

MACHINE LEARNING WITH PERIODIC DATA

Cross-Reference to Related Application(s)

5 [0001] This application claims the benefit of U.S. Patent Application No. 17/666,076 filed on February 07, 2022, entitled “MACHINE LEARNING WITH PERIODIC DATA”, which is hereby incorporated by reference in its entirety.

Background

10 [0002] Periodic or cyclic data are frequently encountered in a wide range of machine learning scenarios. For example, in recommender systems, it is observed that users may usually log in an application within a relatively fixed time window each day (e.g. before bed or after work), resulting in a strong cyclical pattern in the recommendations to the users. In financial markets, asset prices may rise and fall periodically on a yearly basis, a phenomenon commonly known as “seasonality.” In search engines, the hits of certain keywords can also display periodic patterns. How to exploit the periodicity within training data to learn a better prediction model is thus an important issue for those applications.

Brief Description of the Drawings

20 [0003] Through the following detailed descriptions with reference to the accompanying drawings, the above and other objectives, features and advantages of the example embodiments disclosed herein will become more comprehensible. In the drawings, several example embodiments disclosed herein will be illustrated in an example and in a non-limiting manner, where:

[0004] Fig. 1 illustrates a block diagram of an environment in which the embodiments of the present disclosure can be implemented;

25 [0005] Fig. 2 illustrates a block diagram of a machine learning system with Fourier learning in accordance with some example embodiments of the present disclosure;

[0006] Fig. 3 illustrates a block diagram of a machine learning system with Fourier learning in accordance with some other example embodiments of the present disclosure;

[0007] Fig. 4 illustrates a block diagram of a machine learning system with Fourier learning in accordance with some further example embodiments of the present disclosure;

[0008] Fig. 5 illustrates a diagram of an example algorithm for Fourier learning with pseudo gradient descent in accordance with some embodiments of the present disclosure;

5 [0009] Fig. 6 illustrates a flowchart of a process for Fourier learning in accordance with some example embodiments of the present disclosure; and

[0010] Fig. 7 illustrates a block diagram of an example computing system/device suitable for implementing example embodiments of the present disclosure.

Detailed Description

10 [0011] Principle of the present disclosure will now be described with reference to some embodiments. It is to be understood that these embodiments are described only for the purpose of illustration and help those skilled in the art to understand and implement the present disclosure, without suggesting any limitation as to the scope of the disclosure. The disclosure described herein can be implemented in various manners other than the ones
15 described below.

[0012] In the following description and claims, unless defined otherwise, all technical and scientific terms used herein have the same meaning as commonly understood by one of ordinary skills in the art to which this disclosure belongs.

20 [0013] References in the present disclosure to “one embodiment,” “an embodiment,” “an example embodiment,” and the like indicate that the embodiment described may include a particular feature, structure, or characteristic, but it is not necessary that every embodiment includes the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an example embodiment, it is submitted
25 that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0014] It shall be understood that although the terms “first” and “second” etc. may be used herein to describe various elements, these elements should not be limited by these terms. These terms are only used to distinguish one element from another. For example, a

first element could be termed a second element, and similarly, a second element could be termed a first element, without departing from the scope of example embodiments. As used herein, the term “and/or” includes any and all combinations of one or more of the listed terms.

5 **[0015]** The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting example embodiments. As used herein, the singular forms “a”, “an” and “the” are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms “comprises”, “comprising”, “has”, “having”, “includes” and/or “including”, when used
10 herein, specify the presence of stated features, elements, and/or components etc., but do not preclude the presence or addition of one or more other features, elements, components and/or combinations thereof.

15 **[0016]** As used herein, the term “model” is referred to as an association between an input and an output learned from training data, and thus a corresponding output may be generated for a given input after the training. The association may be represented by a function, which processes the input and generates the output. The generation of the model may be based on a machine learning technique. The machine learning technique may also be referred to as artificial intelligence (AI) technique. In general, a machine learning model can be built, which receives input information and makes a prediction based on the input information.
20 Such a machine learning model may be referred to as a prediction model. For example, a classification model may predict a class of the input information among a predetermined set of classes, a recommendation model may predict a recommendation result to a user based on context information related to the user, a model applied in a search engine may predict a probability of the hits of a certain keyword based on user behaviors. As used herein, “model”
25 may also be referred to as “machine learning model”, “learning model”, “machine learning network” or “learning network,” which are used interchangeably herein.

