{RNN}}$ but have high complexity when approximated by $\mathcal{H}_{\text{CNN}}$ [42]. These Jackson-type results highlight the interesting differences between the RNN and the CNN architectures with respect to the types of sequential relationships they are adapted to approximating.

4.2 Encoder-decoder architectures

Encoder-decoder architectures [16, 17, 43, 76] are a class of sequence to sequence models where an encoder first maps the input sequence into a fixed-sized context vector, and then a decoder maps the context vector into the output sequence. The development of encoder-decoder models was motivated by the need to handle input and output sequences with varying lengths. The encoder-decoder architecture is flexible and allows for the use of various configurations for the encoder and decoder components.

We consider the simplest setting where the encoder and the decoder are both recurrent networks [17, 76]. This has the advantage that we can compare the results here with those in ordinary RNNs in Section 3. The RNN encoder-decoder (REncDec) architecture

(ignoring bias for simplicity) can be written as

$$\begin{aligned} \dot{h}(s) &= \sigma_E\big(Wh(s) + Ux(s)\big), \quad v = Qh_0, \quad s \le 0, \\ \dot{g}(t) &= \sigma_D\big(Vg(t)\big), \qquad\qquad g_0 = Pv, \\ \widehat{y}(t) &= c^\top g(t), \qquad\qquad t \ge 0, \end{aligned} \tag{4.17}$$

where $W \in \mathbb{R}^{m\times m}$, $U \in \mathbb{R}^{m\times d}$, $Q \in \mathbb{R}^{N\times m}$, $V \in \mathbb{R}^{m\times m}$, $P \in \mathbb{R}^{m\times N}$ and $c \in \mathbb{R}^m$. Sequences $\boldsymbol{h}$ and $\boldsymbol{g}$ are the RNN-type hidden states corresponding to the encoder and decoder dynamics, respectively. The encoder is first applied to the entire input sequence $\boldsymbol{x}$ in order to produce a fixed-size context vector v, which is the final hidden state of the encoder. This context vector summarizes the input sequence. The vector is then utilized as the initial state of the decoder, which generates an output at each time step. This defines sequential relationship between two semi-infinite sequences, with the input $\boldsymbol{x}$ having support in $(-\infty, 0]$ and the output $\boldsymbol{y}$ having support in $[0, \infty)$. The complexity of these maps (approximation budget) is controlled by the RNN width m and context vector size N.

Approximation properties of the REncDec architecture are investigated in detail for the linear case (σ_E, σ_D are identity maps) in [57]. For simplicity of presentation, we take $d = 1$, corresponding to scalar input sequences. Then, one can rewrite the REncDec hypothesis space as

$$\mathcal{H}_{\text{L-REncDec}} = \bigcup_{m,N} \mathcal{H}^{(m,N)}_{\text{L-REncDec}} = \bigcup_{m,N} \left\{ \widehat{\boldsymbol{H}} : \widehat{H}_t(\boldsymbol{x}) = \int_0^\infty \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s)x(-s)ds \right\}, \tag{4.18}$$

where one may recall that m is the width of the RNNs used for the encoder and the decoder, and 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

$$\widehat{\psi}_n(t) = \left(\sum_{i,j=1}^{m} c_i P_{jn} \left[e^{Vt}\right]_{ij} \right), \quad \widehat{\phi}_n(t) = \left(\sum_{i,j=1}^{m} u_i Q_{nj} \left[e^{Wt}\right]_{ji} \right). \tag{4.19}$$

Since the REncDec architecture maps sequences of disjoint support, it is no longer meaningful to consider time-homogeneity and causality. Indeed, causality is always satisfied and time-homogeneity is not satisfied. This is the case by design: the REncDec architecture is used to model sequential relationships without the shift-equivariant condition. Consequently, the target functional sequences considered here are only assumed to be continuous and linear. In this case, the Riesz representation of these targets take the form

$$H_t(\boldsymbol{x}) = \int_0^\infty \rho(t,s)^\top x(-s)ds, \quad t \ge 0. \tag{4.20}$$

This is a more general form where $\boldsymbol{\rho}$ depends on two temporal indices t (outputs) and s (inputs) simultaneously.

The density of the hypothesis space (4.18) in the space of sufficiently regular continuous linear functional is established in [57]. This result follows from the observation that we now seek approximations of $\rho(t,s)$ via a product of two exponential sums. Hence, one

may follow essentially the same approach as in the RNN case to prove density. More interestingly, Jackson-type estimates can also be derived. In particular, one has the following approximation rate under similar settings as in Eq. (3.13):

$$\|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \leq \frac{C_1(\alpha)\gamma}{\beta^2 m^\alpha} + C_2(\boldsymbol{H}, N), \tag{4.21}$$

where the meaning of various constants are defined similarly as in Eq. (3.13). Observe that the first term is similar to the RNN rate (3.13), as both the encoder and decoder are implemented using RNNs. The estimate $C_2(\boldsymbol{H}, N)$ highlights the new complexity measure associated with encoder-decoder architectures, since N is the complexity of the context (coding) vector that acts as the only intermediary between the encoder and decoder components. To see what the complexity measure may be, let us compare Eq. (4.19) and Eq. (4.20). Observe that approximating a target $\boldsymbol{H}$ simply amounts to approximating its Riesz representation $\boldsymbol{\rho}$ by a tensor-product summation of the form

$$\widehat{\rho}(t,s) = \sum_{n=1}^{N} \widehat{\psi}_n(t)\widehat{\phi}_n(s). \tag{4.22}$$

