# Fractional Fourier Transform in Time Series Prediction

Emirhan Koç and Aykut Koç, *Senior Member, IEEE*

***Abstract*—Several signal processing tools are integrated into machine learning models for performance and computational cost improvements. Fourier transform (FT) and its variants, which are powerful tools for spectral analysis, are employed in the prediction of univariate time series by converting them to sequences in the spectral domain to be processed further by recurrent neural networks (RNNs). This approach increases the prediction performance and reduces training time compared to conventional methods. In this letter, we introduce fractional Fourier transform (FrFT) to time series prediction by RNNs. As a parametric transformation, FrFT allows us to seek and select better-performing transformation domains by providing access to a continuum of domains between time and frequency. This flexibility yields significant improvements in the prediction power of the underlying models without sacrificing computational efficiency. We evaluated our FrFT-based time series prediction approach on synthetic and real-world datasets. Our results show that FrFT gives rise to performance improvements over ordinary FT.**

***Index Terms*—Fourier transform, fractional Fourier transform, time series, recurrent neural networks, encoder, decoder.**

## I. Introduction

NEURAL network-based machine learning methods are ubiquitously used in a wide range of application areas such as time series/sequence prediction, image processing, computer vision, and natural language processing (NLP) [1], [2], [3]. Although deep neural networks (DNNs) ensure increased performances with sophisticated algorithms, they are also open to further improvements through integration with classical signal processing tools and ideas. Recently, there has been a thriving tendency towards deep neural network models relying on more theoretical and analytical foundations. To this end, utilization of several classical signal processing tools, mainly in convolutional neural networks (CNNs) and recurrent neural networks (RNNs), has become prevalent and provided considerable performance improvements [4], [5], [6], [7], [8], [9], [10].

In [6], wavelet scattering transforms are combined with CNNs to address text-independent speaker identification, where features extracted from wavelet scattering are used as the first layer of CNNs, and the remaining layers are learned with supervision. The reported experiments show that the wavelet scattering transform offers efficient feature extraction for speaker identification tasks. In [5], fractional wavelet scattering network (FrScatNet) is used for gland segmentation task on colon histology images, where corresponding features are extracted by fractional wavelet transform and used to train CNNs. [7] constructs a mathematical model for FrScatNet and reaches outperforming results compared to ordinary CNNs in image classification tasks. In [8], Fourier-Bessel series expansion (FBSE)-based empirical wavelet transforms are proposed for better time-frequency representations (TFR) of non-stationary signals. Motivated by [8], a new TFR method based on Fourier-Bessel decomposition for the classification of sleep stages using CNNs is also proposed in [9]. [10] proposes a TFR method using improved eigenvalue decomposition of the Hankel matrix and Hilbert transform (IEVDHM-HT) for the classification of epileptic seizures using least-square support vector machines (SVM).

Fourier Transform (FT) is also deployed in machine learning mostly to reach computational accelerations. In [11], [12], [13], the convolution property of FT is leveraged such that convolution operations in layers of DNNs are replaced with Hadamard point-wise products in FT domain [13]. Since convolutions are computationally costly, the proposed approach enables significant speed-up without considerable degradation in model performance. FT is also deployed in machine learning models for the NLP domain. In [14], replacing costly self-attention sub-layers in the transformer encoders with the FT yields comparable accuracy with remarkable computational efficiency as FT is a linear and non-parametric transform with fast implementations. [15] proposes the introduction of FT into autoencoders to detect anomalies with less noisy features. The proposed model yields competitive performances with the state-of-the-art. FT is also used in RNNs to address vanishing/exploding gradient problems and learn hidden-state information [16].

Time series prediction is a fundamental task in numerous applications from various domains such as weather forecasting, electricity market design, and traffic management [17], [18]. Machine learning methods, especially RNNs, have also gained overwhelming preference against statistical methods in time series prediction [19]. Following the trend of combining signal processing with machine learning, [20] has recently presented a method that combines short-time Fourier transform (STFT) and RNNs in time series prediction. In [20], as a comparative method, complex-gated recurrent units (cgGRU) [21] are utilized to handle back-propagation with complex-valued signals.

Fractional Fourier Transform (FrFT) is the generalized version of the ordinary FT with a real fraction order $a$ [22]. FrFT can be considered a rotation in the time-frequency plane and allows us to observe the representation of signals in a continuum of infinitely many intermediate domains between time and frequency, controlled by a single parameter. Being the direct generalization of FT, FrFT inherits FT's power and

Manuscript received 14 October 2022; revised 3 December 2022; accepted 6 December 2022. Date of publication 9 December 2022; date of current version 22 December 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xuesong Wang. *(Corresponding author: Aykut Koç.)*

The authors are with the Department of Electrical and Electronics Engineering, Bilkent University, 06800 Ankara, Turkey, and also with UMRAM, Bilkent University, 06800 Ankara, Turkey (e-mail: aykut.koc@bilkent.edu.tr).