30 **[0017]** Generally, machine learning may usually involve three stages, i.e., a training stage, a validation stage, and an application stage (also referred to as an inference stage). At the training stage, a given machine learning model may be trained (or optimized) iteratively using a great amount of training data until the model can obtain, from the training data, consistent inference similar to those that human intelligence can make. During training, a set of parameter values of the model is iteratively updated until a training objective is

reached. Through the training process, the machine learning model may be regarded as being capable of learning the association between the input and the output (also referred to an input-output mapping) from the training data. At the validation stage, a validation input is applied to the trained machine learning model to test whether the model can provide a correct output, so as to determine the performance of the model. At the application stage, the resulting machine learning model may be used to process an actual model input based on the set of parameter values obtained from the training process and to determine the corresponding model output.

[0018] Online machine learning is a method of machine learning in which training data becomes available in a sequential order and is used to update the optimal machine learning model for future data at each step, as opposed to batch learning techniques which generate the optimal machine learning model by learning on the entire training data set at once.

Example Environment

[0019] As mentioned above, it is expected to exploit the periodicity within training data to learn a better prediction model. A prediction model is constructed and utilized according to machine learning techniques. Reference is made to Fig. 1 to describe an environment of machine learning.

[0020] Fig. 1 illustrates a block diagram of an environment 100 in which embodiments of the present disclosure can be implemented. In the environment 100, it is expected to train and apply a machine learning model 105 for a prediction task. The machine learning model 105 may be of any machine learning or deep learning architectures, for example, a neural network.

[0021] In practical systems, the machine learning model 105 may be configured to process an input data sample and generate a prediction result for the input data sample. The prediction task may be defined depending on practical applications where the machine learning model 105 is applied. As an example, in a recommendation system, the prediction task is to predict one or more items or objects which a user is of interest and provide a recommendation to the user based on the prediction. In this example, the input data sample to the machine learning model 105 may comprise context information related to the user such as user information, historical user interactions, and so on, and information related to items to be recommended. The output from the machine learning model 105 is a prediction

result indicating which items or which types of items the user may be of interest. As another example, in a financial application, the prediction task is to predict the sales of a product at a further time. In this example, the input data sample to the machine learning model 105 may comprise the further time, information related to the product and/or other related products, historical sales of the product and/or other related products, information related to target geographical areas and target users of the product, and so on. It would be appreciated that only a limited number of examples are listed above, and the machine learning model 105 may be configured to implement any other prediction tasks.

[0022] The machine learning model 105 may be constructed as a function which processes input data and generates an output as a prediction result. The machine learning model 105 may be configured with a set of parameters whose values are to be learned from training data through a training process. In Fig. 1, the model training system 110 is configured to implement a training process to train the machine learning model 105 based on a training dataset 112. At an initial stage, the machine learning model 105 may be configured with initial parameter values. During the training process, the initial parameter values of the machine learning model 105 may be iteratively updated until a learning objective is achieved.

[0023] The training dataset 112 may include a large number of input data samples provided to the machine learning model 105 and labeling information indicating corresponding groundtruth labels for the input data samples. In some embodiments, an objective function is used to measure the error (or distance) between the outputs of the machine learning model 105 and the groundtruth labels. Such an error is also called a loss of the machine learning, and the objective function may also be referred to as a loss function. The loss function may be represented as $\ell(f(x), y)$, where x represents the input data sample, $f()$ represents the machine learning model, $f(x)$ represents an output of the machine learning model, and y represents a groundtruth label for x . During training, the parameter values of the machine learning model 105 are updated to reduce the error calculated from the objective function. The learning objective may be achieved until the objective function is optimized, for example, until the calculated error is minimized or reaches a desired threshold value.