One may immediately notice that this is a rank-N approximation of a two-variable function as sums of products of univariate functions. The optimal approximation is obtained through the proper orthogonal decomposition (POD) [13], which is an infinite-dimensional version of the optimal low rank approximation of matrices via truncated singular value decomposition. In fact, we may write the formal POD expansion for $\boldsymbol{\rho}$ as

$$\rho(t,s) = \sum_{n=1}^{\infty} \sigma_n \psi_n(t)\phi_n(s), \tag{4.23}$$

where σ_n are the singular values and ψ_n, ϕ_n are the left and right singular vectors (functions). This is called a temporal product structure in [57]. Then, an analogue of the classical Eckart-Young theorem implies that the optimal approximation error is simply the tail-sum of the singular values. This is precisely the estimate in C_2, i.e.

$$C_2(\boldsymbol{H}, N) \propto \left(\sum_{n=N+1}^{\infty} \sigma_n^2 \right)^{1/2}. \tag{4.24}$$

This is considered as the effective rank of the target, and the Jackson-type estimate in Eq. (4.21) says that a target with low effective rank can be approximated efficiently with REncDec (with small context vector). Note that this notion of rank is different from the tensorisation rank discussed for CNNs in Section 4.1. The concept of effective rank of the sequential relationship under temporal products is similar to that in linear algebra, where the rank of a matrix is the dimension of its range space. This definition can be extended to apply to sequential relationships. Fig. 4.1 illustrates this idea: A low rank temporal relationship results in a more regular output sequence. In particular, local perturbations to the input sequence result into global perturbations of the output sequence. This is very

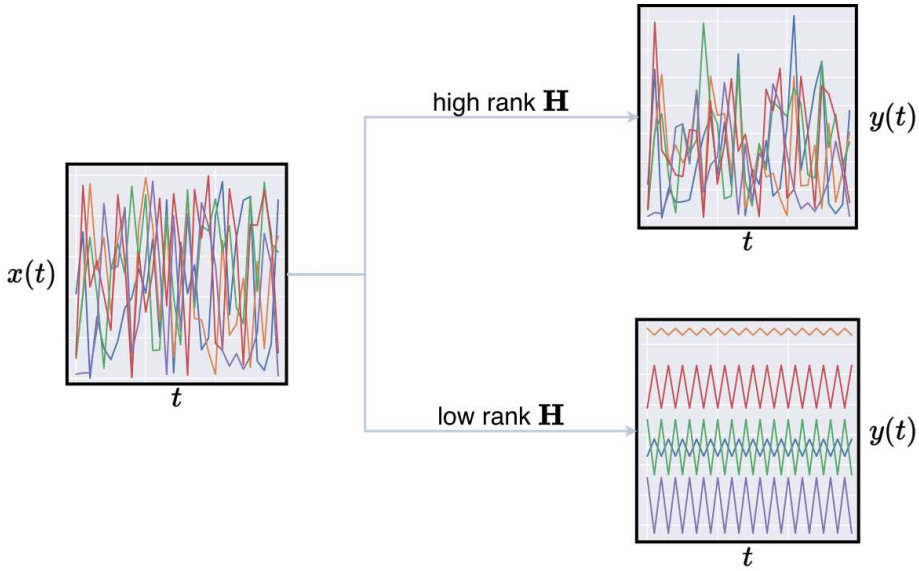

Figure 4.1: Schematic illustration of a high-rank vs low-rank sequential relationship under the temporal product structure. A dataset of input sequences (left) are fed into a functional sequence producing the corresponding output sequences (right). The top (resp. bottom) right plot shows the resulting sequence of a high-rank (resp. low-rank) relationship. Observe that the high rank relationship yields a complex and input-sensitive temporal structure. In contrast, the outputs of the low rank relationship exhibit greater regularity, with only macroscopic structures present. It is precisely the latter that REncDec is adapted to model.

different from both the CNN and the RNN architectures, and the Jackson-type estimate makes this difference precise.

Currently known approximation results only focus on linear RNN encoder-decoder. However, the density result can be extended to non-linear cases by following the same approaches outlined in Section 3, due to the similarity with RNNs. The rate estimate is less straightforward to extend to non-linear activations. Nevertheless, one may expect that the uncovered relationship between the size the context vector and a low-rank type of approximation should hold generally for encoder-decoder architectures. This is because in all such structures, the input and output sequence (both may be infinite-dimensional) only communicate through a bottle-neck coding vector (finite dimensional), and thus the approximation should be viewed as a generalized low-rank approximation.

4.3 Attention-based architectures

In the final part of this section, we discuss approximation theory for the growingly popular attention-based architectures. The attention mechanism was first proposed in [4] in the context of RNNs. Subsequently, it was employed in a variety of practical network architectures. The attention mechanism, much like the encoder-decoder mechanism, is a component that can be incorporated into existing models. Since its introduction, the attention mechanism has become popular tool in applications, including natural language

processing [85] and computer vision [24]. In fact, one of the most successful model families, the Transformer [85], is based on both the attention mechanism and the encoder-decoder mechanism. However, our theoretical understanding of the attention mechanism is currently limited, particularly with regard to its approximation properties.

Let us focus our discussion on the Transformer family of attention-based architectures. Currently established approximation results include the universal approximation capabilities of Transformer networks [81] and its sparse variants [82]. It is important to note that in this context, the term Transformer refers specifically to the encoder component of the original architecture proposed in [85].

In order to study the Transformer under the sequence approximation setting, it is convenient to restrict the index set $\mathcal{T}$ to a finite set $\mathcal{T} = \{1, 2, \ldots, \tau\}$. Then, the approximation problem becomes finite-dimensional. The reason is as follows. The use of position encoding in Transformer networks is necessary to eliminate their permutation equivariance (we will show this exactly later). Position encoding is a sequence $\boldsymbol{e}$ where $t \mapsto e(t)$ is a fixed or trainable function, independent of $\boldsymbol{x}$. The sequence $\boldsymbol{e}$ preserves the information of temporal order. For training convenience, the length of this encoding is fixed. As a result, Transformer networks are unable to directly process infinite-length sequences, unlike RNNs and CNNs based architectures.