Source codes are available at https://github.com/koc-lab/FrFTimeSeries

Digital Object Identifier 10.1109/LSP.2022.3228131

provides an additional degree of freedom such that a continuum of alternative transformations is accessible. These alternatives lead to different information flows and feature extractions in neural networks, and the most representative signal domain that works best in the underlying neural network can be chosen. Equally importantly, FrFT also has a well-established $O(N \log N)$ time-efficient computational algorithm [23], making it possible to enjoy its advantages without paying for additional computational costs. Motivated by these advantages, FrFT is very recently introduced to the state-of-the-art deep learning architectures by two pioneering works [24] and [25] for vision and NLP transformers, respectively.

In this work, we introduce FrFT to the machine learning domain by combining it with RNNs for the time series prediction problem with the motivation of utilizing a generalized transform capable of implementing infinitely many transformations to increase model performance. We present a model where FrFT is applied to each segment of windowed time-series signals in the time domain by replacing STFT-based sequence predictions with fractional orders by changing neither the number of computation steps nor the computational costs of these steps. In other words, FrFT is computed in our proposed method the same number of times as the FT is computed in the FT-based time series prediction, and a fast computation algorithm for FrFT ([23]) can also be used. We show that FrFT-based RNN models give better prediction performances than those based on either time and frequency domain. Moreover, we also use a mechanism that incorporates the FrFT order $a$ as a learnable parameter and learns its optimal value by training. This contribution adds an attribute to our method that refrains us from manually selecting the fraction order by incorporating $a$ into the network as a learnable weight similar to the remaining standard weights. We demonstrated the superiority of our approach with several experiments conducted on domain-general time series data from a wide range of areas, such as nonlinear differential equations, finance, and electric energy consumption.

## II. Primer on Fractional Fourier Transform

For $a \in \mathbb{R}$, the $a$th order FrFT $\mathcal{F}^a$ of a function or signal $f(t) \in \mathcal{L}^2(\mathbb{R})$ is defined as follows [22]:

$$\mathcal{F}_a(u) = \mathcal{F}^a\{f(t)\}(u) = \int_{-\infty}^{\infty} K_a(u,t) f(t) dt,$$

$$K_a(u,t) = A_\phi e^{i\pi(u^2 \cot\phi - 2utcsc\phi + t^2 \cot\phi)},$$

$$A_\phi = \sqrt{1 - i\cot\phi}, \;\; \phi = a\pi/2,$$

$$K_a(u,t) = \begin{cases} A_\phi e^{i\pi(u^2 \cot\phi - 2utcsc\phi + t^2 \cot\phi)} & \text{if } a \neq 2k \\ \delta(u-t) & \text{if } a = 4k-2 \\ \delta(u+t) & \text{if } a = 4k+2, \end{cases}$$

where $k$ is an integer. FrFT operates with period 4 such that $\mathcal{F}^a = \mathcal{F}^b$ where $a \equiv b \mod 4$. In other words, $\mathcal{F}^a = \mathcal{F}^{4k+a}$ where $a$ can be evaluated in the intervals $[-2, 2]$ or [0, 4]. The inverse FrFT for a certain fraction is $(\mathcal{F}^a)^{-1} = \mathcal{F}^{-a}$. Further information on FrFT can be found in [22], [26], [27].

One needs to emphasize that the discrete version of FrFT is also established [28], [29], and it is readily possible to represent discrete-time signals in fractional Fourier domains. Similar to the discrete Fourier transform (DFT), which can be computed via a matrix-vector multiplication, the discrete fractional Fourier transform (DFrFT) can also be expressed as matrix-vector multiplication [28]. Similarly, there are established fast computation algorithms [23] for FrFT that have the same order of computational cost as the famous fast Fourier transform (FFT).

Let $\mathcal{X}_n = \{x_0, x_1, \ldots ..., x_{L-1}\}$ is a sequence of length $L$ and the formal definition for DFrFT of $\mathcal{X}_n$ is given as a matrix multiplication:

$$X_{F^a} = W^a \mathcal{X}_n, \tag{1}$$

where $\mathcal{X}_n$ is manifested as a column vector, and $W^a$ is the $L \times L$ DFrFT matrix with order $a$ as given below:

$$W^a[m,n] = \sum_{k=0, k\neq(L-1+(L)_2)}^{L} u_k[m] e^{-i\frac{\pi}{2}ka} u_k[n], \tag{2}$$

where $(L)_2 \equiv L \mod 2$ and $u_k$ is the $k$th discrete Hermite-Gaussian function [28]. Further implementation details and derivations can be found in [23], [28], [30].