[0024] After the training process, the trained machine learning model 105 configured

with the updated parameter values may be provided to the model application system 120 which applies a real-world input data sample 122 to the machine learning model 105 to output a prediction result 124 for the input data sample 122.

[0025] In Fig. 1, the model training system 110 and the model application system 120 may be any systems with computing capabilities. It should be appreciated that the components and arrangements in the environment shown in Fig. 1 are only examples, and a computing system suitable for implementing the example implementation described in the subject matter described herein may include one or more different components, other components, and/or different arrangement manners. For example, although shown as separate, the model training system 110 and the model application system 120 may be integrated in the same system or device. The embodiments of the present disclosure are not limited in this respect.

[0026] In some cases, input data processed by a machine learning model may be of a certain periodicity. Such data is called periodic or cyclic data. For examples, users of an application may usually log in the application within relatively fixed time windows each day (e.g. before bed and after work) and show the same interest at the same time window on different days. Such a cyclical pattern may lead to different predicting recommendations to the users. As such, it is expected that the machine learning model 105 may be trained to exploit the periodicity within the training data.

[0027] The problem of exploiting the periodicity within training data to learn a better prediction model may be set up as follows. Given samples denoted by a triplet (x, y, t) , with $x \in \mathcal{X} \subset \mathbb{R}^d$ being the feature of an input data sample, $y \in \mathcal{Y} \subseteq \mathbb{R}$ being a prediction result for the input data sample, and $t \in \mathbb{R}$ being the point of time at which the input data sample is generated, it is expected to learn a prediction model (represented as $f \in \mathcal{F}$) that can predict y with x for any given point of time t . The data samples may arrive in a cyclical fashion. More specifically, between two consecutive updates of the model at t and $t+\delta$, only samples arrived at the interval $[t, t+\delta)$ is available for training. In addition, if (x, y) is generated from a time-dependent distribution $\mathcal{D}_t$, then there exists a periodicity of T such that $\mathcal{D}_t = \mathcal{D}_{t-T}$ for all t . Under the further assumption that, for any (x, y, t) , the triplet $(x, y, \text{mod}(t, T))$ is sampled from a joint distribution $p(\text{mod}(t, T))\mathcal{D}_{\text{mod}(t, T)}(x, y)$, the goal is to solve the following set of optimization

problems for the loss function $\ell(f(x), y)$:

$$f_t^*(x) \in \operatorname{argmin}_{f \in \mathcal{L}_2(\mathcal{X})} \mathbb{E}_{x, y \sim \mathcal{D}_t} [\ell(f(x), y)] \quad \forall t \in \mathbb{R}. \quad (1)$$

[0028] It is assumed that $\mathcal{X}$ and $\mathcal{Y}$ are convex and compact sets, and the loss function $\ell(f(x), y)$ is strongly convex with respect to f for all $y \in \mathcal{Y}$. The above set of optimization problems in Equation (1) may be solved by learning a set of finite-energy and continuous functions $f_t^*(x)$ (which represents the expected prediction model) to minimize the expected loss for each point of time $t \in \mathbb{R}$. The optimization is conducted within the space $\mathcal{L}_2(\mathcal{X}) = \{f : \mathcal{X} \rightarrow \mathbb{R} \mid \int_{\mathcal{X}} f^2(x) dx < \infty\}$, which is a function space that contains all finite-energy functions defined over $\mathcal{X}$.

[0029] The concept of periodicity plays an important role in Equation (1). Specifically, due to periodicity, a function $f_t^*(x)$ for the point of time t is also guaranteed to be a solution at $t+nT$ (where n is an integer larger than zero). This implies that the prediction model learned at time t may offer useful information to improve the prediction accuracy at $t+nT$. Hence, the inventors are motivated to design a learning algorithm that can effectively exploit such useful information offered by the cyclical nature of the data.

[0030] Surprisingly, existing optimization and machine learning techniques offer little insight on how to exploit periodicity within training data to solve Equation (1) efficiently under a big-data setting, whereas industrial systems implement algorithms that simply underrate the periodicity within the training data.