The simplest transformer block consists of the following components,

$$\begin{aligned} \mathrm{Attn}(\boldsymbol{x})(t) &= x(t) + \sum_{i=1}^{q} W_o^i \sum_{s=1}^{\tau} \sigma\left[\left(W_Q^i x(t)\right)^\top W_K^i x(s)\right] W_V^i x(s), \\ \mathrm{Trans}(\boldsymbol{x})(t) &= \mathrm{Attn}(\boldsymbol{x})(t) + f\big(\mathrm{Attn}(\boldsymbol{x})(t)\big), \end{aligned} \tag{4.25}$$

where $W_Q^i, W_K^i, W_K^i \in \mathbb{R}^{m\times d}$, $W_o^i \in \mathbb{R}^{d\times m}$. Here, $\mathrm{Attn}(\boldsymbol{x})$ is the attention block, σ is a normalization usually taken as the softmax function, and τ is the maximum input sequence length. The attention mechanism produces an output which is subsequently fed into a common trainable feed-forward network f, pointwise in time. This constitutes a Transformer block. Define the Transformer hypothesis space by

$$\mathcal{H}_{\mathrm{Trans}}^{(m_1,m_2,q,l)} = \left\{\widehat{\boldsymbol{H}} : \widehat{\boldsymbol{H}} \text{ is a composition of } l \text{ Transformer blocks } t^{(m_1,m_2,q)}\right\}, \tag{4.26}$$

where $t^{(m_1,m_2,q)} = \mathrm{Trans}(\cdot)$ is a Transformer block defined in (4.25), m_1 is the trainable dimension of the attention block (total degrees of freedom of W_o, W_Q, W_K, W_V), q is the number of attention heads and m_2 is the dimension of the trainable parameters in the pointwise feed-forward network f. In [82], a sparse variants is defined, where the W_Q^i matrix in the attention block satisfies certain sparsity conditions. We denote the sparse Transformer hypothesis space by

$$\mathcal{H}_{\mathrm{SpTrans}}^{(m_1,m_2,q,l)} \subset \mathcal{H}_{\mathrm{Trans}}^{(m_1,m_2,q,l)}, \tag{4.27}$$

which is a subset of the Transformer hypothesis space.

We start with density results for the Transformer. First, note that without position encoding, the Transformer hypothesis space is permutation equivariant. Concretely, let p

be a permutation of the sequence index, which is a bijection on $\{1, \ldots, \tau\}$. For a sequence $\boldsymbol{x}$, we denote by $\boldsymbol{x} \circ p$ the permuted sequence $[\boldsymbol{x} \circ p](t) = x(p(t))$. A functional sequence $\boldsymbol{H}$ is said to be permutation equivariant if for all p and $\boldsymbol{x}$ we have $\boldsymbol{H}(x \circ p) = \boldsymbol{H}(\boldsymbol{x}) \circ p$. We can check that the Transformer block (4.25) is permutation equivariant if one does not perform positional encoding. This certainly limits approximation properties, and thus we hereafter assume that a fixed position encoding is added to the input $\boldsymbol{x}$, such that the model input becomes $\boldsymbol{x} + \boldsymbol{e}$.

In [81, 82], a density results for the Transformer is proved under the following conditions. Assuming the target $\boldsymbol{H}$ is continuous, and the input sequence space is uniformly bounded, then $\boldsymbol{H}$ can be approximated by

$$\widehat{\boldsymbol{H}} \in \bigcup_{l} \mathcal{H}_{\text{SpTrans}}^{(1,4,2,l)} \subset \bigcup_{l} \mathcal{H}_{\text{Trans}}^{(1,4,2,l)}. \tag{4.28}$$

This result is proved by a special construction. First, one uses a stack of attention blocks to achieve the following condition:

1. For any input $\boldsymbol{x}$, the value of the output $\tilde{\boldsymbol{x}}$ are all distinct.
2. For all inputs $\boldsymbol{x_1}, \boldsymbol{x_2}$ such that $\boldsymbol{x_1} \neq \boldsymbol{x_2}$, their outputs $\tilde{\boldsymbol{x}}_{\boldsymbol{1}}$ and $\tilde{\boldsymbol{x}}_{\boldsymbol{2}}$ have no common value.

These conditions can be understood as for each $t, \tilde{x}(t)$ captures the information of the entire input sequence. Next, a deep stack of pointwise feed-forward blocks are constructed to map each $\tilde{x}(t)$ to the desired output. This construction results in a deep Transformer architecture with a small width.

However, this construction is not generally how Transformer operates, since the first part of the construction is an attention-only network, which is shown to degenerate quickly [23]. In a similar vein, several studies such as [19, 53] have demonstrated that a Transformer can represent a CNN through careful parameterization. Therefore, density results from CNN imply the density of the transformer. Again, there is little empirical evidence that the Transformer behaves like a CNN in applications.

The ability for the Transformer to mimic other architectures is not surprising, since it has many highly flexible components (encoder-decoders, fully connected networks, attention mechanisms) that can be carefully, but often artificially, adjusted to represent other known architectures as a special case.

