Relative Positional Encoding for Transformers with Linear Complexity

Antoine Liutkus [*1] **Ondřej Cífka** [*2] **Shih-Lun Wu** [3 4 5] **Umut Şimşekli** [6] **Yi-Hsuan Yang** [3 5] **Gaël Richard** [2]

Abstract

Recent advances in Transformer models allow for unprecedented sequence lengths, due to linear space and time complexity. In the meantime, relative positional encoding (RPE) was proposed as beneficial for classical Transformers and consists in exploiting lags instead of absolute positions for inference. Still, RPE is not available for the recent linear-variants of the Transformer, because it requires the explicit computation of the attention matrix, which is precisely what is avoided by such methods. In this paper, we bridge this gap and present *Stochastic Positional Encoding* as a way to generate PE that can be used as a replacement to the classical additive (sinusoidal) PE and provably behaves like RPE. The main theoretical contribution is to make a connection between positional encoding and cross-covariance structures of correlated Gaussian processes. We illustrate the performance of our approach on the Long-Range Arena benchmark and on music generation.

1. Introduction

1.1. Linear Complexity Transformers

The Transformer model (Vaswani et al., 2017) is a new kind of neural network that quickly became state-of-the-art in many application domains, including the processing of natural language (He et al., 2020), images (Dosovitskiy et al., 2020), audio (Huang et al., 2018; Pham et al., 2020) or bioinformatics (AlQuraishi, 2019) to mention just a few.

The core, novel component of the Transformer is the *attention layer*. It computes M output values $\mathbf{y}_m$ from N input values $\mathbf{v}_n$, all being vectors of an arbitrary dimension. Following classical non-parametric regression principles (Nadaraya, 1964; Watson, 1964), it consists in a simple weighted sum:

$$\mathbf{y}_m = \frac{\sum_n a_{mn}\mathbf{v}_n}{\sum_n a_{mn}}, \tag{1}$$

where each *attention* coefficient $a_{mn} \in \mathbb{R}_+$ – gathered in the $M \times N$ matrix $\mathbf{A}$ – indicates how important the value $\mathbf{v}_n$ is in the computation of the output $\mathbf{y}_m$.

One of the main contributions of the Transformer is an original method to compute these coefficients. D-dimensional feature vectors $\mathbf{k}_n$ and $\mathbf{q}_m$ are attached to all items of the input and output sequences and are called *keys* and *queries*, respectively. Gathering them in the $N \times D$ and $M \times D$ matrices $\mathbf{K}$ and $\mathbf{Q}$, we get *softmax dot-product attention* as:

$$\mathbf{A} = \exp\left(\mathbf{Q}\mathbf{K}^\top \middle/ \sqrt{D}\right) \equiv [a_{mn} = \mathcal{K}(\mathbf{q}_m, \mathbf{k}_n)]_{mn}, \tag{2}$$

where the function exp is applied element-wise. The right-hand side in (2) is a generalization introduced by Tsai et al. (2019) and Choromanski et al. (2020), where $\mathcal{K}$ is a *kernel* function. Parameters pertain to how keys $\mathbf{k}_n$, values $\mathbf{v}_n$ and queries $\mathbf{q}_m$ are obtained from the raw sequences, usually by time-distributed fully connected layers.

The original Transformer architecture (Vaswani et al., 2017) explicitly computes the attention matrix $\mathbf{A}$, leading to a $\mathcal{O}(MN)$ complexity that prevents it from scaling to very long sequence lengths. Although this is not necessarily a problem when sequence lengths are barely on the order of a few hundreds, as in some language processing tasks, it is

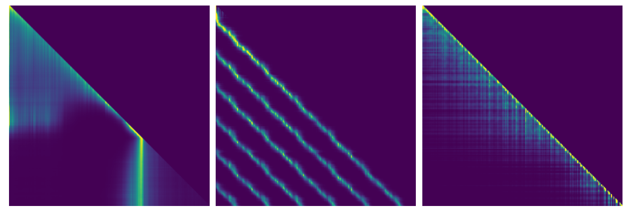

Figure 1. Examples of attention patterns observed in the Performers trained for pop piano music generation (section 3.2) at inference time, for sequence length $M = N = 3\,072$ while training sequences have length $2\,048$. (left) Absolute PE. (middle) Sinusoidal SPE. (right) Convolutional SPE. Note that SPE never requires computing these full attention patterns.

[*]Equal contribution [1]Inria, Zenith Team, UMR LIRMM, Univ. Montpellier, France [2]LTCI, Télécom Paris, Institut Polytechnique de Paris, France [3]Research Center for IT Innovation, Academia Sinica, Taiwan [4]National Taiwan University, Taiwan [5]Taiwan AI Labs, Taiwan [6]INRIA – Département d'Informatique de l'École Normale Supérieure – PSL Research University, Paris, France. Correspondence to: Liutkus Antoine <antoine.liutkus@inria.fr>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

prohibitive for very large signals like high-resolution images or audio.

Focusing on this scalability issue, several approaches have been recently investigated to allow for long sequences:

- *Attention clustering* schemes group items among which dependencies are computed through regular attention. This is either done by using simple proximity rules within the sequences, leading to chunking strategies (Dai et al., 2019), or by clustering the keys and values (Roy et al., 2020). Inter-cluster dependencies are either ignored or summarized via fixed-length context vectors that are coined in as *memory* (Wu et al., 2020).
- Assuming the attention matrix to be *sparse*. In this case, only a few a_{mn} are nonzero (Child et al., 2019).
- Assuming $\mathbf{A}$ has a particular (low-rank) *structure* and can be decomposed as the product of two smaller matrices. A prototypical example is the Linformer (Wang et al., 2020b), which is limited to fixed-length inputs. Another very recent line of research in this same vein takes:
$$\mathbf{A} \approx \phi(\mathbf{Q})\phi(\mathbf{K})^{\top}, \tag{3}$$
where $\phi : \mathbb{R}^D \rightarrow \mathbb{R}^R$ is a non-linear *feature map* applied to each key $\mathbf{k}_n$ and query $\mathbf{q}_m$, and $R \ll \min(M, N)$ (Shen et al., 2020; Katharopoulos et al., 2020).
- When $\mathcal{K}$ in (2) is a positive (semi)definite kernel, the Performer (Choromanski et al., 2020) leverages *reproducing kernel Hilbert spaces* to show that a random ϕ may be used to exploit this convenient decomposition (3) *on average*, even when $\mathbf{A}$ is not low rank:
$$\mathcal{K} \succeq 0 \Leftrightarrow \mathbf{A} = \mathbb{E}_{\phi}\left[\phi(\mathbf{Q})\phi(\mathbf{K})^{\top}\right], \tag{4}$$
where ϕ is drawn from a distribution that depends on $\mathcal{K}$. A simple example is $\phi_{\mathbf{W}}(\mathbf{k}_n) = \max(0, \mathbf{W}\mathbf{k}_n)$, with a random $\mathbf{W} \in \mathbb{R}^{R \times D}$ for some $R \in \mathbb{N}$.

Whenever an efficient scheme like (3) or (4) is used, the outputs can be obtained without computing the attention coefficients a_{mn}, as in (10).[1]

1.2. Positional Encoding

In Transformer networks, the outputs $\mathbf{y}_m$ are computed as linear combinations of *all* input values $\mathbf{v}_n$, weighted by attention coefficients a_{mn}. In sequence modeling, it is reasonable to assume that the actual *positions* m and n should play a role in the computation, in addition to the *content* at these locations; otherwise, any permutation of the sequence would lead to the same output. Two core approaches were undertaken to incorporate position information:

- The original Transformer (Vaswani et al., 2017) adds this information to the inputs of the network, i.e. before the first attention layer. This can be equivalently understood as augmenting the keys, values and queries:
$$\mathbf{k}_n \leftarrow \mathbf{k}_n + \overline{\mathbf{k}}_n,\, \mathbf{v}_n \leftarrow \mathbf{v}_n + \overline{\mathbf{v}}_n,\, \mathbf{q}_m \leftarrow \mathbf{q}_m + \overline{\mathbf{q}}_m, \tag{5}$$
where we write $\overline{\mathbf{k}}_n \in \mathbb{R}^D$ for the *keys positional encoding* (PE; Sukhbaatar et al., 2015) at position $n \in \mathbb{N}$ and analogously for values and queries. Vaswani et al. propose a deterministic scheme based on trigonometric functions, which is shown to work as well as trainable embeddings.
- As an example of positional encoding *in the attention domain*, a *relative positional* encoding (RPE) was proposed by Shaw et al. (2018), building on the idea that time lags $m - n$ are more important than absolute positional encoding (APE) for prediction. It is written as:
$$\mathbf{A} = \exp\Big(\big(\mathbf{Q}\mathbf{K}^{\top} + \mathbf{\Omega}\big)\Big/\sqrt{D}\Big), \text{ with:} \tag{6}$$
$$\mathbf{\Omega} \equiv \left[\omega_{mn} = \sum_{d=1}^{D} q_{md}\mathcal{P}_d(m-n)\right]_{mn}. \tag{7}$$
The terms $\mathcal{P}_d$ now act as D different encodings for *time lags* selected based on the queries. This change is advocated as bringing important performance gains in many application areas and has enjoyed a widespread use ever since.