[0031] When faced with machine learning on periodic data, one straight-forward design to encode periodicity into the model structure is to simply include t as a model input and learn a function $f(x, t)$. Unfortunately, this approach does not work out-of-the-box. When the function $f(x, t)$ is represented as a machine learning model, it has been shown that it fails to learn periodicity unless using special activation functions. When $f(x, t)$ belongs to a non-parametric family that encodes periodicity, such as a reproducing kernel Hilbert space (RKHS) with a periodic kernel or a Sobolev-Hilbert space with a periodic spline, the periodicity is automatically encoded across all input dimensions, whereas in Equation (1) $f(x, t)$ may be aperiodic in x .

[0032] An enhanced version of this approach is to pre-process the time t and learn a function represented as $f(x, \text{mod}(t, T))$ instead, which focuses on a single period of $f(x, t)$. Although the pre-processing of t into $\text{mod}(t, T)$ guarantees periodicity during the inference stage, it still often requires laborious feature engineering, especially when x is high-dimensional and $f(x, t)$ has a complicated design.

[0033] Another approach to Equation (1) is to simply learn a prediction model for every t . This is often practically impossible, and hence the time axis is often discretized so that the learner only needs to learn a finite set of models for several discretized points of time, resulting in a pluralistic approach. For machine learning systems, this set of models can share a “base” part of a neural network, and differ only in the last few layers. On the positive side, when the time-dependent distribution $\mathcal{D}_t$ is piece-wise constant in time, e.g., $\mathcal{D}_t = \mathcal{D}_0 \mathbb{1}[0 \leq \text{mod}(t, T) < T/2] + \mathcal{D}_1 \mathbb{1}[T/2 \leq \text{mod}(t, T) < T]$, this approach allows each separate model to converge to its optimal as the time $t/T \rightarrow \infty$. On the other hand, however, this pluralistic approach requires storing multiple models, which is hard to scale for large-scale industrial systems that often cost terabytes of memory space to store. Although computationally efficient methods exist, e.g., partially sharing the network structure between the models, they typically compromise the theoretical guarantees as a trade-off.

[0034] A further solution for training prediction models using sequential data is to follow the online learning protocol, where newly-generated periodic data are applied to optimize the model. The performance of the learning algorithm is typically evaluated using the concept of dynamic regret, which measures the model’s capability to consistently and accurately predict the labels of the latest batch of arriving data. Crudely speaking, when t takes a set of discretized values, the dynamic regret measures $\sum_t \mathbb{E}[\ell(f_t(x), y) - \ell(f_t^*(x), y)]$, the cumulative sum of the differences between the loss under the learned model and the optimal loss under $f_t^*(x)$ defined in Equation (1). Although many optimization algorithms have been proposed to improve the dynamic regret analysis, none of them shed light on how to exploit periodicity within the training data. What is more, when $\mathcal{D}_t$ does not converge to a fixed distribution as t diverges, the dynamic regret scales *linearly* in t , implying that the gap between the learned model and the desired optimal does not vanish even when the data is known to be cyclical.

[0035] In summary, the problem of exploiting the cyclical pattern in data distributions to train a better model remains largely an open problem in a large-scale setting.

Work Principle and Theory Analysis

5 [0036] According to embodiments of the present disclosure, there provides an improved solution to address the challenges of machine learning with cyclical data. This solution proposes a new learning framework, called Fourier learning. Fourier learning can be applied in learning a prediction model for use in various applications where periodic data are generated.

10 [0037] Before describing the applying of Fourier learning within a prediction model, it is first analyzed and proved theoretically how the Fourier learning can solve the problem of exploiting the periodicity within training data to learn a better prediction model, e.g., the optimization problem in Equation (1).