In fact, we give here another example of such a representation that, to the best of our knowledge, has not been reported in the literature, but is straight-forward to derive. We can show that a two-layer Transformer can mimic the form of a generalized Kolmogorov representation theorem [68]. This result states that for d dimensional compact sets $I_1, \ldots, I_\tau \subset \mathbb{R}^d$, any continuous $\boldsymbol{H} : I_1 \times \cdots \times I_\tau \to \mathbb{R}$ can be written as

$$H(\boldsymbol{x}) = \sum_{q=0}^{2d\tau} \Phi_q \left(\sum_{s=1}^{\tau} \phi_{q,s}\big(x(s)\big) \right), \tag{4.29}$$

where Φ_q and $\{\phi_{q,s}\}$ are continuous functions. It is possible to design a two-layer Transformer exhibiting a similar form, implying density. For simplicity, we consider $d = 1$ and

only the output at H_t when $t = 1$. The general case can be constructed similarly. One can ensure that with position encoding, a pointwise feed-forward function is able to apply different mappings at each temporal index position. To see this, observe that for a collection of continuous functions $f_i : [0,1]^d \to \mathbb{R}$, $i = 1, \ldots, \tau$, we can find vectors $\{e_i\}$ and a continuous function $F : \mathbb{R}^d \to \mathbb{R}$ such that $F(x + e_i) = f_i(x)$. Now suppose we have an input sequence $\boldsymbol{x}$.

- Layer 1. We set $W_o^i = 0$ in the attention block, so that the input directly goes into the pointwise feed-forward block. From the previous discussion, the pointwise feed-forward network can be constructed to give an output $y^{(1)} : \mathbb{R} \to \mathbb{R}^{2\tau+1}$, such that $y^{(1)}(s) = c_s[\widehat{\phi}_{0,s}(x(s)), \ldots, \widehat{\phi}_{2\tau,s}(x(s))]^\top$, with $c_1 = 1/2$ and $c_s = 1$ when $s > 1$. Due to the density of feed-forward neural networks, each $\widehat{\phi}_{j,s}$ can be chosen to approximate any continuous function.

- Layer 2. In the attention block, by letting $W_K^i = 0$, the softmax function gives a constant output where $\text{softmax}[(W_Q^i x(t))^\top W_K^i x(s)] \equiv 1/\tau$. Let $W_V^i = I, W_o^i = \tau I$ and $h = 1$, then we have $\text{Attn}(\boldsymbol{y}^{(1)})(1) = \sum_{s=1}^{\tau} y^{(1)}(s)$. Hence, the final output after the feed-forward network with linear readout $c^\top = (1, \ldots, 1) \in \mathbb{R}^{2\tau+1}$ gives

$$\begin{aligned} H_1(\boldsymbol{x}) &= c^\top \widehat{\Phi}\big(\text{Attn}(\boldsymbol{y}^{(1)})(1)\big) \\ &= \sum_{q=0}^{2\tau} \widehat{\Phi}_q \left(\sum_{s=1}^{\tau} \left[y^{(1)}(s) \right]_q \right) \\ &= \sum_{q=0}^{2\tau} \widehat{\Phi}_q \left(\sum_{s=1}^{\tau} \widehat{\phi}_{q,s}(s) \right), \end{aligned} \tag{4.30}$$

  where $\widehat{\Phi}$ is again a feed-forward neural network that can be adjusted to approximate any continuous function. Thus, the Kolmogorov representation can be approximated through this particular construction.

This highlights a common issue in current approximation results for complex structures such as the transformer. Density-type results are rarely illuminating, since they can be constructed in many ways due to the structural flexibility. However, they rarely reveal the working principles of the complex model under study. In particular, it gives little insights to why and when these models should be used for applications. Such insights may result from finer analysis of approximation properties, including Jackson-type and Bernstein-type results as outlined for the other architectures previously.

To date, there are few - if any - Jackson or Berstein-type results for sequence modelling using the Transformer. We mention a related series of works on static function approximation with a variant of the Transformer architecture [1, 48, 49]. Here, the targets are continuous functions $H : [0,1]^\tau \to K$, and $K \subset \mathbb{R}^n$ is a compact set. Examples include classification problems where K is a probability simplex, and covariance matrix prediction problems with K being the set of symmetric positive semi-definite matrices. The authors consider a variant of the transformer architecture to approximate this target function

family. For $x \in [0,1]^\tau$, and $Y_1, \cdots, Y_N \in K$, an approximant of the following form is considered:

$$\widehat{H}(x) = \mathrm{Attn}\big(\widehat{D}(\widehat{E}(x)), Y\big) = \sum_{i=1}^{N} \mathrm{softmax}\big(\widehat{D}(\widehat{E}(x))\big)_i \delta_{Y_i}, \tag{4.31}$$

where $\widehat{E} : \mathbb{R}^\tau \to \mathbb{R}^m$, $\widehat{D} : \mathbb{R}^m \to \mathbb{R}^N$ are two fully-connected neural networks, and δ_{Y_i} is a point mass at Y_i. The softmax is taken along the $i = 1, \ldots, N$ direction. One may observe the deviations of this architecture compared with the Transformer used in sequence modelling (4.25). For this modified architecture, the authors derive an approximation error estimate based on increasing the complexities of the encoder $\widehat{E}$ and the decoder $\widehat{D}$. The identified notion of regularity for the target to induce efficient approximation is smoothness, similar to classical approximation of functions. However, the sequence approximation properties of the practical Transformer architecture (4.25), and in particular its relation to memory structures in the data, remains an open problem. This is an important direction of future research.

5 Discussion and outlook

Let us first summarize in Table 5.1 the approximation results we discussed in Sections 3 to 4. Observe that most results in the literature are of the density-type, and current Jackson and Bernstein-type results are often limited to the simplified setting of linear activations. Nevertheless, these rate estimates are instructive in revealing some key insights on the approximation of sequence relationships using different architectures.

Table 5.1: Summary of approximation results for sequence modelling.