## III. Time Series Prediction Using Fractional Fourier Representations

Our proposed method consists of two major stages. The first stage (feature extraction) comprises the series of cascaded operations that generate feature vectors from sequential time series data. Similar to the STFT [31], [32], any univariate data sequence $\mathcal{X}_n = \{x_1, x_2, x_3, \ldots, x_L\}$ of length $L$ is multiplied with a sliding window $w_n$ of length $\tau$ and split into $M$ segments. Resulting segments are stacked to create $\mathbb{X}_W \in \mathbb{R}^{M\times\tau}$. Subsequently, we apply FrFT to each row vector of $\mathbb{X}_W$ by multiplying its transpose with DFrFT matrix of order $a$, and $\mathbb{X}_{F^a} \in \mathbb{C}^{\tau\times M}$ is obtained. Finally, $\{\mathbf{x}_F^m\}_{m=1}^M$, which are the columns of $\mathbb{X}_{F^a}$, are extracted as features as the following:

$$X_n[m] = w_n[n - Sm]\mathcal{X}_n[n], \tag{3}$$

$$\mathbb{X}_W = \mathbb{W}(\mathcal{X}_n) \in \mathbb{R}^{M\times\tau}, \tag{4}$$

$$\mathbb{X}_{F^a} = W^a \mathbb{X}_W^\intercal, \tag{5}$$

$$\mathbb{X}_{F^a} \in \mathbb{C}^{\tau\times M} \mapsto \{\mathbf{x}_F^m\}_{m=1}^M \in \mathbb{C}^\tau, \tag{6}$$

where $\mathbb{W}$ is a mapping from a sequence to matrix of where each row is a windowed segment $X_n[m]$ and $W^a \in \mathbb{C}^{\tau\times\tau}$ is a DFrFT matrix of $a$.

The second stage (encoder-decoder) of our proposed method comprises a many-to-many encoder-decoder architecture with Gated Recurrent Units (GRU) or basic RNNs cells [20], [33], yielding two proposed variants. The features produced by the first stage are fed into the encoder block of the second stage, while the decoder part processes only the original time-domain data. Our overall model is illustrated in Fig. 1.

The model is then trained using the training set of the sequential information that is being studied. The feature extraction procedure is applied to only encoder training sequences, and the features $\{\mathbf{x}_F^i\}_{i=1}^M$ are generated. These features are then fed to the encoder block GRU (or basic RNNs) units sequentially such that $\mathbf{x}_F^1$ goes to the first encoder unit, and $\mathbf{x}_F^n$ goes to the $n$th one. In the training procedure's forward pass, the encoder's hidden state is initialized to zero, and weights are initialized with Xavier initializer [34]. In each cell along the encoder, a new hidden state is produced using the current input and previous state and passed to the next cell. The state information of the last cell is transferred to the decoder to initialize the decoder's hidden states. Similarly, hidden state information is produced and propagated as in the encoder. Along the decoder, the output vector of each cell is calculated as the multiplication of hidden

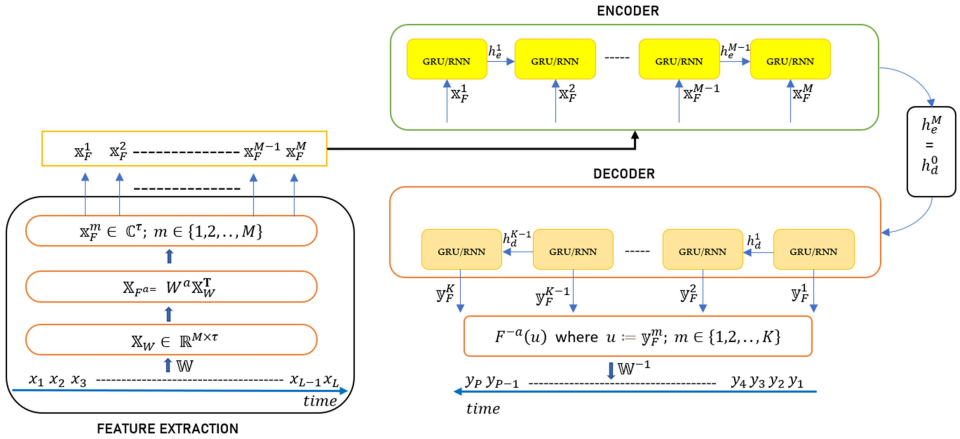


Fig. 1. Feature vectors are extracted from the univariate time series using 3 cascaded operations. The proposed feature extraction method is applied to a sequence of encoder blocks and fed into GRU/RNN cells. The prediction sequence of length $P$ is generated at the decoder block.