15 [0038] In embodiments of the present disclosure, the proposed Fourier learning can solve the set of optimization problems in Equation (1) as a single optimization problem in a function space that naturally contains time-periodic functions. In particular, the function space may be a tensor product of two Hilbert spaces, one contains model snapshots at a fixed point in time, while the other contains time-periodic functions. As will be demonstrated below, it turns out that this leads to a partial Fourier expansion for these functions. Under a convex analysis setting, it is also possible to learn the Fourier coefficients using streaming-stochastic gradient descent (SGD). Theoretically, the proposed Fourier learning framework can be supported from two different aspects: (i) from a modeling perspective, the Fourier learning is naturally derived from a functional optimization problem that is equivalent to the optimization problem in Equation (1) under a strongly convex and a realizable setting; (ii) in terms of optimization, it is demonstrated that the coefficient functions updated with streaming-SGD provably converge in the frequency domain. For practical applications, the Fourier learning can be integrated to various prediction models, to allow the prediction models to provide more accurate prediction results. By integrating with the Fourier learning, one single model framework may be sufficient for predictions of periodic data.

30 [0039] The theoretical foundations for the proposed Fourier learning is first introduced, which can be derived as a natural solution to a function optimization problem. In

embodiments of the present disclosure, the set of learning problems in Equation (1) is reformulated as one single learning problem in a Hilbert space. In practice, this will allow to learn a unified model that takes both x and t as its inputs. Specifically, the learning objective takes the form of Equation (2) below, where the expectation can be replaced by the empirical mean over datasets in practice:

$$\min_{f \in \mathcal{H}} L(f) := \mathbb{E}_{x, y, t \sim \mathcal{D}_t(x, y) p(t)} \left[\ell(f(x, t), y) \right]. \quad (2)$$

In Equation (2), $f(x, t)$ is a model to be learned to exploit the periodicity of input data x generated at a point of time t , y is the groundtruth label for x , the triplet (x, y, t) is generated from a time-dependent distribution $\mathcal{D}_t(x, y) p(t)$, where $\mathcal{D}_t(x, y)$ is the distribution of (x, y) , and $p(t)$ is the distribution of the point of time t (e.g., $p(t) = 1/T$) for $t \in [0, T]$. According to Equation (2), it is expected to find, from a Hilbert Space $\mathcal{H}$, a model $f(x, t)$ that can minimize a loss function $\ell(f(x, t), y)$ whose loss is calculated between the prediction result from the model $f(x, t)$ and the groundtruth label y .

[0040] An important element in Equation (2) is the design of Hilbert Space $\mathcal{H}$ in which $f(x, t)$ is searched for. For the problem of learning with cyclical data, it is particularly focused on functions in a Hilbert Space that are continuous, periodic in time, and have a finite energy in a single period of time. The inventors have found that the unified objective in Equation (2) is related to Equation (1) via the following Lemma 1.

Lemma 1. For $f_t^*(x)$ in Equation (1), let $f_0(x, t) = f_t^*(x) \quad \forall t \in \mathbb{R}$. If $f_0(x, t) \in \mathcal{H}$, then $f_0(x, t)$ minimizes the loss function $L(f)$ in $\mathcal{H}$.

[0041] In Lemma 1, T represents the periodicity of x . The proof of the above Lemma 1 is as follows:

Starting from Equation (2), for any $f(x, t) \in \mathcal{H}$,

$$\begin{aligned}
L(f_0(x, t)) &= \mathbb{E}_{x, y, t \sim \mathcal{D}_{\text{mod}(t, T)}(x, y) p(\text{mod}(t, T))} \left[\ell(f_0(x, t), y) \right] \\
&= \mathbb{E}_{p(\text{mod}(t, T))} \left\{ \mathbb{E}_{x, y \sim \mathcal{D}_{\text{mod}(t, T)}(x, y)} \left[\ell(f_0(x, t), y) \right] \right\}, \\
&\leq \mathbb{E}_{p(\text{mod}(t, T))} \left\{ \mathbb{E}_{x, y \sim \mathcal{D}_{\text{mod}(t, T)}(x, y)} \left[\ell(f(x, t), y) \right] \right\} \\
&= L(f(x, t)),
\end{aligned} \tag{3}$$

where the inequality follows from the assumption that

$$\mathbb{E}_{x, y \sim \mathcal{D}_{\text{mod}(t, T)}(x, y)} \left[\ell(f_0(x, t), y) \right] \leq \mathbb{E}_{x, y \sim \mathcal{D}_{\text{mod}(t, T)}(x, y)} \left[\ell(f(x, t), y) \right] \tag{4}$$

for any $f(x, t) \in \mathcal{H}$. Hence, $f_0(x, t)$ is a minimizer of Equation (2).