	Density-type	**Jackson-type**	**Bernstein-type**
RNN	✓	Barron, linear	linear
CNN	✓	linear	-
REncDec	✓	linear	-
Transformer	✓	-	-

We can collectively summarize this insight as a form of structural compatibility. That is to say:

> *Each model architecture is efficient in approximating precisely those targets that resemble its temporal structure.*

For example, we saw that RNNs are particularly good at approximating relationships with an exponentially decaying memory pattern. We can attribute this to the fact that RNNs themselves have an exponentially decaying memory structure, as evidenced by the expression Eq. (3.9). Similarly, temporal CNNs are effective in approximating targets whose memory structure has low-rank under tensorisation, i.e. can be written as the product-sum of few tensors. This is indeed what the temporal CNN itself looks like: we recall that the tensorisation rank of the temporal CNN with K layers and M channels scales like $KM^{1/K}$.

The same holds for the RNN encoder-decoder with respect to its low-rank structure under temporal products, induced by the context vector.

We emphasize that this notion of rank is very different from that in temporal CNN approximation. In the convolution case, the rank refers to the tensorisation procedure related to the stacked convolutional structure of the temporal CNN. In the case of recurrent encoder-decoders, the rank refers to the amount of temporal coupling between the input sequence and the output sequence. Mathematically, this coupling is measured by the rate of decay of the singular values in the expansion (4.23). The recurrent encoder-decoder with small context vector precisely parameterize a temporal relationship that has little coupling between inputs and outputs. The Jackson-type results show that the recurrent encoder-decoder is indeed adapted to approximate target relationships having the same property.

The requirement of structural compatibility is consistent with classical approximation theory. For example, trigonometric polynomials with low-orders are themselves smooth functions with small gradient-norms, and thus are adapted to approximating these functions. The same pattern is observed for non-linear approximation, wavelets and multi-resolution analysis, where weakened smoothness, sparsity and multi-scale structures dictate both the model structures and effective targets for their application [22].

Now, let us discuss future research directions to further our understanding of the approximation theory of sequence modelling. Besides the obvious task of completing Table 5.1, we may wish to ask: What does a successful theory of sequence approximation entail? While there is no singular definition of success, it is reasonable to discuss desired outcomes in two broad categories.

On the practical side, one pressing need is to reduce the amount of trial and error during model selection. The understanding of the suitability of different model architectures for different problem types is essential in guiding implementations in practice. Therefore, an important task is to formalize a model selection workflow for sequence modelling. This certainly requires more than approximation theory, but the current understanding already suggests that we should quantify the memory patterns observed in datasets to select the model archetype. Developing this concrete pipeline based on well-understood theory is of great interest and importance. Another practical application worth noting is the simplification of model architectures. Modern architectures developed for specific applications may be very complex, and it is likely that some of their components are not performance critical. A theory of sequence modelling should help to identify the components that may not be necessary, so as to simplify and distil the essential modelling techniques.

On the mathematical side, following the development of classical approximation theory [22], it is of interest to characterize the so-called approximation spaces that are associated with each sequence modelling hypothesis space. Recall that the results for RNN in Section 3 suggests a type of approximation space in the form of

$$\mathcal{C}_{\mathrm{RNN}} = \big\{ \boldsymbol{H} \in \mathcal{C} : \|\boldsymbol{H}\|_{\mathrm{RNN}} < \infty \big\}, \tag{5.1}$$

where the norm $\|\cdot\|_{\mathrm{RNN}}$ may take the form

$$\|\boldsymbol{H}\|_{\mathrm{RNN}} = \|\boldsymbol{H}\| + |\boldsymbol{H}|_{*}. \tag{5.2}$$

Here, $\|\boldsymbol{H}\|$ is the usual uniform norm $\|\boldsymbol{H}\| = \sup_t \sup_{\|\boldsymbol{x}\|_{L^\infty} \leq 1} |H_t(\boldsymbol{x})|$, and $|\boldsymbol{H}|_*$ is a suitable semi-norm measuring exponential decay. For example, motivated by Eq. (3.13), we may take

$$|\boldsymbol{H}|_* = \sup_{t \in \mathbb{R}} \max_{r=1,\ldots,\alpha+1} \sup_{\boldsymbol{x} \in \mathcal{X}_0} \left| e^{\beta t} H_t^{(r)}(\boldsymbol{x}) \right|, \tag{5.3}$$

where β is the supremum value for which $|e^{\beta t} H_t^{(r)}(\boldsymbol{x})|$ remains finite for all t, $\boldsymbol{x} \in \mathcal{X}_0$, and $r = 1, \ldots, \alpha + 1$. The set $\mathcal{X}_0 \subset \mathcal{X}$ is a suitable set of test sequences. One can check that $(\mathcal{C}_{\text{RNN}}, \|\cdot\|_{\text{RNN}})$ forms a normed linear subspace. Then, the Jackson-type estimate can be rewritten as the familiar form

$$\inf_{\widehat{\boldsymbol{H}} \in \mathcal{H}^m_{\text{L-RNN}}} \|\boldsymbol{H} - \widehat{\boldsymbol{H}}\| \leq \text{Constant } \times \frac{\|\boldsymbol{H}\|_{\text{RNN}}}{m^\alpha}. \tag{5.4}$$