state vector $\boldsymbol{h}_d$ with output weights $W_{hy}$, and passed through $\tanh$ activation function. The output vectors are multiplied with DFrFT of order $-a$ to calculate the inverse FrFT, then converted to one-dimensional sequences as in the case with the inverse STFT [31], [32]. The resulting sequence is finally used to perform the prediction. It is also necessary to mention that the length of the decoder block can differ from that of the encoder.

In the backward pass, the mean-squared error (MSE) between the predicted and the decoder training sequences is calculated, and weights are updated via back-propagation using RMSProp optimizer [35]. After the training phase, model parameters and weights are stored for the inference phase with unseen test data.

Finally, the stored parameters and weights are used to initialize the test model in the inference phase. Features extracted from encoder test sequences are fed to the encoder network, and the last hidden state is transferred to the decoder. In each cell along the decoder, output vectors are calculated to generate a prediction sequence for decoder test data.

## IV. Experiments & Results

We conducted experiments on one synthetic dataset and two real-world datasets to demonstrate our proposed approach.

### A. Datasets

*1) Mackey-Glass Chaotic Time Series:* It is generated from a nonlinear, time delay differential system that is described by the following differential equation [36]:

$$\frac{dx}{dt} = \frac{\beta x(t-T)}{1+x^{10}(t-T)} - \gamma x(t), \tag{7}$$

where $\beta = 0.2$, $\gamma = 0.1$, $dt = 0.1$, and $T = 17$. Starting values up to $T$-th second are initialized as $1 + u[-0.1, +0.1]$ where $u$ stands for the uniform distribution.

*2) TRY Currency Exchange Ratio:* Currency exchange ratio data between USD and Turkish lira (TRY) from Jan. 1, 2007 to Jan. 1, 2020 are used [37]. In recent years, the USDTRY exchange ratio has shown rapid fluctuations, which makes this data challenging to predict the series of future values based on past information. We filled in missing days data in this dataset with the average data from the previous and following days.

*3) University of California - Irvine (UCI) Electricity Load Dataset:* UCI dataset in [38] contains the electricity consumption of 370 customers from Jan. 1, 2011, to Jan. 1, 2015, in 15 min. resolution, and values are in kW. Due to many missing values, data before 2011 is not utilized. Each value is divided by 4 (since 15 minutes is one-fourth of an hour) to convert energy consumption into kWh. Daily total energy consumption is then used instead of 15 minutes resolution. Data is also normalized using min-max normalization for each customer.

**Algorithm 1:** Learning Sequential Information Using FRFT Representations. $\odot$ Stands for Hadamard Point-Wise Multiplication.

**Input:** Encoder feature vectors: $\left\{\mathbf{x}_F^i\right\}_{i=1}^{E}$
**Output:** Model weights: $\boldsymbol{W}_{final}$
**Parameters:** $E$ and $D$ (encoder-decoder sequence lengths), $T_{iter}$ (iteration number)
**Initialize:** $\boldsymbol{W}_0$, $\boldsymbol{h}_e^0 = 0$
**for** $t = 1$ to $T_{iter}$ **do**
  **Forward Pass**:
  **for** $i = 1$ to $E$ **do**
    **if** GRU **then**
      $z_e^i = \sigma_g(W_{zx}\mathbf{x}_F^i + W_{hz}\boldsymbol{h}_e^{i-1})$
      $r_e^i = \sigma_g(W_{rx}\mathbf{x}_F^i + W_{hr}\boldsymbol{h}_e^{i-1})$
      $\hat{h}_e^i = \tanh(W_{xx}\mathbf{x}_F^i + W_{hh}(r_e^i \odot \boldsymbol{h}_e^{i-1}))$
      $\boldsymbol{h}_e^i = z_e^i \odot \hat{h}_e^i + (1 - z_e^i) \odot \boldsymbol{h}_e^{i-1}$
    **else if** RNN **then**
      $\boldsymbol{h}_e^i = \sigma_g(W_{xx}\mathbf{x}_F^i + W_{hh}\boldsymbol{h}_e^{i-1})$
    **end if**
  **end for**
  $\boldsymbol{h}_d^0 \leftarrow \boldsymbol{h}_e^E$
  **for** $j = 1$ to $D$ **do**
    **if** GRU **then**
      $z_d^j = \sigma_g(W_{hz}\boldsymbol{h}_d^{j-1})$
      $r_d^j = \sigma_g(W_{hr}\boldsymbol{h}_d^{j-1})$
      $\hat{h}_d^j = \tanh(W_{hh}(r_d^j \odot \boldsymbol{h}_d^{j-1}))$
      $\boldsymbol{h}_d^j = z_d^j \odot \hat{h}_d^j + (1 - z_e^j) \odot \boldsymbol{h}_d^{j-1}$
    **else if** RNN **then**
      $\boldsymbol{h}_d^j = \sigma_g(W_{hh}\boldsymbol{h}_d^{j-1})$
    **end if**
    $\hat{\mathbf{y}}_F^j = \tanh(W_{hy}\boldsymbol{h}_d^j)$
  **end for**
  $\left\{\hat{\mathbf{y}}_p\right\}_{p=1}^{P} \leftarrow \left\{\hat{\mathbf{y}}_F^j\right\}_{j=1}^{D}$ (inverse fractional Fourier Transform and sequence reconstruction)
  **Backward Pass:**
  $\mathcal{L}_{MSE}(\hat{y}, y) = \frac{1}{P}\sum_{i=1}^{P}(\hat{y}_i - y_i)^2$
  Update weights: $\boldsymbol{W}_t \leftarrow \boldsymbol{W}_{t-1}$
**end for**
**return** $\boldsymbol{W}_{final}$