Although writing down the positional encoding in the attention domain is beneficial for performance (Shaw et al., 2018; Dai et al., 2019; Tsai et al., 2019), we are only aware of implementations that either require the computation of $\mathbf{A}$, or clustered attention schemes, which *in fine* decompose $\mathbf{A}$ into smaller attention matrices, and *compute them*. This is in sharp contrast to (3) and (4), which never compute the attention matrix.

Our contributions can be summarized as follows:

- We propose *Stochastic Positional Encoding* (SPE) as a general PE scheme *in the keys domain*, that enforces a particular attention pattern devised *in the attention domain*. This enables RPE without explicit computation of attention. To our knowledge, it is the first RPE strategy that is compatible with $\mathcal{O}(N)$ Transformers like Choromanski et al. (2020) and Katharopoulos et al. (2020).
- We study the impact of SPE on performance on the Long-Range Arena benchmark (Tay et al., 2021) and two music generation tasks. Since RPE was so far limited to short sequences, we believe this is the first study of its advantages on long-range predictions. Our results demonstrate better validation losses and extrapolation ability.
- We provide additional resources on our companion website,[2] including Python implementations of SPE for PyTorch and JAX/Flax.

[1] A somewhat related strategy is used by the recent LambdaNetworks (Bello, 2020), which encapsulate the key-value information as a so-called *lambda* function to be applied query-wise, hence also avoiding the computation of a full attention matrix.

[2] https://cifkao.github.io/spe/

Algorithm 1 Stochastic Positional Encoding.

Input
- position kernel $\mathcal{P}(m,n)$, number of replicas R.
- initial $M \times D$ and $N \times D$ queries $\mathbf{Q}$ and keys $\mathbf{K}$.

Positional encoding:
- Draw the D independent couples $\{\overline{\mathbf{Q}}_d, \overline{\mathbf{K}}_d\}_d$ of $M \times R$ and $N \times R$ matrices as in section 2.1
- Set $\widehat{\mathbf{Q}}$ and $\widehat{\mathbf{K}}$ as in (16) and (17)

Inference compute outputs $\mathbf{Y}$ with the $\mathcal{O}(N)$ Transformer:

$$\mathbf{Y} \leftarrow \mathrm{diag}(\mathbf{d})^{-1}\left[\phi\left(\widehat{\mathbf{Q}}\right)\left[\phi\left(\widehat{\mathbf{K}}\right)^\top \mathbf{V}\right]\right] \tag{10}$$

with $\mathbf{d} = \phi(\widehat{\mathbf{Q}})\left[\phi\left(\widehat{\mathbf{K}}\right)^\top \mathbf{1}_N\right]$ and ϕ discussed in (3)/(4).

2. Stochastic Positional Encoding

Index set and notation. We assume that the input/output sequences are indexed by $n, m \in \mathbb{T}$, where $\mathbb{T}$ is the *index set*. For regularly sampled sequences, we have $\mathbb{T} = \mathbb{N}$, but more settings are possible, like irregularly sampled time series ($\mathbb{T} = \mathbb{R}$) or images ($\mathbb{T} = \mathbb{N}^2$). In any case, the particular lists of input / output locations under consideration are written: $\mathcal{N}$ and $\mathcal{M}$, with respective sizes N and M (the case $\mathcal{N} = \mathcal{M}$ is called *self-attention*). The corresponding keys and values are hence indexed as $\{\mathbf{k}_n\}_{n\in\mathcal{N}}$ and $\{\mathbf{v}_n\}_{n\in\mathcal{N}}$, while queries are $\{\mathbf{q}_m\}_{m\in\mathcal{M}}$. For convenience, we write a_{mn} for the entries of the $M \times N$ attention matrix $\mathbf{A}$.
We use bold uppercase for matrices, bold lowercase for vectors and a NumPy-like notation: if $\mathbf{X}_k$ is a $I \times J$ matrix, $\mathbf{x}_{k,i}$ and $\mathbf{x}_{k,:,j}$ stand for its i^{th} row and j^{th} column, respectively.

Assumptions. In the remainder of this paper, we will seek an attention matrix $\mathbf{A}$ given by:

$$\mathbf{A} = \exp\left(\left[\sum_{d=1}^{D} q_{md}\mathcal{P}_d(m,n)k_{nd}\right]_{mn} \Big/ \sqrt{D}\right), \tag{8}$$

where $\{\mathcal{P}_d\}_{d=1}^{D}$ are *position kernels*. Defining $\mathbf{P}_d \equiv [\mathcal{P}_d(m,n)]_{mn}$, this can be written in matrix form as:

$$\mathbf{A} = \exp\left(\sum_{d=1}^{D} \mathrm{diag}\left(\mathbf{q}_{:,d}\right)\mathbf{P}_d\,\mathrm{diag}(\mathbf{k}_{:,d}) \Big/ \sqrt{D}\right), \tag{9}$$

which is understood as having D positional attention templates $\mathbf{P}_d$ jointly activated by the queries $\mathbf{q}_{:,d}$ and keys $\mathbf{k}_{:,d}$. Original RPE (7) can be seen as a special case, where some entries are kept constant.

Positional attention as covariance. The key idea for SPE is to see the attention kernel $\mathcal{P}_d(m,n)$ as a *covariance*:

$$(\forall \mathcal{M}, \mathcal{N})\,(\forall m,n)\,\mathcal{P}_d(m,n) = \mathbb{E}\left[\overline{Q}_d(m)\overline{K}_d(n)\right], \tag{11}$$

where $\overline{Q}_d(m)$ and $\overline{K}_d(n)$ are two real and zero-mean random variables, which will be chosen with the single condition that their covariance function matches $\mathcal{P}_d$. Semantically, they should be understood as (randomly) encoding position m for queries and position n for keys, respectively. When multiplied together as in dot-product attention, they yield the desired attention template $\mathcal{P}_d(m,n)$ on average. The central intuition is that the actual positional encodings do not matter as much as their dot-product.

In what follows, we will impose specific structures on the cross-covariance $\mathcal{P}_d(m,n)$, which will in turn allow us to design *random processes* $\overline{Q}_d = \{\overline{Q}_d(m)\}_{m\in\mathcal{M}}$ and $\overline{K}_d = \{\overline{K}_d(n)\}_{n\in\mathcal{N}}$ such that (11) holds. The core advantage of this construction is to allow for $\mathbf{P}_d$ to be factorized. Let us for now assume that we construct the distributions of $\{\overline{Q}_d(m), \overline{K}_d(n)\}_d$ in such a way that we can sample from them (we will see how in section 2.1) and consider R independent realizations of them for given $\mathcal{M}$ and $\mathcal{N}$, gathered in the $M \times R$ and $N \times R$ matrices $\overline{\mathbf{Q}}_d$ and $\overline{\mathbf{K}}_d$:

$$\overline{\mathbf{Q}}_d \equiv [q_{d,m,r} \sim \overline{Q}_d(m)]_{mr}, \;\; \overline{\mathbf{K}}_d \equiv [k_{d,n,r} \sim \overline{K}_d(n)]_{nr}. \tag{12}$$

For large R, by the law of large numbers, we obtain:

$$\mathbf{P}_d \approx \left[\overline{\mathbf{Q}}_d\overline{\mathbf{K}}_d^\top\right]/R. \tag{13}$$

