SEQ2TENS: AN EFFICIENT REPRESENTATION OF SEQUENCES BY LOW-RANK TENSOR PROJECTIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential data such as time series, video, or text can be challenging to analyse as the ordered structure gives rise to complex dependencies. At the heart of this is non-commutativity, in the sense that reordering the elements of a sequence can completely change its meaning. We use a classical mathematical object – the free algebra – to capture this non-commutativity. To address the innate computational complexity of this algebra, we use compositions of low-rank tensor projections. This yields modular and scalable building blocks that give state-of-the-art performance on standard benchmarks such as multivariate time series classification and generative models for video.

1 INTRODUCTION

A central task of learning is to find representations of the underlying data that efficiently and faithfully capture their structure. In the case of sequential data, one data point consists of a sequence of objects. This is a rich and non-homogeneous class of data and includes classical uni- or multi-variate time series (sequences of scalars or vectors), video (sequences of images), and text (sequences of letters). Particular challenges of sequential data are that each sequence entry can itself be a highly structured object and that data sets typically include sequences of different length which makes naive vectorization troublesome.

Contribution. Our main result is a generic method that takes a *static feature map* for a class of objects (e.g. a feature map for vectors, images, or letters) as input and turns this into a feature map for sequences of arbitrary length of such objects (e.g. a feature map for time series, video, or text). We call this feature map for sequences Seq2Tens for reasons that will become clear; among its attractive properties are that it (i) provides a structured, parsimonious description of sequences; generalizing classical methods for strings, (ii) comes with theoretical guarantees such as universality, (iii) can be turned into modular and flexible neural network (NN) layers for sequence data. The key ingredient to our approach is to embed the feature space of the static feature map into a larger linear space that forms an algebra (a vector space equipped with a multiplication). The product in this algebra is then used to "stitch together" the static features of the individual sequence entries in a structured way. The construction that allows to do all this is classical in mathematics, and known as the *free algebra* (over the static feature space).

Outline. Section 2 formalizes the main ideas of Seq2Tens and introduces the free algebra $\mathrm{T}(V)$ over a space V as well as the associated product, the so-called *convolution tensor product*. It shows how low rank (LR) constructions combined with sequence-to-sequence transforms allows one to efficiently use this rich algebraic structure. Section 3 applies the results of Section 2 to build modular and scalable NN layers. Section 4 demonstrates the flexibility and modularity of this approach on both discriminative and generative benchmarks. Section 5 makes connections with previous work and summarizes this article. In the appendices we provide mathematical background, extensions, and detailed proofs for our theoretical results.

2 CAPTURING ORDER BY NON-COMMUTATIVE MULTIPLICATION

We denote the set of sequences of elements in a set $\mathcal{X}$ by

$$\mathrm{Seq}(\mathcal{X}) = \{\mathbf{x} = (\mathbf{x}_i)_{i=1,\ldots,L} : \mathbf{x}_i \in \mathcal{X},\, L \geq 1\} \tag{1}$$

where $L \geq 1$ is some arbitrary length. Even if $\mathcal{X}$ itself is a linear space, e.g. $\mathcal{X} = \mathbb{R}$, $\mathrm{Seq}(\mathcal{X})$ is never a linear space since there is no natural addition of two sequences of different length.

Seq2Tens in a nutshell. Given any vector space V we may construct the so-called free algebra $\mathrm{T}(V)$ over V. We describe the space $\mathrm{T}(V)$ in detail below, but as for now the only thing that is important is that $\mathrm{T}(V)$ is also a vector space that includes V, and that it carries a non-commutative product, which is, in a precise sense, "the most general product" on V.

The main idea of Seq2Tens is that any "static feature map" for elements in $\mathcal{X}$

$$\phi : \mathcal{X} \to V$$

can be used to construct a new feature map $\Phi : \mathrm{Seq}(\mathcal{X}) \to \mathrm{T}(V)$ for sequences in $\mathcal{X}$ by using the algebraic structure of $\mathrm{T}(V)$: the non-commutative product on $\mathrm{T}(V)$ makes it possible to "stitch together" the individual features $\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_L) \in V \subset \mathrm{T}(V)$ of the sequence $\mathbf{x}$ in the larger space $\mathrm{T}(V)$ by multiplication in $\mathrm{T}(V)$. With this we may define the feature map $\Phi(\mathbf{x})$ for a sequences $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_L) \in \mathrm{Seq}(\mathcal{X})$ as follows

(i) lift the map $\phi : \mathcal{X} \to V$ to a map $\varphi : \mathcal{X} \to \mathrm{T}(V)$,

(ii) map $\mathrm{Seq}(\mathcal{X}) \to \mathrm{Seq}(\mathrm{T}(V))$ by $(\mathbf{x}_1, \ldots, \mathbf{x}_L) \mapsto (\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_L))$,

(iii) map $\mathrm{Seq}(\mathrm{T}(V)) \to \mathrm{T}(V)$ by multiplication $(\varphi(\mathbf{x}_1), \ldots, \varphi(\mathbf{x}_L)) \mapsto \varphi(\mathbf{x}_1) \cdots \varphi(\mathbf{x}_L)$.

In a more concise form, we define Φ as

$$\Phi : \mathrm{Seq}(\mathcal{X}) \to \mathrm{T}(V), \quad \Phi(\mathbf{x}) = \prod_{i=1}^{L} \varphi(\mathbf{x}_i) \tag{2}$$

where $\prod$ denotes multiplication in $\mathrm{T}(V)$. We refer to the resulting map Φ as the Seq2Tens map. Why is this construction a good idea? First note, that step (i) is always possible since $V \subset \mathrm{T}(V)$ and we discuss the simplest such lift before Theorem 2.1 as well as other choices in Appendix B. Further, if ϕ, respectively φ, provides a faithful representation of objects in $\mathcal{X}$, then there is no loss of information in step (ii) and since step (iii) uses "the most general product" to multiply $\varphi(\mathbf{x}_1) \cdots \varphi(\mathbf{x}_L)$ one expects that $\Phi(\mathbf{x}) \in \mathrm{T}(V)$ faithfully represents the sequence $\mathbf{x}$ as an element of $\mathrm{T}(V)$.