TABLE I
MACKEY-GLASS FOR VARIOUS ARCHITECTURES AND ORDERS $a$

| $a$ | GRU64 | RNN64 | GRU64$_{LP_{16}}$ | RNN64$_{LP_{16}}$ | cgGRU64 | cgGRU32 |
|---|---|---|---|---|---|---|
| 0.5 | 0.030 | **0.022** | 71.84 | 71.63 | **0.005** | **0.022** |
| 0.6 | **0.021** | 0.034 | 64.14 | 64.20 | **0.010** | **0.020** |
| 0.7 | 0.026 | **0.026** | 47.14 | 47.55 | **0.011** | **0.032** |
| 0.8 | **0.020** | **0.017** | 2.256 | 3.614 | **0.006** | **0.054** |
| 0.9 | 0.024 | **0.013** | **0.039** | 0.027 | **0.010** | **0.039** |
| 1.0 | 0.023 | 0.034 | 0.044 | 0.012 | 0.015 | 0.175 |
| 1.1 | 0.027 | **0.031** | **0.040** | 0.041 | **0.004** | **0.078** |
| 1.2 | 0.033 | **0.023** | 0.942 | 2.556 | **0.011** | **0.029** |
| 1.3 | **0.018** | **0.019** | 46.85 | 46.91 | **0.009** | **0.013** |
| 1.4 | 0.024 | **0.016** | 64.13 | 64.19 | **0.009** | **0.014** |
| 1.5 | **0.016** | **0.017** | 71.71 | 71.81 | **0.013** | **0.035** |

TABLE II
TRY/USD FOR VARIOUS ARCHITECTURES AND ORDERS $a$

| | Model | | | | | |
|---|---|---|---|---|---|---|
| $a$ | GRU64 | RNN64 | GRU64$_{LP_{16}}$ | RNN64$_{LP_{16}}$ | cgGRU64 | cgGRU32 |
| 0.5 | **9.33** | 9.78 | 63.22 | 63.31 | 9.07 | **8.81** |
| 0.6 | **9.72** | 9.32 | 49.01 | 48.95 | 8.95 | **7.65** |
| 0.7 | **9.61** | 10.17 | 43.43 | 43.05 | 9.03 | **8.48** |
| 0.8 | 9.84 | 9.28 | **10.08** | **10.40** | 8.84 | **8.49** |
| 0.9 | 9.99 | 8.72 | **9.91** | **10.39** | 9.23 | **9.10** |
| 1.0 | 9.74 | 8.15 | 10.74 | 10.92 | 8.57 | 9.73 |
| 1.1 | **9.54** | 9.03 | **10.08** | **10.39** | 9.33 | **8.94** |
| 1.2 | **9.43** | 9.89 | 10.83 | 10.92 | 8.84 | **8.48** |
| 1.3 | **9.70** | 10.44 | 44.55 | 44.58 | **8.06** | **8.15** |
| 1.4 | 10.13 | 10.05 | 49.08 | 48.89 | 9.38 | **8.23** |
| 1.5 | **9.05** | 10.17 | 62.70 | 62.96 | 9.22 | **8.68** |