This leads $\mathbf{A}$ in (9) to be given by:

$$\mathbf{A} \approx \exp\left(\sum_{d=1}^{D}\mathrm{diag}\left(\mathbf{q}_{:,d}\right)\frac{\overline{\mathbf{Q}}_d\overline{\mathbf{K}}_d^\top}{R}\,\mathrm{diag}(\mathbf{k}_{:,d})/\sqrt{D}\right) \tag{14}$$

$$\approx \exp\frac{\left(\sum_{d=1}^{D}\mathrm{diag}\left(\mathbf{q}_{:,d}\right)\overline{\mathbf{Q}}_d\right)\left(\sum_{d=1}^{D}\mathrm{diag}(\mathbf{k}_{:,d})\overline{\mathbf{K}}_d\right)^\top}{R\sqrt{D}}. \tag{15}$$

Here, a *crucial* observation is that for large R, the cross-terms $\overline{\mathbf{Q}}_d\overline{\mathbf{K}}_{d'\neq d}^\top$ are negligible due to independence, provided that the means of the processes are selected to be zero. Finally, picking queries and keys as:

$$\widehat{\mathbf{Q}} \leftarrow \sum_{d=1}^{D}\mathrm{diag}\left(\mathbf{q}_{:,d}\right)\overline{\mathbf{Q}}_d/\sqrt[4]{DR}, \tag{16}$$

$$\widehat{\mathbf{K}} \leftarrow \sum_{d=1}^{D}\mathrm{diag}(\mathbf{k}_{:,d})\overline{\mathbf{K}}_d/\sqrt[4]{DR}, \tag{17}$$

we see from (15-17) that we get back to the usual multiplicative scheme (2) with $\mathbf{A} = \exp(\widehat{\mathbf{Q}}\widehat{\mathbf{K}}^\top/\sqrt{R})$, where the queries/keys now have dimension R and can be used in (10) to directly get outputs without computing $\mathbf{A}$.

The procedure is summarized in Algorithm 1: we provide a way (16-17) to achieve PE in the *keys domain*, such that the desired model (8) is enforced in the *attention domain*, parameterized by the attention kernels $\mathcal{P}_d$. Interestingly, this is done without ever computing attention matrices, complying with $\mathcal{O}(N)$ Transformers. The remaining challenge, which we discuss next, is to generate $\overline{\mathbf{Q}}_d$ and $\overline{\mathbf{K}}_d$ enforcing (13).

2.1. Drawing Stochastic Positional Encodings

Inspecting (11), we notice that our objective is to draw samples from D pairs of centered random processes $\{\overline{Q}_d, \overline{K}_d\}_d$, with a prescribed cross-covariance structure $\mathcal{P}_d$. It is reasonable to use Gaussian processes for this purpose (Williams & Rasmussen, 2006), which have the maximum entropy for known mean and covariance. Such distributions are frequently encountered in geophysics in the *co-kriging* literature (Matheron, 1963; Genton & Kleiber, 2015), where scientists routinely handle correlated random fields. The particular twists of our setup are: we have a *generative* problem, e.g. as in Vořechovský (2008); however, as opposed to their setting, we are not directly interested in the marginal covariance function of each output, provided that the desired cross-covariance structure holds.

The most straightforward application of SPE arises when we pick $\mathcal{P}_d(m,n) = \mathcal{P}_d(m-n)$, i.e. a stationary position kernel, which was coined in as choosing *relative* attention in Shaw et al. (2018) and boils down to enforcing a *Toeplitz* structure for the cross-covariance matrix $\mathbf{P}_d \equiv [\mathcal{P}_d(m-n)]_{m,n}$ between $\overline{Q}_d$ and $\overline{K}_d$.

We propose two variants of SPE to handle this important special case, illustrated in Figure 2. The first variant yields *periodic* covariance functions. It can be beneficial whenever attention should not vanish with large lags, as in traffic prediction (Xue & Salim, 2020) or, as we show, in music generation. The second variant generates *vanishing* covariance functions; a concept which has recently been shown useful (Wang et al., 2021), and notably yields smaller validation losses in some of our experiments.

Variant I. Relative and periodic attention (`sineSPE`). In our first approach, we consider the case where $\mathcal{P}_d$ is periodic, which gets a convenient treatment. We assume:

$$\mathcal{P}_d(m,n) = \sum_{k=1}^{K} \lambda_{kd}^2 \cos(2\pi f_{kd}\,(m-n) + \theta_{kd})\,, \quad (18)$$

where $K \in \mathbb{N}$ is the number of *sinusoidal* components and $\mathbf{f}_d \in [0\ 1]^K$, $\boldsymbol{\theta}_d \in [-\pi\ \pi]^K$ and $\boldsymbol{\lambda}_d \in \mathbb{R}^K$ gather their K frequencies, phases, and weights, respectively. By using the matrix notation, we can rewrite (18) as:

$$\mathbf{P}_d = \boldsymbol{\Omega}(\mathcal{M}, \boldsymbol{f}_d, \boldsymbol{\theta}_d)\,\mathrm{diag}\Big(\ddot{\boldsymbol{\lambda}}_d\Big)^2 \boldsymbol{\Omega}(\mathcal{N}, \boldsymbol{f}_d, \mathbf{0})^\top, \quad (19)$$

where $\ddot{\mathbf{v}} \equiv \left[v_{\lfloor p/2 \rfloor}\right]_p \in \mathbb{R}^{2K}$ denotes a twice upsampled version of a vector $\mathbf{v} \in \mathbb{R}^K$, $\lfloor \cdot \rfloor$ denotes the floor operation, and for an index set $\mathcal{I}$, $\boldsymbol{\Omega}(\mathcal{I}, \mathbf{a}, \mathbf{b})$ is a matrix of size $|\mathcal{I}| \times 2K$, with entries (0-based indexing):

$$[\boldsymbol{\Omega}\,(\mathcal{I}, \boldsymbol{a}, \boldsymbol{b})]_{nl} = \begin{cases} \cos(2\pi a_k n + b_k) & \text{if } l = 2k \\ \sin(2\pi a_k n + b_k) & \text{if } l = 2k+1 \end{cases}$$

It can be shown that if $\boldsymbol{\theta}_d = \mathbf{0}$ and $\mathcal{M} = \mathcal{N}$, we get back to the (unique) Vandermonde decomposition for positive definite Toeplitz matrices[3] (Yang et al., 2016), which boils down in our context to assuming that $\forall\tau, \mathcal{P}_d(0) \geq \mathcal{P}_d(\tau)$. Since this is not always desirable, we keep the more general (19).

At this point, we can easily build $\overline{\mathbf{Q}}_d$ and $\overline{\mathbf{K}}_d$. We draw a $2K \times R$ matrix $\mathbf{Z}_d$ with independent and identically distributed (i.i.d.) Gaussian entries of unit variance, and define:

$$\overline{\mathbf{Q}}_d \leftarrow \boldsymbol{\Omega}(\mathcal{M}, \boldsymbol{f}_d, \boldsymbol{\theta}_d)\,\mathrm{diag}\Big(\ddot{\boldsymbol{\lambda}}_d\Big)\mathbf{Z}_d/\sqrt{2K}\,, \quad (20)$$

$$\overline{\mathbf{K}}_d \leftarrow \boldsymbol{\Omega}(\mathcal{N}, \boldsymbol{f}_d, \mathbf{0})\,\mathrm{diag}\Big(\ddot{\boldsymbol{\lambda}}_d\Big)\mathbf{Z}_d/\sqrt{2K}\,. \quad (21)$$

It is easy to check that such a construction leads to (13). Its parameters are $\{\mathbf{f}_d, \boldsymbol{\theta}_d, \boldsymbol{\Lambda}_d\}_d$, which can be trained through stochastic gradient descent (SGD) as usual.

Variant II. Relative (vanishing) attention with regular sampling (`convSPE`). Due to their periodic structure, the covariance functions generated by Variant I are *non-vanishing*. Yet, our framework is flexible enough to allow for vanishing covariance structures, which may be more desirable depending on the application (Wang et al., 2021).

As opposed to Variant I, where we imposed a specific structure on $\mathcal{P}_d$, we will now follow an indirect approach, where $\mathcal{P}_d$ will be *implicitly* defined based on our algorithmic construction. In this case, we assume that the signals are regularly sampled (typical in e.g. text, images, audio), and we will exploit the structure of Gaussian random matrices and basic properties of the convolution operation.