Indeed in Theorem 2.1 below we show an even stronger statement, namely that if the static feature map $\phi : \mathcal{X} \to V$ contains enough non-linearities so that non-linear functions from $\mathcal{X}$ to $\mathbb{R}$ can be approximated as *linear functions* of the static feature map ϕ, then the above construction extends this property to functions of sequences. Put differently, *if ϕ is a universal feature map for $\mathcal{X}$, then Φ is a universal feature map for* $\mathrm{Seq}(\mathcal{X})$; that is, any non-linear function $f(\mathbf{x})$ of a sequence $\mathbf{x}$ can be approximated as a linear functional of $\Phi(\mathbf{x})$, $f(\mathbf{x}) \approx \langle \ell, \Phi(\mathbf{x}) \rangle$. We also emphasize that the domain of Φ is the space $\mathrm{Seq}(\mathcal{X})$ of sequences of *arbitrary* (finite) length.

The free algebra $\mathrm{T}(V)$ over a vector space V. Let V be a vector space. We denote by $\mathrm{T}(V)$ the set of sequences of tensors indexed by their degree m,

$$\mathrm{T}(V) := \{\mathbf{t} = (\mathbf{t}_m)_{m \geq 0} \,|\, \mathbf{t} \in V^{\otimes m}\} \tag{3}$$

where by convention $V^{\otimes 0} = \mathbb{R}$. For example, if $V = \mathbb{R}^d$ and $\mathbf{t} = (\mathbf{t}_m)_{m \geq 0}$ is some element of $\mathrm{T}(\mathbb{R}^d)$, then its degree $m = 1$ component is a d-dimensional vector $\mathbf{t}_1$, its degree $m = 2$ component is a $d \times d$ matrix $\mathbf{t}_2$, and its degree $m = 3$ component is a degree 3 tensor $\mathbf{t}_3$. By defining addition and scalar multiplication as

$$\mathbf{s} + \mathbf{t} := (\mathbf{s}_m + \mathbf{t}_m)_{m \geq 0}, \quad c \cdot \mathbf{t} = (c\mathbf{t}_m)_{m \geq 0} \tag{4}$$

the set $\mathrm{T}(V)$ becomes a linear space. By identifying $v \in V$ as the element $(0, v, 0, \ldots, 0) \in \mathrm{T}(V)$ we see that V is a linear subspace of $\mathrm{T}(V)$. Moreover, while V is only a linear space, $\mathrm{T}(V)$ carries

a product that turns $\mathrm{T}(V)$ into an algebra. This product is the so-called *tensor convolution product*, and is defined for $\mathbf{s}, \mathbf{t} \in \mathrm{T}(V)$ as

$$\mathbf{s} \cdot \mathbf{t} := \big(\sum_{i=0}^{m} \mathbf{s}_i \otimes \mathbf{t}_{m-i}\big)_{m \geq 0} = \big(1, \mathbf{s}_1 + \mathbf{t}_1, \mathbf{s}_2 + \mathbf{s}_1 \otimes \mathbf{t}_1 + \mathbf{t}_2, \dots\big) \in \mathrm{T}(V) \tag{5}$$

where $\otimes$ denotes the usual outer tensor product; e.g. for vectors $u = (u_i), v = (v_i) \in \mathbb{R}^d$ the outer tensor product $u \otimes v$ is the $d \times d$ matrix $(u_i v_j)_{i,j=1,\dots,d}$. We emphasize that like the outer tensor product $\otimes$, the tensor convolution product $\cdot$ is non-commutative, i.e. $\mathbf{s} \cdot \mathbf{t} \neq \mathbf{t} \cdot \mathbf{s}$. In a mathematically precise sense, $\mathrm{T}(V)$ *is the most general algebra that contains V; it is a "free construction"*. Since $\mathrm{T}(V)$ is realized as series of tensors of increasing degree, it is also known as the *tensor algebra* in the literature. Appendix A contains background on tensors and further examples.

Lifting static feature maps. Step (i) in the construction of Φ requires turning a given feature map $\phi : \mathcal{X} \to V$ into a map $\varphi : \mathcal{X} \to \mathrm{T}(V)$. Throughout the rest of this article we use the lift

$$\varphi(\mathbf{x}) = (1, \phi(\mathbf{x}), 0, 0 \dots) \in \mathrm{T}(V). \tag{6}$$

We discuss other choices in Appendix B, but attractive properties of the lift 6 are that (a) the evaluation of Φ against low rank tensors becomes a simple recursive formula (Proposition 2.3, (b) it is a generalization of sequence subpattern matching as used in string kernels (Appendix B.3, (c) despite its simplicity it performs exceedingly well in practice (Section 4).

Universality. A function $\phi : \mathcal{X} \to V$ is said to be *universal* if all continuous functions on $\mathcal{X}$ can be approximated as linear functions on the image of ϕ. One of the most powerful features of neural nets is their universality (Hornik, 1991). A very attractive property of Φ is that it preserves universality: if $\phi : \mathcal{X} \to V$ is universal, then $\Phi : \mathrm{Seq}(X) \to \mathrm{T}(V)$ is universal. To make this precise, note that $V^{\otimes m}$ is a linear space and therefore any $\ell = (\ell_0, \ell_1, \dots, \ell_M, 0, 0, \dots) \in \mathrm{T}(V)$ consisting of M tensors $\ell_m \in V^{\otimes m}$, yields a linear functional on $\mathrm{T}(V)$; e.g. if $V = \mathbb{R}^d$ and we identify ℓ_m in coordinates as $\ell_m = (\ell_m^{i_1,\dots,i_m})_{i_1,\dots,i_m \in \{1,\dots,d\}}$ then

$$\langle \ell, \mathbf{t} \rangle := \sum_{m=0}^{M} \langle \ell_m, \mathbf{t}_m \rangle = \sum_{m=0}^{M} \sum_{i_1,\dots,i_m \in \{1,\dots,d\}} \ell_m^{i_1,\dots,i_m} \mathbf{t}_m^{i_1,\dots,i_m}. \tag{7}$$

Thus linear functionals of the feature map Φ, are real-valued functions of sequences. Theorem 2.1 below shows that any continuous function $f : \mathrm{Seq}(\mathcal{X}) \to \mathbb{R}$ can by arbitrary well approximated by a $\ell \in \mathrm{T}(V)$, $f(\mathbf{x}) \approx \langle \ell, \Phi(\mathbf{x}) \rangle$.