TABLE III
ELECTRIC CONSUMPTION FOR VARIOUS ARCHITECTURES AND ORDERS $a$

| $a$ | GRU64 | RNN64 | GRU64$_{LP_{16}}$ | RNN64$_{LP_{16}}$ | cgGRU64 | cgGRU32 |
|---|---|---|---|---|---|---|
| 0.5 | 28.54 | 11.98 | 79.52 | 71.43 | 17.25 | 22.30 |
| 0.6 | 21.26 | 10.92 | 59.73 | 55.93 | 19.39 | **20.30** |
| 0.7 | 20.62 | 11.58 | 56.78 | 47.2 | **16.27** | **19.78** |
| 0.8 | 26.65 | **10.34** | 12.98 | 14.33 | **15.14** | 21.71 |
| 0.9 | 19.95 | **10.36** | 12.47 | **7.51** | 18.41 | **19.00** |
| 1.0 | 19.00 | 10.54 | 10.12 | 8.38 | 16.98 | 20.63 |
| 1.1 | **17.24** | 11.52 | 12.94 | **8.03** | **16.86** | **20.62** |
| 1.2 | **18.58** | **10.40** | 11.89 | 8.65 | 17.18 | **19.70** |
| 1.3 | 25.38 | 10.64 | 53.55 | 47.26 | 18.68 | 25.65 |
| 1.4 | **18.82** | **10.42** | 61.86 | 56.41 | 19.72 | **18.38** |
| 1.5 | 21.54 | 12.19 | 85.77 | 71.43 | 18.41 | **20.31** |

TABLE IV
BEST MODEL PERFORMANCES OF THE PROPOSED METHOD ARE COMPARED TO BASELINES THAT USE GRU/RNN

| **Dataset** | **Model** | $NMSPE(\%)$ |
|---|---|---|
| Mackey-Glass | RNN64 ($a_l$=0.944) | **0.0035** |
| | cgGRU64 ($a$=1.1) | 0.004 |
| | GRU64$_{baseline}$ | 0.23 |
| | RNN64$_{baseline}$ | 5.48 |
| TRY-USD | cgGRU32 ($a_l$=1.277) | **6.59** |
| | cgGRU64 ($a$=1.3) | 8.06 |
| | GRU64$_{baseline}$ | 10.30 |
| | RNN64$_{baseline}$ | 12.01 |
| Electric Consumption | GRU64 ($a_l$=0.858) | **6.20** |
| | RNN64$_{LP_{16}}$ ($a$=0.9) | 7.51 |
| | GRU64$_{baseline}$ | 10.84 |
| | RNN64$_{baseline}$ | 14.76 |

### B. Implementation Details

In our experiments, the initial state of the encoder block is set to zero. A learning rate of 0.001 is used in training where the number of iterations for each dataset is 30,000, 10,000, and 30,000, respectively. The learning rate is exponentially decayed by 0.9 in every 1,000 steps. The Gaussian window is used with a length of 64 samples for currency and electric consumption datasets and 128 samples for the Mackey-Glass.

### C. Results

We consider six variants of our proposed FrFT-based time series prediction method in our experiments. Each variant is designed to process features in fractional Fourier domains. GRU64 and RNN64 stand for GRU and RNN architectures with hidden state sizes of 64. Similarly, GRU64$_{LP_{16}}$ and RNN64$_{LP_{16}}$ are GRU and RNN architectures with a hidden state size of 64. However, they are fed with feature vectors filtered using a low-pass filter where the cutoff frequency is $1/16$ of the original signal bandwidth. cgGRU32 and cgGRU64 architectures stand for complex-gated GRUs with hidden sizes of 32 and 64, respectively. Complex-gated GRU [21] is an advanced architecture capable of being trained with complex-valued weights.

We use normalized mean-squared percentage error (NMSPE) as our evaluation metric. It is defined as $100 \times \sum_{i=1}^{N}(y_i - \hat{y}_i)^2 / \sum_{i=1}^{N}(y_i)^2$ and expressed as a percentage, where $y_i$ is the target value and $\hat{y}_i$ is the predicted value. In Tables I–III, the performances of various methods are reported for different FrFT orders and architectural differences for all three datasets. In the tables, results for the special case of the ordinary FT (order $a = 1$) are boxed as baselines, and better-performing results concerning their corresponding baselines are emboldened.