For ease of notation, we assume self attention, i.e. $\mathcal{M} = \mathcal{N}$. Let $\{\boldsymbol{\Phi}_{\boldsymbol{d}}^{\boldsymbol{Q}}, \boldsymbol{\Phi}_d^K\}_d$ denote a collection of *filters*, which will ultimately be learned from training data. The size and the dimension of these filters can be chosen according to the input data (i.e. can be vectors, matrices, tensors). We then propose the following procedure, which leads to a Toeplitz $\mathbf{P}_d$ by means of *convolutions*:

- We first draw an $M \times R$ random matrix $\mathbf{Z}_d$ with i.i.d. standard Gaussian entries. For multidimensional signals, $\mathbf{Z}_d$ gathers R random vectors, matrices, cubes, etc.
- The desired $\overline{\mathbf{Q}}_d$ and $\overline{\mathbf{K}}_d$ are obtained by convolving $\mathbf{Z}_d$ with respective filters $\boldsymbol{\Phi}_{\boldsymbol{d}}^{\boldsymbol{Q}}$ and $\boldsymbol{\Phi}_d^K$:

$$\overline{\mathbf{Q}}_d = \mathbf{Z}_d * \boldsymbol{\Phi}_d^Q\,,\ \overline{\mathbf{K}}_d = \mathbf{Z}_d * \boldsymbol{\Phi}_d^K\,, \quad (22)$$

where $*$ denotes convolution with appropriate dimension (e.g. 1D, 2D or 3D). Using convolutions with finite filters

[3] If $\mathbf{P}_d \succeq 0$ and $K \geq N$, (19) still holds but is not unique.

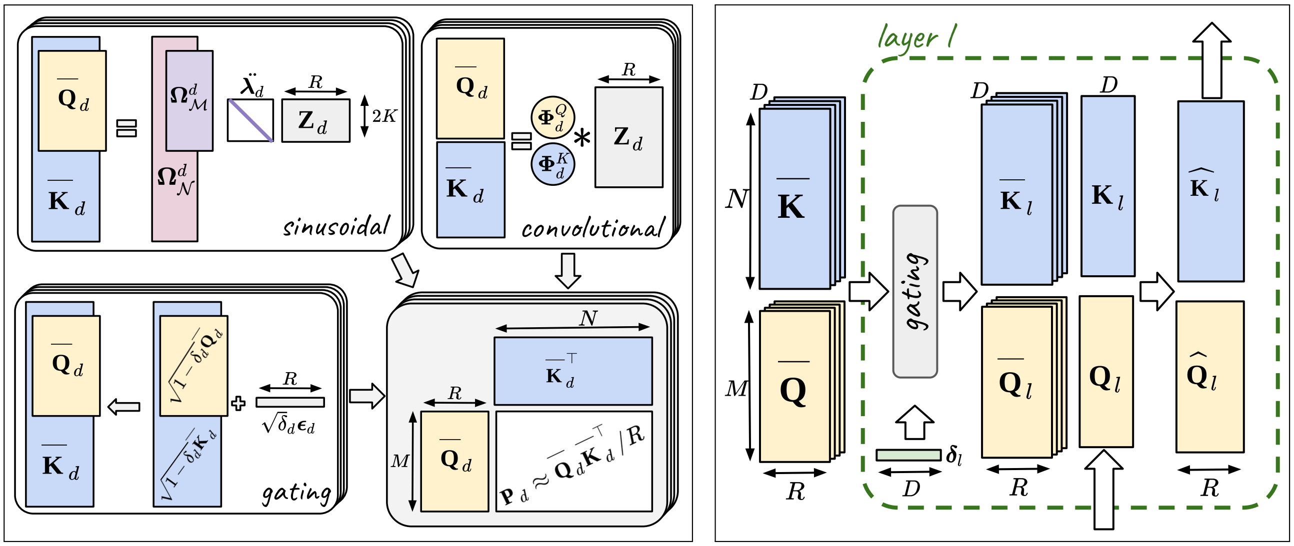

Figure 2. (left) Generation of $\overline{\mathbf{Q}}$ and $\overline{\mathbf{K}}$ in SPE, which approximate the templates $\mathbf{P}_d$ when multiplied together. (right) $\overline{\mathbf{Q}}$ and $\overline{\mathbf{K}}$ can be shared across layers. At each layer l, different gating is (optionally) used, before applying (16-17) to generate new queries $\mathbf{Q}$ and keys $\mathbf{K}$.

ensures vanishing covariance, as proven in the appendix.

Due to the independence of the entries of $\mathbf{Z}_d$, for large R, the product $\mathbf{Z}_d\mathbf{Z}_d^\top/R$ will tend to the identity matrix. Given the fact the convolution operations in (22) can be equivalently expressed as a multiplication by triangular Toeplitz matrices constructed from the respective filters, it can be shown that, as $R \to \infty$, $\frac{1}{R}\overline{\mathbf{Q}}_d\overline{\mathbf{K}}_d^\top$ tends to the product of two triangular Toeplitz matrices. Hence, by using the properties of triangular Toeplitz matrices (cf. Kucerovsky et al. 2016, Theorem 2.4), we conclude that, as $R \to \infty$, our construction yields a Toeplitz matrix $\mathbf{P}_d$ as desired.
This approach is parameterized by the filters$\{\mathbf{\Phi}_d^Q, \mathbf{\Phi}_d^K\}_d$, which will be learned from training data through SGD.

The variety of attention patterns $\mathcal{P}(m-n)$ that can be obtained directly depends on the kernel sizes, which is a classical result from signal processing (Vetterli et al., 2014). Cascading several convolutions as in the VGGNet (Simonyan & Zisserman, 2014) may be a convenient way to augment the expressive power of this convolutional SPE variant.

From a more general perspective, the two operations in (22) can be understood as producing PE through filtering white noise, which is the core idea we introduce for PE. Other classical signal processing techniques may be used like using *infinite impulse response* filters. Such considerations are close to the ideas proposed in (Engel et al., 2020).

To summarize, the core difference between the two proposed constructions (20-21) and (22) lies in the behaviour of RPE beyond a maximum lag, implicitly defined through the frequencies $\mathbf{f}_d$ for (20-21) and through the sizes of the filters for (22). While the sinusoidal construction leads to a periodic RPE, the filtering construction leads to a vanishing RPE, which is called *monotonic* in (Wang et al., 2021). Both may be the desired option depending on the application.

2.2. Gated SPE

Although RPE and the generalization (9) we propose are novel and efficient strategies to handle position information, it may be beneficial to also allow for attention coefficients that are computed without positional considerations, simply through $\langle \mathbf{q}_m, \mathbf{k}_n \rangle$. As a general *gating* mechanism, we propose to weight between positional and non-positional attention through a *gate parameter* $\delta_d \in [0\ 1]$:

$$\mathbf{P}_d \equiv \left[\delta_d + (1-\delta_d)\mathcal{P}_d(m,n)\right]_{m,n}. \tag{23}$$

This gating scheme can be implemented simply by augmenting $\overline{\mathbf{Q}}_d$ and $\overline{\mathbf{K}}_d$ generated as above through:

$$\overline{\mathbf{q}}_{d,m} \leftarrow \sqrt{1-\delta_d}\,\overline{\mathbf{q}}_{d,m} + \sqrt{\delta_d}\,\boldsymbol{\epsilon}_d\,, \tag{24}$$

$$\overline{\mathbf{k}}_{d,m} \leftarrow \sqrt{1-\delta_d}\,\overline{\mathbf{k}}_{d,m} + \sqrt{\delta_d}\,\boldsymbol{\epsilon}_d\,, \tag{25}$$

where $\boldsymbol{\epsilon}_d \in \mathbb{R}^R$ in (24) and (25) is the same and has i.i.d. standard Gaussian entries.

In practice, we can share some SPE parameters across the network, notably across layers, to strongly reduce computing time and memory usage. In our implementation, *sharing* means generating a single instance of $\overline{\mathbf{Q}}$ and $\overline{\mathbf{K}}$ for each head, on which a layer-wise gating is applied, before achieving PE through (16-17). This is illustrated in Figure 2.

Table 1. Long-Range Arena results (higher scores are better). Mean and standard deviation of accuracy over three runs is reported, except for Performer with convolutional SPE, where only a single run was completed. For comparison, the best result reported by Tay et al. (2021), along with the name of the best-performing model (in parentheses), is included.