Theorem 2.1. *Let $\phi : \mathcal{X} \to V$ be a universal map with a lift that satisfies some mild constraints, then the following map is universal:*

$$\Phi : \mathrm{Seq}(\mathcal{X}) \to \mathrm{T}(V), \quad \mathbf{x} \mapsto \Phi(\mathbf{x}). \tag{8}$$

A detailed proof and the precise statement of Theorem 2.1 is given in Appendix B.

The combinatorial explosion of tensor coordinates and what to do about it. The universality of Φ suggests the following approach to represent a function $f : \mathrm{Seq}(\mathcal{X}) \to \mathbb{R}$ of sequences: first compute $\Phi(\mathbf{x})$ and then optimize over ℓ (and possibly also the hyperparameters of φ) such that $f(\mathbf{x}) \approx \langle \ell, \Phi(\mathbf{x}) \rangle = \sum_{m=0}^{M} \langle \ell_m, \Phi_m(\mathbf{x}) \rangle$. Unfortunately, tensors suffer from a combinatorial explosion in complexity in the sense that even just storing $\Phi_m(\mathbf{x}) \in V^{\otimes m} \subset \mathrm{T}(V)$ requires $O(\dim(V)^m)$ real numbers. Below we resolve this computational bottleneck as follows: in Proposition 2.3 we show that for a special class of elements $\ell \in \mathrm{T}(V)$, the functional $\mathbf{x} \mapsto \langle \ell, \Phi(\mathbf{x}) \rangle$ can be efficiently computed in both time and memory. This is somewhat analogous to a kernel trick since it shows that $\langle \ell, \Phi(\mathbf{x}) \rangle$ can be cheaply computed without explicitly computing the feature map $\Phi(\mathbf{x})$. However, Theorem 2.1 guarantees universality under no restriction on ℓ, thus restriction to rank-1 functionals limits the class of functions $f(\mathbf{x})$ that can be approximated. Nevertheless, by iterating these "low-rank functional" constructions in the form of Sequence-to-Sequence transformations this can be ameliorated. We give the details below but to gain intuition, we invite the reader to think of this iteration analogous to stacking layers in neural network: each layer is a relatively simple non-linearity (e.g. a sigmoid composed with an affine function) but by composing such layers, complicated functions can be efficiently approximated.

Rank-1 functionals are computationally cheap. Degree $m = 2$ tensors are matrices and LR approximations of matrices are widely used in practice (Udell & Townsend, 2019) to address the quadratic complexity. The definition below generalizes the rank of matrices (tensors of degree $m = 2$) to tensors of any degree m.

Definition 2.2. The *rank* (also called *CP rank* (Carroll & Chang, 1970)) of a degree m tensor $\mathbf{t} \in V^{\otimes m}$ is the smallest number $r \geq 0$ such that one may write

$$\mathbf{t} = \sum_{i=0}^{r} \mathbf{v}_i^1 \otimes \cdots \otimes \mathbf{v}_i^m, \quad \mathbf{v}_i^1, \ldots, \mathbf{v}_i^m \in V. \tag{9}$$

We say that $\ell = (\ell_m)_{m \geq 0} \in \mathrm{T}(V)$ has rank 1 (and degree M) if each $\ell_m \in V^{\otimes m}$ is a rank-1 tensor and $\ell_i = 0$ for $i > M$.

A direct calculation shows that if ℓ is of rank 1, then $\langle \ell, \Phi(\mathbf{x}) \rangle$ can be computed very efficiently by inner product evaluations in V.

Proposition 2.3. *Let $\ell = (\ell_m)_{m \geq 0} \in \mathrm{T}(V)$ be of rank-1 and degree M. If ϕ is lifted to φ as in equation 6, then*

$$\langle \ell, \Phi(\mathbf{x}) \rangle = \sum_{m=0}^{M} \sum_{1 \leq i_1 < \cdots < i_m \leq L} \prod_{k=1}^{m} \langle \mathbf{v}_k^m, \phi(\mathbf{x}_{i_k}) \rangle \tag{10}$$

where $\ell_m = \mathbf{v}_1^m \otimes \cdots \otimes \mathbf{v}_m^m \in V^{\otimes m}$, $\mathbf{v}_i^m \in V$ and $m = 0, \ldots, M$.

Note that the inner sum is taken over all non-contiguous subsequences of $\mathbf{x}$ of length m, analogous to m-mers of strings and we make this connection precise in Appendix B.3; the proof of Proposition 2.3 is given in Appendix B.1.1. As pointed out above, the restriction to rank-1 functionals would limit the class of functions of sequences that can be approximated. To address this, we introduce Sequence-to-Sequence transforms that allow us to stack such rank-1 functionals.

Sequence-to-Sequence transforms. We can use the Seq2Tens map Φ to build Sequence-to-Sequence transformations: fix a feature map $\phi_{\theta_1} : \mathcal{X} \to V$ parametrized by θ_1 and use the resulting Seq2Tens map $\Phi_{\theta_1}(\mathbf{x}) = \prod_{i=1}^{L} \varphi_{\theta_1}(\mathbf{x}_i)$ to transform

$$\mathrm{Seq}(\mathcal{X}) \to \mathrm{Seq}(\mathbb{R}), \quad \mathbf{x} \mapsto \big(\langle \ell, \Phi(\mathbf{x}_1) \rangle, \ldots, \langle \ell, \Phi(\mathbf{x}_1, \ldots, \mathbf{x}_L) \rangle \big). \tag{11}$$