Hence, this suggests that the approximation space $\mathcal{C}_{\text{RNN}}$ is the RNN analogue of the usual Sobolev spaces ($\mathcal{W}^{\alpha,\infty}$) that characterizes trigonometric polynomial approximation. The space $\mathcal{C}_{\text{RNN}}$ is reminiscent of the Schwartz spaces [74] arising in Fourier analysis, except that we are now concerned with exponentially (instead of polynomially) decaying derivatives. Note that here, we are primarily concerned with the effect of temporal structure on approximation. Since time is one-dimensional, regularity related to smoothness should be expected. In the case where the ambient dimension of the input sequence d is large (and the relationship is non-linear), one expects that smoothness alone is insufficient to ensure efficient approximation. In this case, one may envision approximation spaces with a combination of smoothness conditions in the temporal direction and Barron-type conditions in the spatial direction. Similar constructions of approximation spaces can be made from Jackson-type results for the other architectures we described before. To characterize these spaces, their interpolation theory and whether they correspond to familiar spaces arising from analysis is of keen mathematical interest. Another aspect is characterizing the difference between linear and non-linear approximation. Taking the RNN as an example, the usual RNN (with trainable W, U, b) is a non-linear hypothesis space, in the sense that the linear combination of two functional sequences from $\mathcal{H}^m_{\text{RNN}}$ is in general a new functional sequence not in $\mathcal{H}^m_{\text{RNN}}$, but $\mathcal{H}^{2m}_{\text{RNN}}$. On the contrary, reservoir computing systems take W, U, b as fixed random realizations, and $\mathcal{H}^m_{\text{ESN}} \oplus \mathcal{H}^m_{\text{ESN}} = \mathcal{H}^m_{\text{ESN}}$. That is to say, the $\mathcal{H}^m_{\text{ESN}}$ is a linear approximation space. In classical approximation theory, linear and non-linear (adaptive) approximation lead to different approximation spaces [22]. It is thus of interest to investigate this distinction for sequence modelling, e.g., clarifying the difference of using ESNs versus RNNs for approximation.

Beyond approximation theory, it is important to note that a comprehensive understanding of sequence modelling should also account for optimization and generalization aspects. Indeed, principled sequence modelling in machine learning is not only the design of model architectures, but also how to train them and how to regularize them to maximize testing performance. For example, it is observed that while RNN training can be shown to be stable in the certain regimes [3, 38], it can sometimes be provably ineffective in the presence of long-term memory [55, 56]. Generalization theories have also been explored, e.g. in [14, 79, 83].

Other than deterministic model families, there are also results on modeling sequential data via (latent) neural controlled (stochastic) differential equations, such as hybrid architectures with GANs [44], universal neural operators for causality [30], and neural SPDE models motivated by mild solutions [40, 71]. Applications include time series generation [60], irregular and long time series analysis [45, 64], and online prediction [63]. These interesting aspects of sequence modelling theory are beyond the scope of the current survey.

Acknowledgments

We thank the reviewers for their comments, and Lukas Gonon, Lyudmila Grigoryeva and Juan-Pablo Ortega for their useful suggestions on the manuscript.

QL is supported by the National Research Foundation, Singapore, under the NRF fellowship (project No. NRF-NRFF13-2021-0005). HJ is supported by National University of Singapore under the PGF scholarship. SW is supported by National University of Singapore under the Research scholarship.

References

[1] B. Acciaio, A. Kratsios, and G. Pammer, Designing universal causal deep learning models: The geometric (hyper) transformer, *arXiv:2201.13094*, 2022.

[2] N. I. Achieser, Theory of Approximation, Courier Corporation, 2013.

[3] Z. Allen-Zhu, Y. Li, and Z. Song, On the convergence rate of training recurrent neural networks, In: Advances in Neural Information Processing Systems, Vol. 32, *Curran Associates Inc.*, 6673–6685, 2019.

[4] D. Bahdanau, K. Cho, and Y. Bengio, Neural machine translation by jointly learning to align and translate, In: International Conference on Learning Representations (ICLR) 2015, May 2016.

[5] S. Bai, J. Z. Kolter, and V. Koltun, An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, *arXiv:1803.01271*, 2018.

[6] A. R. Barron, Neural net approximation, In: Proceedings of the 7th Yale Workshop on Adaptive and Learning Systems, Vol. 1, 69–72, 1992.

[7] A. R. Barron, Universal approximation bounds for superpositions of a sigmoidal function, *IEEE Trans. Inform. Theory*, **39**(3):930–945, 1993.

[8] Y. Bengio, P. Simard, and P. Frasconi, Learning long-term dependencies with gradient descent is difficult, *IEEE trans. neural netw.*, **5**(2):157–166, 1994.

[9] F. E. Benth, N. Detering, and L. Galimberti, Neural networks in Fréchet spaces, *Ann. Math. Artif. Intell.*, **91**(1):75–103, 2023.

[10] S. Boyd and L. Chua, Fading memory and the problem of approximating nonlinear operators with Volterra series, *IEEE Trans. Circuits Syst.*, **32**(11):1150–1161, 1985.

[11] S. Boyd, L. O. Chua, and C. A. Desoer, Analytical foundations of Volterra series, *IMA J. Math. Control Inform.*, **1**(3):243–282, 1984.

[12] S. L. Brunton and J. N. Kutz, Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control, Cambridge University Press, 2019.

[13] A. Chatterjee, An introduction to the proper orthogonal decomposition, *Curr. Sci.*, **78**(7):808–817, 2000.

[14] M. Chen, X. Li, and T. Zhao, On generalization bounds of a family of recurrent neural networks, In: International Conference on Artificial Intelligence and Statistics, September 2018.

[15] T. Chen and H. Chen, Approximation capability to functions of several variables, nonlinear functionals, and operators by radial basis function neural networks, *IEEE trans. neural netw.*, **6**(4):904–910, 1995.