	ListOps	Text	Retrieval	Image
Best result from Tay et al. (2021)	37.27 (Reformer)	65.90 (Linear Trans.)	59.59 (Sparse Trans.)	44.24 (Sparse Trans.)
Linear Transformer-ReLU from Tay et al.	18.01	65.40	53.82	42.77
Performer-softmax (`APE`)	**17.80** $\pm$ 0.00	62.58 $\pm$ 0.22	59.84 $\pm$ 1.46	41.81 $\pm$ 1.16
Performer-softmax + `sineSPE`	17.43 $\pm$ 0.32	62.60 $\pm$ 0.50	60.00 $\pm$ 1.20	41.12 $\pm$ 1.70
Performer-softmax + `convSPE`	**17.80**	60.94	57.22	40.06
Linear Transformer-ReLU (`APE`)	17.58 $\pm$ 1.01	63.98 $\pm$ 0.05	58.78 $\pm$ 0.93	**42.25** $\pm$ 0.01
Linear Transformer-ReLU + `sineSPE`	**17.80** $\pm$ 0.00	**64.09** $\pm$ 0.62	**62.39** $\pm$ 0.59	41.21 $\pm$ 1.18
Linear Transformer-ReLU + `convSPE`	9.50 $\pm$ 1.17	63.23 $\pm$ 1.31	61.00 $\pm$ 1.34	39.96 $\pm$ 1.31

3. Experiments

3.1. Long-Range Arena

Experimental setup. We evaluate the proposed method in the Long-Range Arena (LRA; Tay et al., 2021), a benchmark for efficient Transformers, consisting of sequence classification tasks with a focus on long-range dependencies. We use the following tasks from this benchmark:

- *ListOps*: parsing and evaluation of hierarchical expressions. a longer variant of (Nangia & Bowman, 2018);
- *Text*: movie review sentiment analysis on the IMDB corpus (Maas et al., 2011);
- *Retrieval*: article similarity classification on the All About NLP (AAN) corpus (Radev et al., 2013);
- *Image*: object recognition on the CIFAR10 dataset (Krizhevsky, 2009) represented as pixel sequences.

The tasks are challenging due to the large sequence lengths, deliberately increased by choosing a character-/pixel-level representation. An overview of the tasks can be found in the appendix. We do not include *Pathfinder* (a synthetic image classification task) as we were unable to reproduce the results of Tay et al. on this task, even through correspondence with the authors.

We evaluate SPE (the gated variant) on two efficient Transformer models: the (softmax) Performer (Choromanski et al., 2020), and a Linear Transformer (Katharopoulos et al., 2020) with a ReLU feature map, i.e. choosing $\phi(\cdot) = \max(0, \cdot)$ element-wise in (3).[4] It should be noted that the ReLU feature map does not approximate the softmax kernel, which SPE is designed for (see assumption 8). Nevertheless, it is possible to use SPE with any feature map in practice, allowing us to include Linear Transformer-ReLU as an interesting test of generalization to alternative kernels.

[4] A model named 'Performer' is reported by Tay et al., but communication with the authors revealed it to be in fact equivalent to our Linear Transformer-ReLU, as it does not use random features. To avoid confusion, we refer to this model as such herein.

We adopt the configuration of Tay et al., only changing the PE and the batch sizes/learning rates to allow training on limited hardware with similar results. All other hyperparameters are kept identical to the original LRA. It is worth noting that the *Image* models are different from the rest in that they employ a single-layer network and only use the first position for prediction, dramatically limiting their ability to benefit from relative positional information.

Since we observe some variation between different runs, we train and evaluate each model 3 times (except for Performer with convolutional SPE, which is computationally more costly) and report the mean and standard deviation of the results.

The results of the benchmark are given in Table 1. The accuracies achieved by the baseline Linear Transformer-ReLU (`APE`) are similar to or surpass those reported by Tay et al., which is a clear validation of our experimental setup.

Discussion. Results on ListOps are poor overall, with accuracies around 17 %. This complies with Tay et al. (2021), who reasoned that "kernel-based models [e.g. Performer, Linear Transformers] are possibly not as effective on hierarchically structured data," leaving room for improvement. We also hypothesize this is largely due to some known issues with the training data for this task, which unfortunately have not been fixed at the time of this writing.[5]

Regarding performance of SPE, we first notice that the `sineSPE` variant yields the best results on three tasks, which is a strong achievement and validates our approach, especially considering the difficulty of this evaluation benchmark. While it is only marginally better than APE for *ListOps* and *Text*, it is worth mentioning that `sineSPE` combined with the Linear Transformer-ReLU yields an accuracy improvement of $\sim$3 % on *Retrieval* compared to the best result obtained by Tay et al. (2021).

[5] Currently, the official data loader for ListOps inadvertently strips some characters from the input sequences.

Regarding `convSPE`, its performance in the LRA is not as remarkable as it is for the music generation experiment reported later in section 3.2. This mitigated result appears somewhat in contradiction with the discussion found in Wang et al. (2021), which presents vanishing attention as a desirable property of PE. On the contrary, we empirically observe that our non-vanishing sinusoidal version `sineSPE` does behave better in these particular tasks.

Finally, the superior results of `APE` on *Image* are not unexpected, given the limited ability of these models to exploit relative positions. On the contrary, the relatively good performance of SPE on this task is in fact remarkable, especially considering that the baseline systems for this task use *learnable* APE.

As we will see later in our music generation experiments, there are tasks where our proposed SPE clearly yields remarkable improvements. Here in the LRA, we notice that it does not result in an obvious and systematic boost in performance. This raises interesting considerations:

(i) *The variance of the Monte Carlo estimator might be problematic.* We are enthusiastic about the elegant formulation of stochastic feature maps as in the Performer, which was a strong inspiration. Still, we must acknowledge that their computation relies on a Monte Carlo estimator (15). We suspect that the variance of the estimator might play a role in the final performance in large dimensions, which opens up the direction of exploring variance-reduced estimation methods, rather than plain Monte Carlo.

(ii) *LRA tasks might not benefit from strong (R)PE schemes.* The LRA was designed to compare Transformer *architectures*, filling a gap in this domain and standing as the *de facto* standard, justifying our choice. Still, although PE is known to be important in many cases, it is not known whether it is so in the LRA tasks. We feel that there is room for such a specialized comparison, which is scheduled in our future work, possibly leading to new long-range tasks where PE is critical.

3.2. Pop Piano Music Generation

In our music generation experiments (this subsection and section 3.3), music is represented as sequences of symbols (tokens) and a Performer (Choromanski et al., 2020) is used as an autoregressive language model, which predicts a probability distribution over the next token given the past context. At test time, a new sequence is generated by iteratively sampling the next token, as commonly done in text generation.

Experimental setup. We train Performers for music generation, with 24 layers and 8 heads per layer on a dataset composed of 1 747 pop piano tracks, encoded using the recently proposed *Revamped MIDI-derived format* (REMI; Huang & Yang, 2020). The sequences are composed of

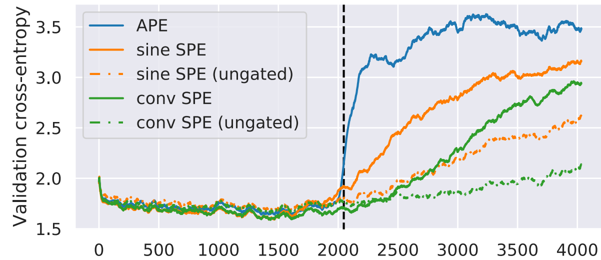

Figure 3. Validation cross-entropy vs. token position on pop piano music generation task. (lower is better; the **black** vertical line indicates the maximum position to which the models are trained.)

metrical tokens: `bar`, `subbeat`, and `tempo`, which represent musical timing; and *note* tokens: `chord`, `pitch`, `duration`, and `volume`, which describe the musical content (see the appendix for more details). We hold out 5% of the songs as the validation set.

We train the models with sequence length $N = 2\,048$, corresponding to $\sim$1 minute of music. The only difference between our models is the PE strategy. We consider baseline `APE`, as well as SPE: sinusoidal or convolutional, with or without gating, resulting in 5 different models.