If $\ell \in \mathrm{T}(V)$ is of rank-1, this sequence transformation is computationally cheap by Proposition 2.3. For a collection $\mathcal{L}$ of n_1 rank-1 elements $\ell_1, \ldots, \ell_{n_1} \in \mathrm{T}(V)$ this can be done cheaply in parallel,

$$\mathrm{Seq}(\mathcal{X}) \to \mathrm{Seq}(\mathbb{R}^{n_1}),\ \mathbf{x} \mapsto (\mathcal{L}_1 \circ \Phi(\mathbf{x}_1), \ldots, \mathcal{L}_1 \circ \Phi(\mathbf{x}_1, \ldots, \mathbf{x}_L)) \tag{12}$$

where $\mathcal{L}_1 \circ \Phi_{\theta_1}(\mathbf{x}_1, \ldots, \mathbf{x}_i)$ denotes the vector $(\langle \ell_j, \Phi(\mathbf{x}_1, \ldots, \mathbf{x}_i) \rangle)_{j=1,\ldots,n_1} \in \mathbb{R}^{n_1}$. Recall now that without the restriction to rank-1 functionals, Theorem 2.1 would guarantee that any multivariate function $f : \mathrm{Seq}(\mathcal{X}) \to \mathbb{R}^{n_1}$ can be approximated by simply evaluating the resulting sequence in $\mathbb{R}^{n_1}$ at the endpoint, i.e. $f(\mathbf{x}) \approx \mathcal{L}_1 \circ \Phi(\mathbf{x}_1, \ldots, \mathbf{x}_L)$. The restriction to rank-1 functionals limits the class of functions but by stacking such Sequence-to-Sequence transformations we recover an expressive model that is computationally cheap since each transformation is cheap.

Stacking Sequence-to-Sequence transforms. Inspired by the empirical successes of stacked RNNs (Graves et al., 2013b;a; Sutskever et al., 2014), we iterate the transformation 12 D-times:

$$\mathrm{Seq}(\mathcal{X}) \to \mathrm{Seq}(\mathbb{R}^{n_1}) \to \mathrm{Seq}(\mathbb{R}^{n_2}) \to \cdots \to \mathrm{Seq}(\mathbb{R}^{n_D}). \tag{13}$$

Each of these mappings to $\mathrm{Seq}(\mathbb{R}^{n_i})$ is parametrized by the parameters θ_i of a static feature map ϕ_{θ_i} and a collection $\mathcal{L}_i$ consisting of n_i rank-1 elements of $\mathrm{T}(V)$; we denote these parameters as $\tilde{\theta}_i = (\theta_i, \mathcal{L}_i)$. Taking the last observation in the sequence in $\mathrm{Seq}(\mathbb{R}^{n_D})$ we are left with a map

$$\Phi_{\tilde{\theta}_1, \ldots, \tilde{\theta}_D} : \mathrm{Seq}(\mathcal{X}) \to \mathbb{R}^{n_D}. \tag{14}$$

Making precise how the stacking of such rank-1 sequence-to-sequence transformations approximates general functions requires more tools from algebra and we provide a rigorous quantitative statement in Appendix C. Here, we just appeal to the analogy made with stacking layers in a neural network mentioned earlier and empirically validate this in our experiments in Section 4.

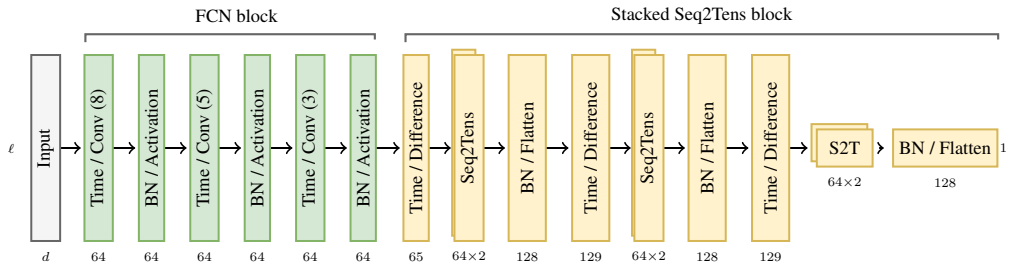

Figure 1: Depiction of the proposed models for TSC. S2T^3 only consists of a stacked Seq2Tens block (yellow), while FCN-S2T^3 also precedes it with an FCN block (green).

3 Building neural networks with Seq2Tens layers

The Seq2Tens map Φ built from a static feature map ϕ is universal if ϕ is universal, Theorem 2.1. NNs form a flexible class of universal feature maps with strong empirical success for data in $\mathcal{X} = \mathbb{R}^d$, and thus make a natural choice for ϕ. Combined with standard deep learning constructions, the framework of Section 2 can build modular and expressive layers for sequence learning.

(Stacked) Seq2Tens layers. We use as static feature map $\phi : \mathcal{X} = \mathbb{R}^d \to \mathbb{R}^n$ a simple multilayer perceptron (MLP), $\phi = \phi_{D'} \circ \cdots \circ \phi_1$ with $\phi_j(\mathbf{x}) = \sigma(\mathbf{W}_j\mathbf{x} + \mathbf{b}_j)$. We then lift this to a map $\varphi : \mathbb{R}^d \to \mathrm{T}(\mathbb{R}^d)$ as prescribed in equation 6. Hence, the resulting Seq2Tens layer $\mathbf{x} \mapsto (\mathcal{L} \circ \Phi_\theta(\mathbf{x}_1, \ldots, \mathbf{x}_i))_{i=1,\ldots,L}$ is a sequence transform from $\mathrm{Seq}(\mathbb{R}^d)$ to $\mathrm{Seq}(\mathbb{R}^n)$ that is parametrized by $\tilde{\theta} = (\mathcal{L}, \theta)$ which consists of the collection $\mathcal{L}$ of n rank-1 elements $\ell_1, \ldots, \ell_n \in \mathrm{T}(V)$, as well as θ containing the D' pairs of MLP weights $(\mathbf{W}_j, \mathbf{b}_j)_{j=1,\ldots,D'}$. By stacking D such sequence transforms and evaluating at the endpoint as in equation 13 we get a map $\Phi_{(\tilde{\theta}_1,\ldots,\tilde{\theta}_D)} : \mathrm{Seq}(\mathbb{R}^d) \to \mathbb{R}^n$ parametrized by a D such parameters $(\tilde{\theta}_j)_{j\in\{1,\ldots,D\}}$, all of which are differentiable and hence can be choosen by stochastic gradient descent. This can also be combined with standard preprocessing such as adding lags and convolutions (Appendix D.1).