We also perform experiments involving the fraction order $a$ to the parameter learning stage. The fraction order $a$ is considered a learnable weight similar to the remaining learnable network parameters. Instead of manually searching for the optimal value, $a$ is randomly initialized from the uniform distribution $U \sim (0.5, 1.5)$ and learned in the network iteratively to minimize training loss. We denote the learned fraction orders as $a_l$ in order to discriminate them from manually selected fraction orders $a$ that are tabulated in Tables I–III. The performances of the best model with the value of learned fraction order $a_l$ are tabulated in Table IV. We also compare our proposed methods with GRU64$_{baseline}$ and RNN64$_{baseline}$, which are conventional time series prediction methods that do not use Fourier analysis, and the results can be found in Table IV.

In each experiment, the best model performance of the proposed method based on manually-selected fraction order $a$ significantly improves the baseline methods at fraction orders different than $a = 1.0$. Moreover, the proposed method based on learned fraction order $a_l$ both automatizes the parameter search stage and significantly improves the model performance. These promising results show that the FrFT-based time series prediction improves the conventional baseline methods and the ordinary FT-based time series prediction.

## V. CONCLUSION

We introduced the FrFT to the time series prediction problem, which is new to its classical application areas. The parametric nature of FrFT helps us to reach and select from infinitely many signal domains to represent signals for better prediction. We made the fraction order $a$ a learnable parameter, which alleviates and accelerates the process of finding the most representative signal domains. This property can also be extended to create custom pre-trained and fine-tuned models to tune models only for the learnable fractional order as future work.

We demonstrated the proposed method on domain-general signals from a wide range of areas. An immediate future direction is to make an in-depth analysis to determine what kind of signals is more suitable for the FrFT-based approaches in the proposed new application domain of FrFT. FrFT-based time series prediction can open up further future developments where the flexibility of FrFT can be leveraged to improve the performances of several neural network-based architectures.

## REFERENCES


[1] C. Sakr and N. R. Shanbhag, "Signal processing methods to enhance the energy efficiency of in-memory computing architectures," *IEEE Trans. Signal Process.*, vol. 69, pp. 6462–6472, 2021.

[2] L. Cheng, J. Li, and Y. Yan, "FSCNet: Feature-specific convolution neural network for real-time speech enhancement," *IEEE Signal Process. Lett.*, vol. 28, pp. 1958–1962, 2021.

[3] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep learning with long short-term memory for time series prediction," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 114–119, Jun. 2019.

[4] M. Shahbazi and H. Aghajan, "A generalizable model for seizure prediction based on deep learning using CNN-LSTM architecture," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2018, pp. 469–473.

[5] L. Liu, J. Wu, D. Li, L. Senhadji, and H. Shu, "Fractional wavelet scattering network and applications," *IEEE Trans. Biomed. Eng.*, vol. 66, no. 2, pp. 553–563, Feb. 2019.

[6] W. Ghezaiel, B. Luc, and O. Lézoray, "Wavelet scattering transform and CNN for closed set speaker identification," in *Proc. IEEE 22nd Int. Workshop Multimedia Signal Process.,* 2020, pp. 1–6.

[7] J. Shi, Y. Zhao, W. Xiang, V. Monga, X. Liu, and R. Tao, "Deep scattering network with fractional wavelet transform," *IEEE Trans. Signal Process.*, vol. 69, pp. 4740–4757, 2021.

[8] A. Bhattacharyya, L. Singh, and R. B. Pachori, "Fourier–Bessel series expansion based empirical wavelet transform for analysis of non-stationary signals," *Digit. Signal Process.*, vol. 78, pp. 185–196, 2018.

[9] V. Gupta and R. B. Pachori, "FBDM based time-frequency representation for sleep stages classification using EEG signals," *Biomed. Signal Process. Control*, vol. 64, 2021, Art. no. 102265.

[10] R. R. Sharma and R. B. Pachori, "Time–frequency representation using IEVDHM–HT with application to classification of epileptic EEG signals," *IET Sci., Meas. Technol.*, vol. 12, no. 1, pp. 72–82, 2018.

[11] M. Mathieu, M. Henaff, and Y. LeCun, "Fast training of convolutional networks through FFTs: International conference on learning representations," in *Proc. 2nd Int. Conf. Learn. Representations*, Banff, Canada, 2014.

[12] H. Pratt, B. Williams, F. Coenen, and Y. Zheng, "FCNN: Fourier convolutional neural networks," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2017, pp. 786–798.

[13] K. Chitsaz, M. Hajabdollahi, P. Khadivi, S. Samavi, N. Karimi, and S. Shirani, "Use of frequency domain for complexity reduction of convolutional neural networks," in *Proc. Int. Conf. Pattern Recognit.*, 2021, pp. 64–74.

[14] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "FNet: Mixing tokens with Fourier transforms," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2022, pp. 4296–4313.