Results and discussion. For qualitative assessment, we first display in Figure 1 one attention pattern for each PE model: `APE` and (gated) `sineSPE`/`convSPE`, obtained as an average over 20 from-scratch generations for a chosen (layer, head). More similar plots can be found in appendix. Interestingly, we notice that for early layers, `APE` attention does not go much beyond training sequence length. This behaviour is not found in SPE variants, which consistently attend to all positions. Another remarkable feature of the proposed model (only displayed in the appendix) is that *gating* as described in section 2.2 visually disables PE altogether for some layers/heads, in which case attention is global.

Since the literature suggests that RPE improves generalization performance (Shaw et al., 2018; Zhou et al., 2019; Rosendahl et al., 2019), we display validation cross-entropy computed with teacher forcing (Williams & Zipser, 1989) in Figure 3, as a function of the target token position. The values would indicate how well the models predict the token at a certain position given the preceding tokens, for tracks in the validation set. We notice that all SPE variants, especially `convSPE`, behave much better than `APE` for token positions beyond 2 048. This suggests that SPE inherits this celebrated advantage of RPE (Huang et al., 2018) while being applicable to much longer sequences.

Recently, Wang et al. (2021) defined metrics for the evaluation of PE, suggesting that *translation invariance* and *monotonicity* are desirable properties. The former states that the distances of two arbitrary τ-offset position embeddings should be identical, while the latter states that neighboring

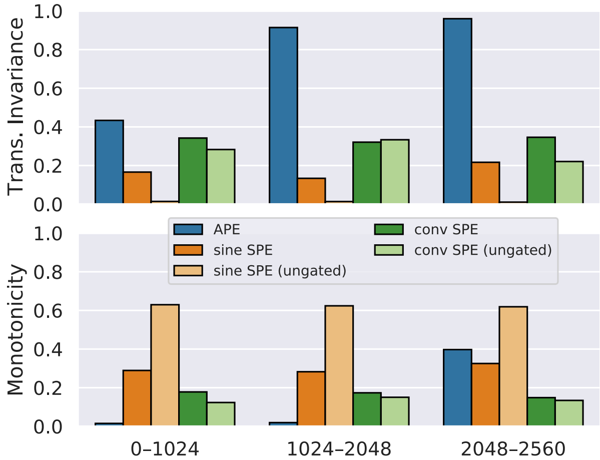

Figure 4. PE evaluation metrics (Wang et al., 2021) for the pop piano music generation task in the 1st layer (lower is better), w.r.t. query positions. Training sequence length is 2 048. Only query-key offsets <128 are considered here. See appendix for details.

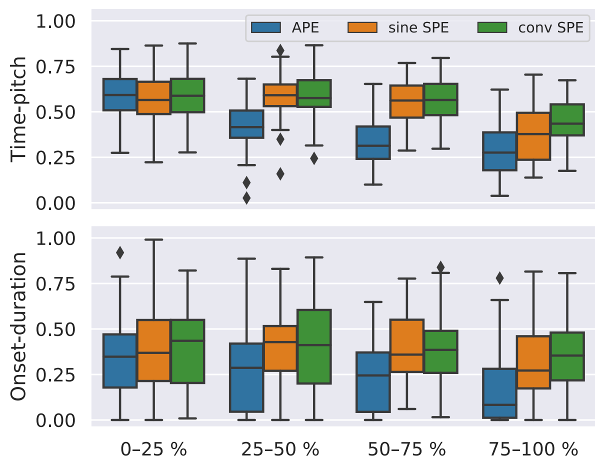

Figure 5. Musical style similarity for groove continuation (higher is better) between output and initial prompt through two musically-motivated metrics, as a function of time in the output. Each data point corresponds to a single musical style.

positions should be assigned with position embeddings that are closer than faraway ones. Following their *identical word probing* methodology, we report these metrics in Figure 4. As expected, SPE variants greatly outperform `APE` in terms of *translation invariance*. However, *monotonicity* does not seem a very relevant criterion in our music application, as can be seen when comparing scores in Figures 3 and 4. It seems that music modeling can benefit from non-vanishing attention patterns. In any case, SPE scores are remarkably stable across positions, contrarily to `APE`, which rapidly degrades beyond the training length.

3.3. Groove Continuation

In this experiment, we evaluate Performers on a *groove continuation* task. After training on a dataset where each example has a uniform style ('groove'), we prime the model with a short *prompt* (2-bar musical fragment) and let it generate a continuation. We then observe whether the generated continuation matches the style of the prompt.

Experimental setup. The models (24-layer Performers with 8 attention heads) are trained on an accompaniment dataset comprising 5 522 samples in 2 761 different musical styles, encoded in a token-based format adopted from Cífka et al. (2020) and detailed in the appendix. All SPE-based models use gating in this experiment. Unlike the previous experiment, which leverages long training sequences, we consider training sequences of length $N = 512$, corresponding to 2–10 bars. At test time, the model is prompted with 2 bars in a style not seen during training and new tokens are sampled to complete the sequence to a length of 1 024, i.e. twice the training length.

We use two musically motivated *style similarity* metrics – *time-pitch* and *onset-duration* proposed by Cífka et al. (2019; 2020) – to quantify the similarity of the generated continuation to the prompt. When listening to the generated music, we perceptually notice a drift in quality along time. For this reason, we divide each generated sample into four successive chunks of identical duration and evaluate them independently. The results are displayed in Figure 5.

Discussion. We clearly see that SPE substantially outperforms `APE` in both metrics. Although `APE` visibly does manage to generate close to the desired style at the beginning of the sequence, this similarity strongly degrades over time. Both `sineSPE` and `convSPE` are much more stable in this regard, confirming the result from section 3.2 that SPE extrapolates better beyond the training sequence length. This matches our informal perceptual evaluation.[6]

This experiment suggests that exploiting a local neighborhood is a robust way to process long sequences. This could appear as contradicting the use of long-range Transformers, but we highlight that *gating* is used here, enabling some heads to exploit long term-attention independently from position. Further comparisons with local attention schemes (e.g. Dai et al., 2019; Hofstätter et al., 2020) could be interesting, although they were not included here due to Tay et al. (2021) suggesting that they are clearly inferior, at least in the LRA setting.

4. Related Work

This paper is concerned with PE (Sukhbaatar et al., 2015), as a way to embed the position of each token as part of its features. This idea is a core ingredient for many subsequent groundbreaking studies (Gehring et al., 2017; Vaswani et al., 2017), and has been the actual topic of many investigations.

[6]Examples: `https://cifkao.github.io/spe/`

Absolute Positional Encoding (APE) based on sinusoids from Vaswani et al. (2017) is the most widely used for Transformer-like architectures. However, PE $\overline{\mathbf{q}}(n)$ and $\overline{\mathbf{k}}(n)$ in (5) can also be trained as in BERT (Devlin et al., 2019; Liu et al., 2019). Although the original Transformer only includes PE at the input layer, it may be included at all layers (Dehghani et al., 2019; Lan et al., 2020).

Relative positional encoding (RPE; Shaw et al., 2018**)** is a way to leverage *relative positions*. It came with a $\mathcal{O}(N^2D)$ space complexity, which was reduced to $\mathcal{O}(N^2)$ in Huang et al. (2018); He et al. (2020). Considering log-distances was proposed in Raffel et al. (2020). Several variants for RPE were introduced (Huang et al., 2020; Wang et al., 2021). They all apply learned RPE in the attention domain. Using fixed embedding functions was also considered for RPE (Pham et al., 2020), and masking RPE is used in Kim et al. (2020) to promote local attention.

Keys-domain vs attention domain. Doing PE in the keys domain introduces position-content cross terms that are advocated as noisy and not beneficial in Ke et al. (2020) and replaced by *Untied* attention, i.e. PE in the attention domain. This is also called *disantangled attention* in He et al. (2020) and already proposed in Tsai et al. (2019) through *separable* content-position attention kernels. All of these studies require the explicit computation and storage of $\mathbf{A}$.

Non-integer positions were considered for structured inputs. Tree-based PE was proposed both for APE (Shiv & Quirk, 2019; Xiao et al., 2019; Ma et al., 2019) and RPE (Omote et al., 2019). Positional encoding of robots within arbitrary polygons is found in Bose et al. (2019).