Bidirectional Seq2Tens layers. The sequence transformation in equation 12 is completely causal. That is, each step of the output sequence depends only on past information. For generative models it helps to make the output depend on both the past and the future information, see Graves et al. (2013a); Baldi et al. (1999); Li & Mandt (2018). Similarly to bidirectional RNNs and LSTMs (Schuster & Paliwal, 1997; Graves & Schmidhuber, 2005), we may achieve this by defining a bidirectional Seq2Tens layer as

$$\Phi_{\mathrm{bi}}(\mathbf{x}) : \mathrm{Seq}(\mathbb{R}^d) \to \mathrm{Seq}(\mathbb{R}^{n+n'}), \quad \mathbf{x} \mapsto (\mathcal{L} \circ \Phi_{\theta_1}(\mathbf{x}_1, \ldots, \mathbf{x}_i), \mathcal{L} \circ \Phi_{\theta_2}(\mathbf{x}_i, \ldots, \mathbf{x}_L))_{i=1}^L. \quad (15)$$

The sequential nature is kept intact by making the distinction between what classifies as past information (the first n coordinates), and what classifies as future information (the last n' coordinates). This amounts to having a form of precognition in the model, and has been applied in e.g. dynamics generation (Li & Mandt, 2018), machine translation (Sundermeyer et al., 2014), and speech processing (Graves & Schmidhuber, 2005; Graves et al., 2013a).

4 Experiments

We demonstrate the modularity and flexibility of the above Seq2Tens layer and its variants by applying it to (i) multivariate time series classification, (ii) generative modelling of sequential data. In both cases, we take a strong baseline model (FCN and GP-VAE as detailed below) and add Seq2Tens layers on top. As Theorem 2.1 states, the Seq2Tens layer is universal on sequences if it is preceded by a universal state-space nonlinearity. This means that these layers should be at least preceded by some time-distributed dense layers, and hence, are expected to perform best as an add-on on top of other models. The additional computation time is negligible (in fact, for FCN it allows us to reduce the number of parameters significantly), but it can yield substantial improvements. This is remarkable, since the results of the original models are often already state-of-the-art on well-established and popular (frequentist and Bayesian) benchmarks.

Table 1: Posterior probabilities given by a Bayesian signed-rank test comparison of the proposed methods against the baselines. $\{>\}$, $\{<\}$, $\{=\}$ refer to the respective events that the row method is better, the column method is better, or that they are equivalent.

Model	$S2T^3$			$FCN\text{-}S2T^3$		
	$p(>)$	$p(=)$	$p(<)$	$p(>)$	$p(=)$	$p(<)$
$S2T^3$	–	–	–	0.000	0.017	**0.983**
SMTS (Baydogan & Runger, 2015a)	0.369	0.000	**0.631**	0.005	0.000	**0.995**
LPS (Baydogan & Runger, 2015b)	0.477	0.002	**0.520**	0.001	0.000	**0.999**
mvARF (Tuncel & Baydogan, 2018)	0.021	0.168	**0.811**	0.000	0.089	**0.911**
DTW (Sakoe & Chiba, 1978)	0.086	0.000	**0.914**	0.000	0.000	**1.000**
ARKernel (Cuturi & Doucet, 2011)	0.136	0.104	**0.760**	0.000	0.044	**0.955**
gRSF (Karlsson et al., 2016)	**0.851**	0.008	0.141	0.063	0.024	**0.913**
MLSTMFCN (Karim et al., 2019)	**0.947**	0.038	0.014	0.342	0.190	**0.469**
MUSE (Schäfer & Leser, 2017)	**0.606**	0.194	0.200	0.021	0.197	**0.781**
FCN (Wang et al., 2017)	**0.971**	0.000	0.029	0.176	0.033	**0.791**
ResNet (Wang et al., 2017)	**0.962**	0.000	0.038	0.122	0.000	**0.878**

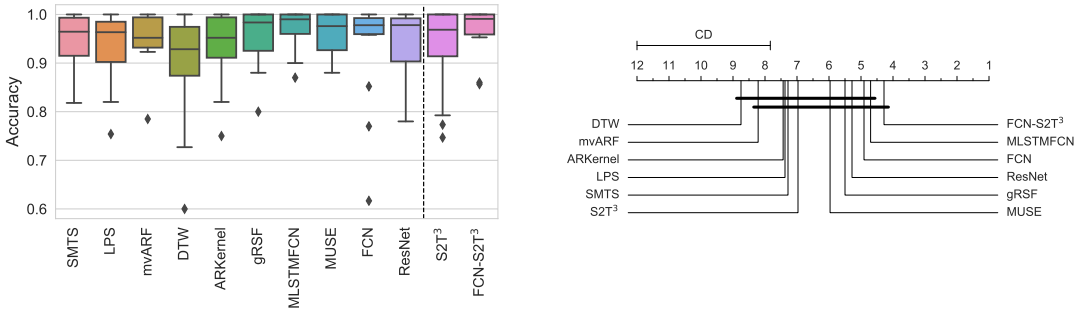

Figure 2: Box-plot of classification accuracies (left) and critical difference diagram (right).

4.1 Multivariate time series classification

First, we consider multivariate time series classification (TSC) on a semi-standardized collection of benchmark datasets unified in Baydogan (2015). A wide range of previous publications report results on this archive, which makes it possible to compare against a wide range of baseline methods without bias in parameter settings. A subset of this archive was also considered in a recent review paper on DL for TSC (Ismail Fawaz et al., 2019) from where we borrow the best performing models as baselines. We provide further details on the experiment, baselines and datasets in Appendix E.1.

As our own models, we introduce two simple architectures that utilize Seq2Tens layers: (i) $S2T^3$ stacks 3 Seq2Tens layers of order 2 and width 64; (ii) $FCN\text{-}S2T^3$ precedes the $S2T^3$ block by a CNN block of 3 convolutional layers of width 64 and filter sizes of $8, 5, 3$; a downsized version of the FCN model of (Wang et al., 2017) with a time-embedding preceding each convolutional layer (Liu et al., 2018a) and without a final global average pooling (GAP) layer, see Figure 1.

Benchmark results. We trained the introduced models on each dataset 5 times while the baseline results were borrowed from their respective publications[1]. Figure 2 depicts a comparison of the results as a box-plot of distributions and a critical difference diagram using the Nemenyi test (Nemenyi, 1963), while Table 6 in Appendix E.1 shows the full list of means and standard deviations. Since mean-ranks based tests raise some paradoxical issues (Benavoli et al., 2016), it is also customary to conduct pairwise comparisons using frequentist (Demšar, 2006) or Bayesian (Benavoli et al., 2017) hypothesis tests. We adopted[2] the Bayesian signed-rank test approach from Benavoli

[1]Except for FCN, ResNet, that we ran 5 times using the settings from Ismail Fawaz et al. (2019). For MLSTMFCN (Karim et al., 2019), the results are the same as reported in Schäfer & Leser (2017)

[2]Using the implementation available at `https://github.com/janezd/baycomp`.