[0042] Lemma 1 implies that, if (2) has a unique minimizer, and if $f_t^*(x)$ belongs to the Hilbert Space $\mathcal{H}$ when treated as a function of both x and t , then the minimizer of (2) leads to the solution of Equation (1). Hence, under a realizable setting, Equation (2) serves as a proxy to solving Equation (1). According to the above proof, it indicates that, under a realizable setting and a strictly convexity used in the Lemma 1, it is possible to obtain a desired set of solutions for Equation (1) by minimizing a proxy loss specified in Equation (2).

[0043] Another critical element in (2) is the design of $\mathcal{H}$. Here, the focus is particularly on functions that are continuous, periodic in time, and have finite energy in a single period. In addition, the functions in $\mathcal{H}$ need to degenerate to $\mathcal{L}_2(\mathcal{X})$ as specified in Equation (1) for every fixed t . Two important elements required for designing such an $\mathcal{H}$ are introduced.

[0044] In addition, defining functions on circles is an important way to characterize periodic functions. As they are defined for points on a circle, these functions take a point's angular information as their inputs, and therefore naturally have a period that is determined by the circle's circumference. To facilitate optimization, a Hilbert space structure is further defined over these functions, based on the intuition that views a circle as a line segment with its end-points glued together, as follows:

$$\mathcal{L}_2(S^T) = \left\{ f : \mathbb{R} \rightarrow \mathbb{R} : \int_0^T |f(t)|^2 dt < \infty \quad \text{and} \quad f(t+T) = f(t) \forall t \in \mathbb{R} \right\}. \tag{5}$$

[0045] Equation (5) indicates that the function f is mapped to such a space where the function f has a finite energy, i.e., $\int_0^T |f(t)|^2 dt < \infty$, and the function f is a periodic function with a periodicity of T , i.e., $f(t+T) = f(t) \forall t \in \mathbb{R}$. As it turns out, if $\langle f, g \rangle \stackrel{\text{def}}{=} \int_0^T f(t)g(t)dt$, then $(\mathcal{L}_2(S^T), \langle \cdot, \cdot \rangle)$ forms a Hilbert space. This Hilbert space meets the needs in the special case when there is no input feature to the model, i.e., when $f(x, t)$ depends on t only.

[0046] To further augment $\mathcal{L}_2(S^T)$ into a Hilbert space that contains functions dependent on both x and t , the concept of tensor product between Hilbert spaces is needed, which is a direct function space extension to the concept of Kronecker product between vectors in Euclidean spaces.