Dynamical models for PE. Attention for machine translation was introduced in Bahdanau et al. (2016), which was retrospectively understood in Ke et al. (2020) as using recurrent neural nets (RNN) for PE. In Chen et al. (2018), the hidden states of encoder RNNs are said to contain enough position information to skip explicit PE. Neishi & Yoshinaga (2019) builds on this view, but explicitly describes the idea for the first time. Their contribution is to replace the additive PE in (5) by an RNN. In the same vein, Liu et al. (2020) generates PE using (neural) ordinary differential equations.

Convolutional contexts. Our `convSPE` variant involves convolving random noise. First, this can be related to Mohamed et al. (2019), who use convolutional neural networks for queries and keys computation. Second, the connections between convolutions and stationary processes have recently been highlighted by Xu et al. (2020) as enforcing PE.

Multiplicative PE. Various levels of content-position interactions are formalized in (Tsai et al., 2019). Multiplicative strategies were proposed for both RPE (Huang et al., 2020) and APE (Dai et al., 2019). The latter was generalized in Tsai et al. (2019). All these require the explicit computation of the attention matrix. Wang et al. (2020a) presents a scheme that is close to our sinusoidal variant, but without the stochastic part that is the key to go from (14) to (15).

The limits of APE and RPE were highlighted by some authors. In Wang & Chen (2020), the best performing models exploit both absolute *and* relative positions. In Irie et al. (2019) and Tsai et al. (2019), it is found that removing APE altogether in the causal decoder part of Transformer-based architectures leads to comparable/better performance. It is also not clear which one is best between incorporating PE in the raw input signal (and hence propagating it through the *value* entries) or using it anew on the queries and keys only, as we do. Our choice is backed by Tsai et al. (2019).

5. Conclusion

We propose a new Stochastic Positional Encoding (SPE), based on filtering random noise. As we show, the procedure generalizes relative PE and is a principled means to enforce any prescribed (but trained) cross-covariance structure, which we demonstrated should be the central concern in dot-product attention. In our experiments, we show that SPE brings an interesting gain in performance for large-scale transformer models (Choromanski et al., 2020; Katharopoulos et al., 2020), as compared to classical (sinusoidal) PE. This was expected, because RPE (Shaw et al., 2018) is often advocated as beneficial. However, no way to incorporate it for long sequences was available so far and this is the core contribution of this paper. The natural future directions for our study are (i) *Signal-dependent PE* that incorporates the input sequence as an additional input for SPE, (ii) *Nonstationary PE* that utilizes both relative and absolute positions, (iii) Extending our approach to *arbitrary attention kernels*, e.g. defined implicitly through their (random) mappings as in (4). Indeed, SPE as it is presented here holds theoretically for dot-product attention kernels only, but our results given in Table 1 suggest that this generalizes, asking an interesting research question.

Acknowledgements

This work was supported by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 765068 (MIP-Frontiers) and in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute).

We would like to thank Yi Tay, Mostafa Dehghani and Philip Pham for their help with troubleshooting the Long-Range Arena, and Krzysztof Choromanski for clarifications about the Performer.

References

AlQuraishi, M. AlphaFold at CASP13. *Bioinformatics*, 35 (22):4862–4865, 2019.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv:1409.0473 [cs, stat]*, May 2016. URL http://arxiv.org/abs/1409.0473. arXiv: 1409.0473.

Bello, I. LambdaNetworks: Modeling long-range interactions without attention. In *Proc. Int. Conf. Learning Representations*, 2020.

Böck, S., Korzeniowski, F., Schlüter, J., Krebs, F., and Widmer, G. Madmom: A new Python audio and music signal processing library. In *Proc. ACM International Multimedia Conf.*, pp. 1174–1178, 2016.

Bose, K., Adhikary, R., Kundu, M. K., and Sau, B. Positional encoding by robots with non-rigid movements. *arXiv:1905.09786 [cs]*, May 2019. URL http://arxiv.org/abs/1905.09786. arXiv: 1905.09786.

Chen, M. X., Firat, O., Bapna, A., Johnson, M., Macherey, W., Foster, G., Jones, L., Parmar, N., Schuster, M., Chen, Z., Wu, Y., and Hughes, M. The best of both worlds: Combining recent advances in neural machine translation. *arXiv:1804.09849 [cs]*, April 2018. URL http://arxiv.org/abs/1804.09849. arXiv: 1804.09849.

Child, R., Gray, S., Radford, A., and Sutskever, I. Generating long sequences with sparse Transformers. *arXiv:1904.10509 [cs, stat]*, April 2019. URL http://arxiv.org/abs/1904.10509. arXiv: 1904.10509.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L., and Weller, A. Rethinking attention with Performers. *arXiv:2009.14794 [cs, stat]*, September 2020. URL http://arxiv.org/abs/2009.14794. arXiv: 2009.14794.

Cífka, O., Şimşekli, U., and Richard, G. Supervised symbolic music style translation using synthetic data. In *Proc. International Society for Music Information Retrieval Conf.*, pp. 588–595, 2019. doi: 10.5281/zenodo.3527878. URL https://doi.org/10.5281/zenodo.3527878.

Cífka, O., Şimşekli, U., and Richard, G. Groove2Groove: One-shot music style transfer with supervision from synthetic data. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 28:2638–2650, 2020. doi: 10.1109/TASLP.2020.3019642. URL https://hal.archives-ouvertes.fr/hal-02923548.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. Transformer-XL: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Dehghani, M., Gouws, S., Vinyals, O., Uszkoreit, J., and Kaiser, L. Universal Transformers. *arXiv:1807.03819 [cs, stat]*, March 2019. URL http://arxiv.org/abs/1807.03819. arXiv: 1807.03819.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional Transformers for language understanding. *arXiv:1810.04805 [cs]*, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv: 1810.04805.

Donahue, C., Mao, H. H., Li, Y. E., Cottrell, G. W., and McAuley, J. Lakhnes: Improving multi-instrumental music generation with cross-domain pre-training. In *ISMIR*, 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Engel, J., Hantrakul, L., Gu, C., and Roberts, A. Ddsp: Differentiable digital signal processing. *arXiv preprint arXiv:2001.04643*, 2020.

Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. Convolutional sequence to sequence learning. *arXiv:1705.03122 [cs]*, July 2017. URL http://arxiv.org/abs/1705.03122. arXiv: 1705.03122.

Genton, M. G. and Kleiber, W. Cross-covariance functions for multivariate geostatistics. *Statistical Science*, pp. 147–163, 2015.

Hawthorne, C., Elsen, E., Song, J., Roberts, A., Simon, I., Raffel, C., Engel, J., Oore, S., and Eck, D. Onsets and Frames: Dual-objective piano transcription. In *Proc. Int. Society for Music Information Retrieval Conf.*, 2018.

He, P., Liu, X., Gao, J., and Chen, W. DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv:2006.03654 [cs]*, June 2020. URL http://arxiv.org/abs/2006.03654. arXiv: 2006.03654.

Hofstätter, S., Zamani, H., Mitra, B., Craswell, N., and Hanbury, A. Local self-attention over long text for efficient document retrieval. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2021–2024, 2020.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. The curious case of neural text degeneration. In *Proc. International Conference on Learning Representations*, 2019.

Hsiao, W.-Y., Liu, J.-Y., Yeh, Y.-C., and Yang, Y.-H. Compound Word Transformer: Learning to compose full-song music over dynamic directed hypergraphs. In *Proc. AAAI Conf. Artificial Intelligence*, 2021.

Huang, C.-Z. A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A. M., Hoffman, M. D., Dinculescu, M., and Eck, D. Music Transformer. *arXiv:1809.04281 [cs, eess, stat]*, December 2018. URL http://arxiv.org/abs/1809.04281. arXiv: 1809.04281.

Huang, Y.-S. and Yang, Y.-H. Pop Music Transformer: Generating music with rhythm and harmony. In *Proc. ACM International Multimedia Conf.*, 2020.

Huang, Z., Liang, D., Xu, P., and Xiang, B. Improve Transformer models with better relative position embeddings. *arXiv preprint arXiv:2009.13658*, 2020.

Irie, K., Zeyer, A., Schlüter, R., and Ney, H. Language modeling with deep Transformers. *Proc. Interspeech*, pp. 3905–3909, 2019. doi: 10.21437/Interspeech.2019-2225. URL http://arxiv.org/abs/1905.04226. arXiv: 1905.04226.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast autoregressive Transformers with linear attention. In *Proc. Int. Conf. Machine Learning*, pp. 5156–5165, 2020.