Table 2: Performance comparison of GP-VAE (B-S2T) with the baseline methods

Method	HMNIST			Sprites	Physionet
	NLL	MSE	AUROC	MSE	AUROC
Mean imputation	-	0.168 ± 0.000	0.938 ± 0.000	0.013 ± 0.000	0.703 ± 0.000
Forward imputation	-	0.177 ± 0.000	0.935 ± 0.000	0.028 ± 0.000	0.710 ± 0.000
VAE	0.599 ± 0.002	0.232 ± 0.000	0.922 ± 0.000	0.028 ± 0.000	0.677 ± 0.002
HI-VAE	0.372 ± 0.008	0.134 ± 0.003	$\mathbf{0.962 \pm 0.001}$	0.007 ± 0.000	0.686 ± 0.010
GP-VAE	0.350 ± 0.007	0.114 ± 0.002	$\mathbf{0.960 \pm 0.002}$	$\mathbf{0.002 \pm 0.000}$	0.730 ± 0.006
GP-VAE (B-S2T)	$\mathbf{0.251 \pm 0.008}$	$\mathbf{0.092 \pm 0.003}$	$\mathbf{0.962 \pm 0.001}$	$\mathbf{0.002 \pm 0.000}$	$\mathbf{0.743 \pm 0.007}$
BRITS	-	-	-	-	$\mathbf{0.742 \pm 0.008}$

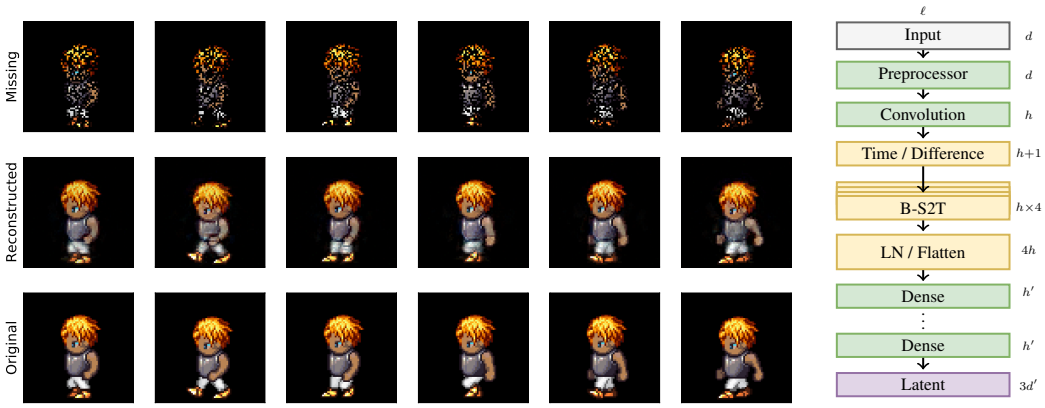

Figure 3: Reconstruction examples from Sprites (left), and encoder network used to represent the means and covariances of the variational posterior with the original in green (right).

et al. (2014), which is depicted in Table 1, while visualizations of the Bayesian posteriors are on Figure 5 in Appendix E.1. The results of the signed-rank test indicate that while S2T^3 is better with moderate probability ($p \geq 0.6$) than 4 of the baselines, FCN-S2T^3 performs better than all with high probability ($p \geq 0.75$), except for MLSTMFCN which is still slightly outperformed by FCN-S2T^3. Note however that *FCN-S2T^3 has fewer parameters than MLSTMFCN by more than* 60%, see Table 5 for a parameter comparison. Also, as MLSTMFCN is the concatenation of an FCN block with an LSTM layer (Karim et al., 2019), the LSTM layer can bottleneck the computations for very long sequences as it scales linearly, while *Seq2Tens layers effectively scale sublinearly in sequence length*, see Appendix D.4 for a computational comparison. In conclusion, we have verified that S2T^3 already performs well on some classification tasks, and by preceding it with an FCN block, its performance is elevated to outperforming all baseline models on the considered problems. This observation is not surprising, as Thm. 2.1 warrants, the full strength of the Seq2Tens layer shines when it is preceded by at least some state-space nonlinearities. Hence, we expect that Seq2Tens layers can find their use as a modular building block being part of larger models.

4.2 Generating sequential data

As another benchmarking experiment, we demonstrate that Seq2Tens layers do not only provide good representations of sequences in discriminative, but also generative models. The considered problem is sequential data imputation for time series and video.

The GP-VAE model. The base model that we upgrade using Seq2Tens layers is the recent GP-VAE (Fortuin et al., 2019), that provides state-of-the-art results for probabilistic sequential data imputation. The GP-VAE is essentially based on the HI-VAE (Nazabal et al., 2018) for handling missing data in variational autoencoders (VAEs) (Kingma & Welling, 2013) adapted to the handling of time series data by the use of a Gaussian process (GP) prior (Williams & Rasmussen, 2006) across time in the latent sequence space to capture temporal dynamics. We provide an in-depth description of the GP-VAE, the experimental setting and baselines in Appendix E.2.

We make one simple change to the GP-VAE architecture without changing any other hyperparameters: we introduce a single bidirectional Seq2Tens layer (B-S2T) into the encoder network and use it in the amortized representation of the means and structured covariances of the variational posterior. As before, the B-S2T layer is preceded by a time-embedding and differencing block, and succeeded by channel flattening and normalization[3] as depicted in Figure 3.