[16] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, In: Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, 103–111, 2014.
[17] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, Learning phrase representations using RNN encoder-decoder for statistical machine translation, In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 1724–1734, 2014.
[18] T. W. S. Chow and X.-D. Li, Modeling of continuous time dynamical systems with input by recurrent neural networks, *IEEE Trans. Circuits Syst. I: Fundamental Theory and Applications*, **47**(4):575–578, 2000.
[19] J.-B. Cordonnier, A. Loukas, and M. Jaggi, On the relationship between self-attention and convolutional layers, In: International Conference on Learning Representations, March 2020.
[20] C. Cuchiero, F. Primavera, and S. Svaluto-Ferro, Universal approximation theorems for continuous functions of càdlàg paths and Lévy-type signature models, *arXiv:2208.02293*, 2022.
[21] G. Cybenko, Approximation by superpositions of a sigmoidal function, *Math. Control Signals Systems*, **2**(4):303–314, 1989.
[22] R. A. DeVore, Nonlinear approximation, *Acta Numer.*, **7**:51–150, 1998.
[23] Y. Dong, J.-B. Cordonnier, and A. Loukas, Attention is not all you need: Pure attention loses rank doubly exponentially with depth, In: Proceedings of the 38th International Conference on Machine Learning, PMLR, 2793–2803, 2021.
[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, and S. Gelly, An image is worth 16x16 words: Transformers for image recognition at scale, *arXiv:2010.11929*, 2020.
[25] W. E, Towards a mathematical understanding of neural network-based machine learning : What we know and what we don't, *CSIAM trans. appl. math.*, **1**(4):561–615, 2020.
[26] W. E, C. Ma, and L. Wu, A priori estimates of the population risk for two-layer neural networks, *Commun. Math. Sci.*, **17**(5):1407–1425, 2019.
[27] W. E and S. Wojtowytsch, On the Banach spaces associated with multi-layer ReLU networks : Function representation, approximation theory and gradient descent dynamics, *CSIAM trans. appl. math.*, **1**(3):387–440, 2020.
[28] T. Erdélyi and P. Borwein, A sharp Bernstein-type inequality for exponential sums, *Journal für die reine und angewandte Mathematik*, **1996**(476):127–142, 1996.
[29] K.-I. Funahashi and Y. Nakamura, Approximation of dynamical systems by continuous time recurrent neural networks, *Neural Netw.*, **6**(6):801–806, 1993.
[30] L. Galimberti, G. Livieri, and A. Kratsios, Designing universal causal deep learning models: The case of infinite-dimensional dynamical systems from stochastic analysis, *arXiv:2210.13300*, 2022.
[31] L. Gonon, L. Grigoryeva, and J.-P. Ortega, Approximation bounds for random neural networks and reservoir systems, *Ann. Appl. Probab.*, **33**(1):28–69, 2023.
[32] L. Gonon and J.-P. Ortega, Reservoir computing universality with stochastic inputs, *IEEE Trans. Neural Netw. Learn. Syst.*, **31**(1):100–112, 2020.
[33] L. Gonon and J.-P. Ortega, Fading memory echo state networks are universal, *Neural Netw.*, **138**:10–13, 2021.
[34] L. Grigoryeva and J.-P. Ortega, Universal discrete-time reservoir computers with stochastic inputs and linear readouts using non-homogeneous state-affine systems, *J. Mach. Learn. Res.*, **19**:1–40, 2018.
[35] L. Grigoryeva and J.-P. Ortega, Echo state networks are universal, *Neural Netw.*, **108**:495–508, 2018.
[36] L. Grigoryeva and J.-P. Ortega, Differentiable reservoir computing, *J. Mach. Learn. Res.*, **20**(179):1–62, 2019.
[37] 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, 120, June 2020.
[38] M. Hardt, T. Ma, and B. Recht, Gradient descent learns linear dynamical systems, *J. Mach. Learn. Res.*, **19**(29):1–44, 2018.
[39] S. Hochreiter and J. Schmidhuber, Long short-term memory, *Neural Comput.*, **9**(8):1735–1780, 1997.
[40] P. Hu, Q. Meng, B. Chen, S. Gong, Y. Wang, W. Chen, R. Zhu, Z.-M. Ma, and T.-Y. Liu, Neural operator with regularity structure for modeling dynamics driven by SPDEs, *arXiv:2204.06255*, 2022.