[0047] Specifically, given two Hilbert spaces denoted by $(\mathcal{H}_1, \langle \cdot, \cdot \rangle_1)$ and $(\mathcal{H}_2, \langle \cdot, \cdot \rangle_2)$, respectively, the tensor product of $\mathcal{H}_1$ and $\mathcal{H}_2$ is a Hilbert space $(\mathcal{H} \triangleq \mathcal{H}_1 \otimes \mathcal{H}_2, \langle \cdot, \cdot \rangle)$ coupled by a bi-linear mapping $\phi : \mathcal{H}_1 \times \mathcal{H}_2 \rightarrow \mathcal{H}$. Together, $\mathcal{H}$ and ϕ satisfy the following properties. (i) The set of vectors $\phi(u_1, u_2)$ with $u_1 \in \mathcal{H}_1$ and $u_2 \in \mathcal{H}_2$ must form a total subset of $\mathcal{H}$. That is, $\mathcal{H} = \text{Span}\{\phi(u_1, u_2) | u_1 \in \mathcal{H}_1, u_2 \in \mathcal{H}_2\}$. (ii) The inner product of $\mathcal{H}$, $\langle \cdot, \cdot \rangle$ satisfies $\langle \phi(u_1, u_2), \phi(v_1, v_2) \rangle = \langle u_1, v_1 \rangle_1 \langle u_2, v_2 \rangle_2$ for any $u_1, v_1 \in \mathcal{H}_1$ and $u_2, v_2 \in \mathcal{H}_2$. Adopting two orthonormal sets of basis functions, $\{e_{1i}\}_{i=1}^{\dim(\mathcal{H}_1)}$ and $\{e_{2j}\}_{j=1}^{\dim(\mathcal{H}_2)}$, for $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively, these aforementioned properties would allow us to expand any element $\phi(u_1, u_2) \in \mathcal{H}$ into $\phi(u_1, u_2) = \sum_{i=1}^{\dim(\mathcal{H}_1)} \sum_{j=1}^{\dim(\mathcal{H}_2)} u_{1i} u_{2j} \phi(e_{1i}, e_{2j})$, where $u_{1i} = \langle u_1, e_{1i} \rangle_1$ and $u_{2j} = \langle u_2, e_{2j} \rangle_2$, respectively. Furthermore, when $\mathcal{H}_1 = \mathcal{L}_2(\mathcal{X})$ and $\mathcal{H}_2(\mathcal{Y}) = \mathcal{L}_2(\mathcal{Y})$ as is the case for the above problem, an isomorphism exists such that $\phi(e_{1i}, e_{2j}) \cong e_{1i} e_{2j}$. This implies that it is possible to consider an isomorphism of $\mathcal{H}_1 \otimes \mathcal{H}_2$ containing functions that are linear combinations of $\{e_{1i} e_{2j}\}_{i=1, j=1}^{\infty}$, i.e., $\mathcal{H} = \mathcal{L}_2(\mathcal{X}) \otimes \mathcal{L}_2(\mathcal{Y}) \cong \mathcal{L}_2(\mathcal{X} \times \mathcal{Y})$.

A Tensor-Product-Based Design of $\mathcal{H}$

[0048] To augment $\mathcal{L}_2(S^T)$ by its tensor product with $\mathcal{L}_2(\mathcal{X})$, a natural choice of the Hilbert space $\mathcal{H}$ is to set $\mathcal{H} \triangleq \mathcal{L}_2(\mathcal{X} \times S^T)$, where

$$\mathcal{L}_2(\mathcal{X} \times S^T) = \left\{ f : \mathcal{X} \times \mathbb{R} \rightarrow \mathbb{R} \left| \|f\|_{\mathcal{L}_2(\mathcal{X} \times [0, T])} < \infty \right. \right. \quad (7)$$

$$\left. \text{and } f(x, t) = f(x, t - T) \forall t \right\}.$$

This $\mathcal{L}_2(\mathcal{X} \times S^T)$ expands $\mathcal{L}_2(\mathcal{X})$ with an additional dimension in t defined on a circle with a circumference T , which naturally restricts $f(x, t) \in \mathcal{H}$ to be a periodic function over the time t for any fixed $x \in \mathcal{X}$. The inventors have found the following Lemma 2, which certifies that $\mathcal{H}$ is a Hilbert space and characterizes its basis functions using the isomorphism between $\mathcal{H}$ and $\mathcal{L}_2(\mathcal{X}) \otimes \mathcal{L}_2(S^T)$.

Lemma 2. Let $\mathcal{H}$ be defined in Equation (7). For $f, g \in \mathcal{H}$, let

$$\langle f, g \rangle \stackrel{\text{def}}{=} \int_{\mathcal{X}} \int_0^T f(x, t) g(x, t) dx dt, \quad (8)$$

then $(\mathcal{H}, \langle \cdot, \cdot \rangle)$ is a Hilbert space. Furthermore, there exists an isomorphism between $\mathcal{H}$ and $\mathcal{L}_2(\mathcal{X}) \otimes \mathcal{L}_2(S^T)$, i.e., if $\{\phi_i\}_{i=1}^{\infty}$ and $\{\psi_j\}_{j=1}^{\infty}$ are two orthonormal sets of basis for $\mathcal{L}_2(\mathcal{X})$ and $\mathcal{L}_2(S^T)$, respectively, then $\{\phi_i \psi_j\}_{i,j=1}^{\infty}$, where, $\varphi_{ij}(x, t) \stackrel{\text{def}}{=} \phi_i(x) \psi_j(t)$, is an orthonormal set of basis functions for $\mathcal{H}$.