Benchmark results. To make the comparison, we ceteris paribus re-ran all experiments the authors originally included in their paper (Fortuin et al., 2019), which are imputation of Healing MNIST, Sprites, and Physionet 2012. The results are in Table 2, which report the same metrics as used in Fortuin et al. (2019), i.e. negative log-likelihood (NLL, lower is better), mean squared error (MSE, lower is better) on test sets, and downstream classification performance of a linear classifier (AUROC, higher is better). For all other models beside our GP-VAE (B-S2T), the results were borrowed from Fortuin et al. (2019). We observe that simply adding the B-S2T layer improved the result in almost all cases, except for Sprites, where the GP-VAE already achieved a very low MSE score. Additionally, when comparing GP-VAE to BRITS on Physionet, the authors argue that although the BRITS achieves a higher AUROC score, the GP-VAE should not be disregarded as it fits a generative model to the data that enjoys the usual Bayesian benefits of predicting distributions instead of point predictions. Thus, we have shown that by simply adding our layer into the architecture, we managed to elevate the performance of GP-VAE to the same level while retaining these same benefits. We believe the reason for the improvement is a tighter amortization gap in the variational approximation (Cremer et al., 2018), that is achieved by increasing the expressiveness of the encoder. For further details and discussion, see Appendix E.2.

5 Related work and Summary

Related Work. The literature on tensor based models in ML is vast. Related to our approach we mention pars-pro-toto Tensor Networks (Cichocki et al., 2016), that use classical LR decompositions, such as CP (Carroll & Chang, 1970), Tucker (Tucker, 1966), tensor trains (Oseledets, 2011) and tensor rings (Zhao et al., 2019); further, CNNs have been combined with LR tensor techniques (Cohen et al., 2016; Kossaifi et al., 2017) and extended to RNNs (Khrulkov et al., 2019); Tensor Fusion Networks (Zadeh et al., 2017) and its LR variants (Liu et al., 2018b; Liang et al., 2019; Hou et al., 2019). Our main contribution to this literature is the use of the tensor algebra $\mathrm{T}(V)$ with its convolution product $\cdot$, instead of $V^{\otimes m}$ with the outer product $\otimes$ that is used in the above papers. While counter-intuitive to work in a larger space $\mathrm{T}(V)$, the additional algebra structure of $(\mathrm{T}(V), \cdot)$ is the main reason for the nice properties of Φ (universality, making sequences of arbitrary length comparable, convergence in the continuous time limit; see Appendix B) which we believe are in turn the main reason for the strong benchmark performance. Stacked LR sequence transforms allow to exploit this rich algebraic structure with little computational overhead. Another related literature are path signatures in ML (Lyons, 2014; Chevyrev & Kormilitzin, 2016; Graham, 2013; Bonnier et al., 2019; Toth & Oberhauser, 2020). These arise as special case of Seq2Tens (Appendix B) and our main contribution to this literature is that Seq2Tens resolves a well-known computational bottleneck in this literature since it *never needs to compute and store a signature*, instead it *directly and efficiently learns the functional of the signature*.

Summary. We used a classical non-commutative structure to construct a feature map for sequences of arbitrary length. By stacking sequence transforms we turned this into scalable and modular NN layers for sequence data. The main novelty is the use of the free algebra $\mathrm{T}(V)$ constructed from the static feature space V. While free algebras are classical in mathematics, their use in ML seems novel and underexplored. We would like to re-emphasize that $(\mathrm{T}(V), \cdot)$ is not a mysterious abstract space: if you know the outer tensor product $\otimes$ then you can easily switch to the tensor convolution product $\cdot$ by taking sums of outer tensor products, as defined in equation 5. As our experiments show, the benefits of this algebraic structure are not just theoretical but can significantly elevate performance of already strong-performing models.

[3] In this case layer normalization, since the additional minibatch noise injected by a batch normalization layer can be detrimental while training VAEs.

REFERENCES

Pierre Baldi, Søren Brunak, Paolo Frasconi, Giovanni Soda, and Gianluca Pollastri. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics*, 15(11):937–946, 1999.

Robert Bamler and Stephan Mandt. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 380–389. JMLR. org, 2017.

Mustafa Baydogan. Multivariate time series classification datasets. `http://mustafabaydogan.com`, 2015. [Accessed: 2020-06-11].

Mustafa Gokce Baydogan and George Runger. Learning a symbolic representation for multivariate time series classification. *Data Mining and Knowledge Discovery*, 29(2):400–422, 2015a.

Mustafa Gokce Baydogan and George C. Runger. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery*, 30:476–509, 2015b.

Alessio Benavoli, Giorgio Corani, Francesca Mangili, Marco Zaffalon, and Fabrizio Ruggeri. A bayesian wilcoxon signed-rank test based on the dirichlet process. In *International conference on machine learning*, pp. 1026–1034, 2014.

Alessio Benavoli, Giorgio Corani, and Francesca Mangili. Should we really use post-hoc tests based on mean-ranks? *J. Mach. Learn. Res.*, 17(1):152–161, January 2016. ISSN 1532-4435.

Alessio Benavoli, Giorgio Corani, Janez Demšar, and Marco Zaffalon. Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis. *The Journal of Machine Learning Research*, 18(1):2653–2688, 2017.

David M Blei and John D Lafferty. Dynamic topic models. In *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120, 2006.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

P Bonnier, C Liu, and H Oberhauser. Adapted topologies and higher rank signatures. *arXiv preprint arXiv:2005.08897*, 2020.

Patric Bonnier, Patrick Kidger, Imanol Perez Arribas, Cristopher Salvi, and Terry Lyons. Deep signature transforms. *33rd Conference on Neural Information Processing Systems, NeurIPS*, 2019.

Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. Brits: Bidirectional recurrent imputation for time series. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 6775–6785. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7911-brits-bidirectional-recurrent-imputation-for-time-series.pdf`.

J Douglas Carroll and Jih-Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

K. T. Chen. Iterated integrals and exponential homomorphisms. *Proc. London Math. Soc, 4, 502–512*, 1954.

K. T. Chen. Integration of paths, geometric invariants and a generalized Baker-Hausdorff formula. *Ann. of Math. (2), 65:163–178*, 1957.

K. T. Chen. Integration of paths - a faithful representation of paths by non-commutative formal power series. *Trans. Amer. Math. Soc. 89 (1958), 395–407*, 1958.

I. Chevyrev and A. Kormilitzin. A primer on the signature method in machine learning. *arXiv preprint arXiv:1603.03788*, 2016.

Andrzej Cichocki, Namgil Lee, Ivan Oseledets, Anh-Huy Phan, Qibin Zhao, and Danilo P Mandic. Tensor networks for dimensionality reduction and large-scale optimization: Part 1 low-rank tensor decompositions. *Foundations and Trends® in Machine Learning*, 9(4-5):249–429, 2016.