Ke, G., He, D., and Liu, T.-Y. Rethinking positional encoding in language pre-training. *arXiv:2006.15595 [cs]*, July 2020. URL http://arxiv.org/abs/2006.15595.

Kim, J., El-Khamy, M., and Lee, J. T-GSA: Transformer with Gaussian-weighted self-attention for speech enhancement. *arXiv:1910.06762 [cs, eess]*, February 2020. URL http://arxiv.org/abs/1910.06762. arXiv: 1910.06762.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Kucerovsky, D., Mousavand, K., and Sarraf, A. On some properties of toeplitz matrices. *Cogent Mathematics*, 3(1), 2016. doi: 10.1080/23311835.2016.1154705. URL http://doi.org/10.1080/23311835.2016.1154705.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv:1909.11942 [cs]*, February 2020. URL http://arxiv.org/abs/1909.11942. arXiv: 1909.11942.

Liu, X., Yu, H.-F., Dhillon, I., and Hsieh, C.-J. Learning to encode position for Transformer with continuous dynamical model. *arXiv:2003.09229 [cs, stat]*, March 2020. URL http://arxiv.org/abs/2003.09229. arXiv: 2003.09229.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv:1907.11692 [cs]*, July 2019. URL http://arxiv.org/abs/1907.11692. arXiv: 1907.11692.

Ma, C., Tamura, A., Utiyama, M., Sumita, E., and Zhao, T. Improving neural machine translation with neural syntactic distance. In *Proc. Conf. North American Chapter of the Association for Computational Linguistics*, pp. 2032–2037, 2019. doi: 10.18653/v1/N19-1205. URL http://aclweb.org/anthology/N19-1205.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proc. Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, 2011. URL https://www.aclweb.org/anthology/P11-1015.

Matheron, G. Principles of geostatistics. *Economic geology*, 58(8):1246–1266, 1963.

Mohamed, A., Okhonko, D., and Zettlemoyer, L. Transformers with convolutional context for asr. *arXiv preprint arXiv:1904.11660*, 2019.

Nadaraya, E. A. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

Nangia, N. and Bowman, S. ListOps: A diagnostic dataset for latent tree learning. In *Proc. Conf. North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 92–99, 2018. doi: 10.18653/v1/N18-4013. URL https://www.aclweb.org/anthology/N18-4013.

Neishi, M. and Yoshinaga, N. On the relation between position information and sentence length in neural machine translation. In *Proc. Conf. Computational Natural Language Learning*, pp. 328–338, 2019. doi: 10.18653/v1/K19-1031. URL https://www.aclweb.org/anthology/K19-1031.

Omote, Y., Tamura, A., and Ninomiya, T. Dependency-based relative positional encoding for Transformer NMT. In *Proc. Natural Language Processing in a Deep Learning World*, pp. 854–861, 2019. ISBN 978-954-452-056-4. doi: 10.26615/978-954-452-056-4_099. URL https://acl-bg.org/proceedings/2019/RANLP2019/pdf/RANLP099.pdf.

Pham, N.-Q., Ha, T.-L., Nguyen, T.-N., Nguyen, T.-S., Salesky, E., Stueker, S., Niehues, J., and Waibel, A. Relative positional encoding for speech recognition and direct translation. *arXiv:2005.09940 [cs, eess]*, May 2020. URL http://arxiv.org/abs/2005.09940.

Radev, D. R., Muthukrishnan, P., Qazvinian, V., and Abu-Jbara, A. The ACL anthology network corpus. *Language Resources and Evaluation*, 47(4):919–944, January 2013. doi: 10.1007/s10579-012-9211-2. URL https://doi.org/10.1007/s10579-012-9211-2.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text Transformer. *arXiv:1910.10683 [cs, stat]*, July 2020. URL http://arxiv.org/abs/1910.10683. arXiv: 1910.10683.

Rosendahl, J., Tran, V. A. K., Wang, W., and Ney, H. Analysis of positional encodings for neural machine translation. In *Proc. IWSLT*, 2019.

Roy, A., Saffar, M., Vaswani, A., and Grangier, D. Efficient content-based sparse attention with routing Transformers. *arXiv:2003.05997 [cs, eess, stat]*, October 2020. URL http://arxiv.org/abs/2003.05997. arXiv: 2003.05997.

Shaw, P., Uszkoreit, J., and Vaswani, A. Self-attention with relative position representations. *arXiv:1803.02155 [cs]*, April 2018. URL http://arxiv.org/abs/1803.02155. arXiv: 1803.02155.

Shen, Z., Zhang, M., Zhao, H., Yi, S., and Li, H. Efficient attention: Attention with linear complexities. *arXiv:1812.01243 [cs]*, November 2020. URL http://arxiv.org/abs/1812.01243. arXiv: 1812.01243.

Shiv, V. and Quirk, C. Novel positional encodings to enable tree-based Transformers. In *Proc. Advances in neural information processing systems*, 2019.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. End-to-end memory networks. *arXiv:1503.08895 [cs]*, November 2015. URL http://arxiv.org/abs/1503.08895. arXiv: 1503.08895.

Tay, Y., Dehghani, M., Abnar, S., Shen, Y., Bahri, D., Pham, P., Rao, J., Yang, L., Ruder, S., and Metzler, D. Long Range Arena: A benchmark for efficient Transformers. In *Proc. Int. Conf. Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. Transformer dissection: An unified understanding for Transformer's attention via the lens of kernel. In *Proc. Conf. Empirical Methods in Natural Language Processing and Int. Joint Conf. Natural Language Processing*, pp. 4335–4344, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Proc. Advances in neural information processing systems*, pp. 5998–6008, 2017.

Vetterli, M., Kovačević, J., and Goyal, V. K. *Foundations of signal processing*. Cambridge University Press, 2014.

Vořechovský, M. Simulation of simply cross correlated random fields by series expansion methods. *Structural safety*, 30(4):337–363, 2008.

Wang, B., Zhao, D., Lioma, C., Li, Q., Zhang, P., and Simonsen, J. G. Encoding word order in complex embeddings. *arXiv:1912.12333 [cs]*, June 2020a. URL http://arxiv.org/abs/1912.12333. arXiv: 1912.12333.

Wang, B., Shang, L., Lioma, C., Jiang, X., Yang, H., Liu, Q., and Simonsen, J. G. On position embeddings in BERT. In *Proc. Int. Conf. Learning Representations*, 2021. URL https://openreview.net/forum?id=onxoVA9FxMw.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity. *arXiv:2006.04768 [cs, stat]*, June 2020b. URL http://arxiv.org/abs/2006.04768. arXiv: 2006.04768.

Wang, Y.-A. and Chen, Y.-N. What do position embeddings learn? An empirical study of pre-trained language model positional encoding. In *Proc. Conf. Empirical Methods in Natural Language Processing*, 2020.

Watson, G. S. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 359–372, 1964.

Williams, C. K. and Rasmussen, C. E. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Williams, R. J. and Zipser, D. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Wu, Q., Lan, Z., Gu, J., and Yu, Z. Memformer: The memory-augmented Transformer. *arXiv:2010.06891 [cs]*, October 2020. URL http://arxiv.org/abs/2010.06891. arXiv: 2010.06891.

Xiao, F., Li, J., Zhao, H., Wang, R., and Chen, K. Lattice-Based Transformer encoder for neural machine translation. *arXiv:1906.01282 [cs]*, June 2019. URL http://arxiv.org/abs/1906.01282. arXiv: 1906.01282.

Xu, R., Wang, X., Chen, K., Zhou, B., and Loy, C. C. Positional encoding as spatial inductive bias in gans. *arXiv preprint arXiv:2012.05217*, 2020.

Xue, H. and Salim, F. D. Trailer: Transformer-based time-wise long term relation modeling for citywide traffic flow prediction. *arXiv preprint arXiv:2011.05554*, 2020.

Yang, Z., Xie, L., and Stoica, P. Vandermonde decomposition of multilevel Toeplitz matrices with application to multidimensional super-resolution. *IEEE Transactions on Information Theory*, 62(6):3685–3701, 2016.

Zhou, P., Fan, R., Chen, W., and Jia, J. Improving generalization of transformer for speech recognition with parallel schedule sampling and relative positional embedding. *arXiv preprint arXiv:1911.00203*, 2019.