[15] D. Lappas, V. Argyriou, and D. Makris, "Fourier transformation autoencoders for anomaly detection," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 1475–1479.

[16] J. Zhang, Y. Lin, Z. Song, and I. Dhillon, "Learning long term dependencies via Fourier recurrent units," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5815–5823.

[17] G. Lewenfus, W. A. Martins, S. Chatzinotas, and B. Ottersten, "Joint forecasting and interpolation of time-varying graph signals using deep learning," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 6, pp. 761–773, 2020.

[18] A. Y. Yildiz, E. Koç, and A. Koç, "Multivariate time series imputation with transformers," *IEEE Signal Process. Lett.*, early access, Nov. 25, 2022, doi: 10.1109/LSP.2022.3224880.

[19] A. Cecaj, M. Lippi, M. Mamei, and F. Zambonelli, "Comparing deep learning and statistical methods in forecasting crowd distribution from aggregated mobile phone data," *Appl. Sci.*, vol. 10, no. 18, 2020, Art. no. 6580.

[20] M. Wolter, J. Gall, and A. Yao, "Sequence prediction using spectral RNNs," in *Proc. Int. Conf. Artif. Neural Netw.*, 2020, pp. 825–837.

[21] M. Wolter and A. Yao, "Complex gated recurrent neural networks," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, vol. 31, pp. 10557–10567.

[22] H. M. Ozaktas and M. A. Kutay, "The fractional Fourier transform," in *Proc. IEEE Eur. Control Conf.*, 2001, pp. 1477–1483.

[23] H. M. Ozaktas, O. Arikan, M. A. Kutay, and G. Bozdagi, "Digital computation of the fractional Fourier transform," *IEEE Trans. Signal Process.*, vol. 44, no. 9, pp. 2141–2150, Sep. 1996.

[24] X. Zhao et al., "Fractional Fourier image transformer for multimodal remote sensing data classification," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 15, 2022, doi: 10.1109/TNNLS.2022.3189994.

[25] F. Sahinuç and A. Koç, "Fractional Fourier transform meets transformer encoder," *IEEE Signal Process. Lett.*, vol. 29, pp. 2258–2262, 2022.

[26] R. Tao, Y.-L. Li, and Y. Wang, "Short-time fractional Fourier transform and its applications," *IEEE Trans. Signal Process.*, vol. 58, no. 5, pp. 2568–2580, May 2010.

[27] L. B. Almeida, "The fractional Fourier transform and time-frequency representations," *IEEE Trans. Signal Process.*, vol. 42, no. 11, pp. 3084–3091, Nov. 1994.

[28] C. Candan, M. A. Kutay, and H. M. Ozaktas, "The discrete fractional Fourier transform," *IEEE Trans. Signal Process.*, vol. 48, no. 5, pp. 1329–1337, May 2000.

[29] Y. Liu, H. Miao, F. Zhang, and R. Tao, "Sliding 2D discrete fractional Fourier transform," *IEEE Signal Process. Lett.*, vol. 26, no. 12, pp. 1733–1737, Dec. 2019.

[30] H. M. Ozaktas, Z. Zalevsky, and M. A. Kutay, *The Fractional Fourier Transform With Applications in Optics and Signal Processing*. Hoboken, NJ, USA: Wiley, 2001.

[31] Y. G. Jin, J. W. Shin, and N. S. Kim, "Spectro-temporal filtering for multichannel speech enhancement in short-time Fourier transform domain," *IEEE Signal Process. Lett.*, vol. 21, no. 3, pp. 352–355, Mar. 2014.

[32] Z. Průša and P. Rajmic, "Toward high-quality real-time signal reconstruction from STFT magnitude," *IEEE Signal Process. Lett.*, vol. 24, no. 6, pp. 892–896, Jun. 2017.

[33] K. Cho et al., "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1724–1734.

[34] S. K. Kumar, "On weight initialization in deep neural networks," 2017, *arXiv:1704.08863*.

[35] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Netw. Mach. Learn.*, vol. 4, no. 2, pp. 26–31, 2012.

[36] E. Chng, S. Chen, and B. Mulgrew, "Gradient radial basis function networks for nonlinear and nonstationary time series prediction," *IEEE Trans. Neural Netw.*, vol. 7, no. 1, pp. 190–194, Jan. 1996.

[37] "USD/TRY Exchange Rate," Accessed: Jun. 21, 2022. [Online]. Available: https://finance.yahoo.com/quote/USDTRY=X/

[38] "UCI Machine Learning Respitory: Electricity load diagram data set," Accessed: Jun. 21, 2022. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014