Nadav Cohen, Or Sharir, and Amnon Shashua. On the expressive power of deep learning: A tensor analysis. In *Conference on learning theory*, pp. 698–728, 2016.

Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1078–1086, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR. URL http://proceedings.mlr.press/v80/cremer18a.html.

N Cristianini and J Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge, 2000.

Marco Cuturi and Arnaud Doucet. Autoregressive Kernels For Time Series. *arXiv e-prints*, art. arXiv:1101.0673, Jan 2011.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

J Diehl, K Ebrahimi-Fard, and N Tapia. Time-warping invariants of multidimensional time series. *arXiv preprint arXiv:1906.05823*, 2019.

Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill DF Campbell, and Ivor Simpson. Structured uncertainty prediction networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5477–5485, 2018.

K. Ebrahimi-Fard and F. Patras. Cumulants, free cumulants and half-shuffles. *Proceedings of the Royal Society*, 2015.

Vincent Fortuin, Dmitry Baranchuk, Gunnar Rätsch, and Stephan Mandt. Gp-vae: Deep probabilistic time series imputation, 2019.

Samuel J. Gershman and Noah D. Goodman. Amortized inference in probabilistic reasoning. *Cognitive Science*, 36, 2014.

R Giles. A generalization of the strict topology. *Transactions of the American Mathematical Society*, 1971.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Benjamin Graham. Sparse arrays of signatures for online character recognition. *arXiv preprint arXiv:1308.0371*, 2013.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602 – 610, 2005. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2005.06.042. URL http://www.sciencedirect.com/science/article/pii/S0893608005001206. IJCNN 2005.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pp. 273–278. IEEE, 2013a.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. IEEE, 2013b.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *Iclr*, 2(5):6, 2017.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. doi: 10.1016/0893-6080(91)90009-t. URL https://doi.org/10.1016/0893-6080(91)90009-t.

Ming Hou, Jiajia Tang, Jianhai Zhang, Wanzeng Kong, and Qibin Zhao. Deep multimodal multilinear fusion with high-order polynomial pooling. In *Advances in Neural Information Processing Systems*, pp. 12136–12145, 2019.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, Jul 2019. ISSN 1573-756X. doi: 10.1007/s10618-019-00619-1. URL https://doi.org/10.1007/s10618-019-00619-1.

Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Samuel Harford. Multivariate lstm-fcns for time series classification. *Neural Networks*, 116:237 – 245, 2019. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2019.04.014. URL http://www.sciencedirect.com/science/article/pii/S0893608019301200.

Isak Karlsson, Panagiotis Papapetrou, and Henrik Boström. Generalized random shapelet forests. *Data Min. Knowl. Discov.*, 30(5):1053–1085, September 2016. ISSN 1384-5810. doi: 10.1007/s10618-016-0473-y. URL https://doi.org/10.1007/s10618-016-0473-y.

Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd, 2017.

Valentin Khrulkov, Oleksii Hrinchuk, and Ivan Oseledets. Generalized tensor models for recurrent neural networks. *arXiv preprint arXiv:1901.10801*, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Franz J Király and Harald Oberhauser. Kernels for sequentially ordered data. *Journal of Machine Learning Research*, 2019.

Jean Kossaifi, Zachary C Lipton, Aran Khanna, Tommaso Furlanello, and Anima Anandkumar. Tensor regression networks. *arXiv preprint arXiv:1707.08308*, 2017.

Serge Lang. *Algebra*. Springer-Verlag New York, 2002.

C Leslie and R Kuang. Fast string kernels using inexact matching for protein sequences. *Journal of Machine Learning Research*, 2004.

Yingzhen Li and Stephan Mandt. Disentangled sequential autoencoder, 2018.

Paul Pu Liang, Zhun Liu, Yao-Hung Hubert Tsai, Qibin Zhao, Ruslan Salakhutdinov, and Louis-Philippe Morency. Learning representations from imperfect time series data via tensor rank regularization. *arXiv preprint arXiv:1907.01011*, 2019.

Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 9628–9639, Red Hook, NY, USA, 2018a. Curran Associates Inc.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. Efficient low-rank multimodal fusion with modality-specific factors. *arXiv preprint arXiv:1806.00064*, 2018b.

Terry Lyons. Rough paths, signatures and the modelling of functions on streams. *arXiv preprint arXiv:1405.4537*, 2014.

Dmytro Mishkin and Jiri Matas. All you need is a good init. *arXiv preprint arXiv:1511.06422*, 2015.

Alfredo Nazabal, Pablo M Olmos, Zoubin Ghahramani, and Isabel Valera. Handling incomplete heterogeneous data using vaes. *arXiv preprint arXiv:1807.03653*, 2018.

P. Nemenyi. *Distribution-free Multiple Comparisons*. Princeton University, 1963. URL https://books.google.nl/books?id=nhDMtgAACAAJ.

Ivan V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

C Reutenauer. *Free Lie Algebras*. Clarendon press – Oxford, 1993.

W. Rudin. *Principles of Mathematical Analysis*. Cambridge University Press, 1965.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26(1):43–49, 1978.

Tim Sauer, James A Yorke, and Martin Casdagli. Embedology. *Journal of statistical Physics*, 65 (3-4):579–616, 1991.

Patrick Schäfer and Ulf Leser. Multivariate time series classification with weasel+muse. *ArXiv*, abs/1711.11343, 2017.

Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

Martin Sundermeyer, Tamer Alkhouli, Joern Wuebker, and Hermann Ney. Translation modeling with bidirectional recurrent neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 14–25, 2014.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 3104–3112. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf.

Floris Takens. Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pp. 366–381. Springer, 1981.

C Toth and H Oberhauser. Bayesian learning from sequential data using gaussian processes with signature covariances. *ICML*, 2020.

Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3): 279–311, 1966.

Kerem Sinan Tuncel and Mustafa Gokce Baydogan. Autoregressive forests for multivariate time series modeling. *Pattern Recognition*, 73:202–215, 2018.

Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 2019.

Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1578–1585, 2017.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. Tensor fusion network for multimodal sentiment analysis. *arXiv preprint arXiv:1707.07250*, 2017.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):2008–2026, 2018.

Qibin Zhao, Masashi Sugiyama, Longhao Yuan, and Andrzej Cichocki. Learning efficient tensor representations with ring-structured networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8608–8612. IEEE, 2019.