[41] G. Hwang and M. Kang, Minimal width for universal property of Deep RNN, *arXiv:2211.13866*, 2022.
[42] 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, 4961–4970, 2021.
[43] N. Kalchbrenner and P. Blunsom, Recurrent continuous translation models, In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, 1700–1709, October 2013, Association for Computational Linguistics.
[44] P. Kidger, J. Foster, X. Li, and T. J. Lyons, Neural SDEs as infinite-dimensional gans, In: International Conference on Machine Learning, PMLR, 5453–5463, 2021.
[45] P. Kidger, J. Morrill, J. Foster, and T. Lyons, Neural controlled differential equations for irregular time series, *Adv. Neural Inf. Process. Syst.*, **33**:6696–6707, 2020.
[46] T. G. Kolda and B. W. Bader, Tensor decompositions and applications, *SIAM Rev.*, **51**(3):455–500, 2009.
[47] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, Neural operator: Learning maps between function spaces, *arXiv:2108.08481*, 2021.
[48] A. Kratsios, V. Debarnot, and I. Dokmanić, Small transformers compute universal metric embeddings, *arXiv:2209.06788*, 2022.
[49] A. Kratsios, B. Zamanlooy, T. Liu, and I. Dokmanić, Universal approximation under constraints is possible with transformers, *arXiv:2110.03303*, 2022.
[50] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Communications of the ACM*, **60**(6):84–90, 2017.
[51] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, Temporal convolutional networks for action segmentation and detection, In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 156–165, 2017.
[52] Q. Li, T. Lin, and Z. Shen, Deep neural network approximation of invariant functions through dynamical systems, *arXiv:2208.08707*, 2022.
[53] S. Li, X. Chen, D. He, C.-J. Hsieh, Can vision transformers perform convolution? *arXiv:2111.01353*, 2021.
[54] X.-D. Li, J. K. L. Ho, and T. W. S. Chow, Approximation of dynamical time-variant systems by continuous-time recurrent neural networks, *IEEE Trans. Circuits Syst. II: Express Briefs*, **52**(10):656–660, 2005.
[55] Z. Li, J. Han, W. E, and Q. Li, On the curse of memory in recurrent neural networks: Approximation and optimization analysis, In: International Conference on Learning Representations, March 2021.
[56] Z. Li, J. Han, W. E, and Q. Li, Approximation and optimization theory for linear continuous-time recurrent neural networks, *J. Mach. Learn. Res.*, **23**(42):1–85, 2022.
[57] Z. Li, H. Jiang, and Q. Li, On the approximation properties of recurrent encoder-decoder architectures, In: International Conference on Learning Representations, February 2022.
[58] T. Lin, Z. Shen, and Q. Li, On the universal approximation property of deep fully convolutional neural networks, *arXiv:2211.14047*, 2022.
[59] G. G. Lorentz, *Approximation of Functions*, AMS, 2005.
[60] P. D. Lozano, T. L. Bagén, and J. Vives, Neural SDEs for conditional time series generation and the Signature-Wasserstein-1 metric, *arXiv:2301.01315*, 2023.
[61] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators, *Nat. Mach. Intell.*, **3**(3):218–229, 2021.
[62] S. G. Mallat, A Wavelet Tour of Signal Processing: The Sparse Way, Elsevier/Academic Press, 2009.
[63] J. Morrill, P. Kidger, L. Yang, and T. Lyons, On the choice of interpolation scheme for neural CDEs, *Transactions on Machine Learning Research*, 2022(9), 2022.
[64] J. Morrill, C. Salvi, P. Kidger, and J. Foster, Neural rough differential equations for long time series, In: International Conference on Machine Learning, PMLR, 7829–7838, 2021.
[65] A. Neufeld and P. Schmocker, Chaotic hedging with iterated integrals and neural networks, *arXiv: 2209.10166*, 2022.
[66] S. Okumoto and T. Suzuki, Learnability of convolutional neural networks for infinite dimensional input via mixed and anisotropic smoothness, In: International Conference on Learning Representations, March 2022.
[67] 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,

4922–4931, 2019.
[68] P. A. Ostrand, Dimension of metric spaces and Hilbert's problem 13, *Bull. New Ser. Am. Math. Soc.*, **71**(4): 619–622, 1965.
[69] P. Petersen and F. Voigtlaender, Equivalence of approximation by convolutional neural networks and fully-connected networks, *Proc. Amer. Math. Soc.*, **148**(4):1567–1581, 2020.
[70] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, *Nature*, **323**(6088):533–536, 1986.
[71] C. Salvi, M. Lemercier, and A. Gerasimovics, Neural stochastic PDEs: Resolution-invariant learning of continuous spatiotemporal dynamics, *arXiv:2110.10249*, 2021.
[72] 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.), Lecture Notes in Computer Science, 632–640, Springer, 2006.
[73] E. Sontag, Neural nets as systems models and controllers, In: Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems, 1992.
[74] E. M. Stein and R. Shakarchi, Fourier Analysis: An Introduction, Vol. 1, Princeton University Press, 2011.
[75] M. B. Stinchcombe, Neural network approximation of continuous functionals and continuous functions on compactifications, *Neural Netw.*, **12**(3):467–477, 1999.
[76] I. Sutskever, O. Vinyals, and Q. V. Le, Sequence to sequence learning with neural networks, In: Advances in Neural Information Processing Systems, **27**, 2014.
[77] D. Tang, B. Qin, and T. Liu, Document modeling with gated recurrent neural network for sentiment classification, In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 1422–1432, September 2015, Association for Computational Linguistics.
[78] S. J. Taylor, Modelling Financial Time Series, World Scientific, 2008.
[79] Z. Tu, F. He, and D. Tao, Understanding generalization in recurrent neural networks, In: International Conference on Learning Representations, March 2020.
[80] D. Yarotsky, Universal approximations of invariant maps by neural networks, *Constr. Approx.*, **55**(1):407–474, 2022.
[81] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar, Are transformers universal approximators of sequence-to-sequence functions? International Conference on Learning Representations, 2020.
[82] C. Yun, Y.-W. Chang, S. Bhojanapalli, A. S. Rawat, S. Reddi, and S. Kumar, O (n) connections are expressive enough: Universal approximability of sparse transformers, *Adv. Neural Inf. Process. Syst.*, **33**:13783–13794, 2020.
[83] L. Wang, B. Shen, B. Hu, and X. Cao, On the provable generalization of recurrent neural networks, In: Advances in Neural Information Processing Systems, Vol. 34, *Curran Associates Inc.*, 20258–20269, 2021.
[84] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, Wavenet: A generative model for raw audio, Proceedings of the 9th ISCA Workshop on Speech Synthesis Workshop (SSW 9), p. 125, 2016.
[85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, In: Advances in Neural Information Processing Systems, Vol. 30., *Curran Associates Inc.*, 2017.
[86] V. Volterra, Theory of Functionals and of Integral and Integro-differential Equations, Blackie & Son Limited, 1930.
[87] Z. Xing, J. Pei, and E. Keogh, A brief survey on sequence classification, *ACM SIGKDD Explorations Newsletter*, **12**(1):40–48, 2010.
[88] D.-X. Zhou, Universality of deep convolutional neural networks, *Appl. Comput. Harmon. Anal.*, **48**(2):787–794, 2020.