[0049] The above lemma paves the way for a theoretically guaranteed algorithm that optimizes $L(f)$ through a basis expansion of f in $\mathcal{H}$, which will be introduced in the following. In the meantime, $\mathcal{H}$ is general enough for the learning purpose in the sense that the function $f_0(x, t)$ defined point-wise by solutions in Equation (1) belongs to $\mathcal{H}$ under mild assumptions. Some definitions and assumptions are introduced as below.

[0050] Definition 3 (Continuity under total variation). Let $\mathcal{D}_t(x)$ be the conditional distribution of y given x under $\mathcal{D}_t$. $\mathcal{D}_t(x)$ is considered continuous in t under total variation distance if, for any fixed t and any $\epsilon > 0$, there exists $\delta > 0$ such that $\|\mathcal{D}_{t'}(x) - \mathcal{D}_t(x)\|_{\text{TV}} \leq \epsilon$ wherever $|t' - t| \leq \delta$.

[0051] Assumption 4. Suppose: (i) $\mathcal{X}$ and $\mathcal{Y}$ are compact and convex sets; (ii) $\mathcal{D}_t(x)$ in Definition 3 is continuous under total variation for all $x \in \mathcal{X}$; (iii) the loss function $\ell(f(x), y)$ is σ -strongly-convex in its first argument for all $y \in \mathcal{Y}$; (iv) $f(x)$ in (1) is bounded and $\max_{x \in \mathcal{X}, y \in \mathcal{Y}} \ell(f(x), y) \leq K$ for some constant K .

5 **[0052]** Assumption 4 can be easily satisfied by a wide range of machine learning systems. For instance, deep neural networks (DNNs) typically have bounded outputs when a clipping on the final output is enforced. The uniform strong convexity of the loss function also holds for a wide range of ℓ such as the mean square loss. With the above definition and assumption, the inventors have found another Lemma, Lemma 5.

10 **[0053]** Lemma 5. Under Assumption 4, $f_t^*(x)$ is continuous in t for any given $x \in \mathcal{X}$. In addition, $f_0(x, t) \stackrel{\text{def}}{=} f_t^*(x) \in \mathcal{H}$.

[0054] Lemma 5 implies that, under Assumption 4, the optimal solution of Equation (2), $f_0(x, t)$, belongs to $\mathcal{H}$. Combining Lemmas 1 and 5, it can be seen that the satisfaction of Assumption 4 allows us to acquire a set of desired solution of Equation (1) by solving
15 Equation (2).

Fourier Learning with Periodic Data

[0055] It now proceeds to introduce Fourier learning, a learning framework that hard-wires the periodicity of the data-distribution into the model's structure via a partial Fourier expansion and learns the model by learning its Fourier coefficient functions. To do so, from
20 a modeling aspect, Lemma 2 is invoked and $f(x, t) \in \mathcal{H}$ may be represented with the following basis expansion:

$$f(x, t) = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} c_{i,j} \phi_i(x) \psi_j(t) = \sum_{j=1}^{\infty} c_j(x) \psi_j(t), \quad (9)$$

where $c_j(x) \in \mathcal{L}_2(\mathcal{X})$ is the sum of $c_{i,j} \phi_i(x)$ over i . It is noticed that a set of basis functions for $\mathcal{L}_2(S^T)$ is the trigonometric functions with a base frequency $1/T$. The inventors reach the following Theorem.

25 **Theorem 6.** Any function $f(x, t) \in \mathcal{H}$ can be represented by a Fourier expansion:

$$f(x, t) = \sum_{n=0}^{\infty} \left[a_n(x) \sin \left[\frac{2\pi nt}{T} \right] + b_n(x) \cos \left[\frac{2\pi nt}{T} \right] \right], \quad (10)$$

where $a_n(x), b_n(x) \in \mathcal{L}_2(\mathcal{X})$, and $a_0(x) \equiv 0$.

[0056] Theorem 6 provides an explicit way of designing periodic models and specifies how the time-feature could be exploited. Note that, it is entirely possible to construct $\mathcal{H}$ with a weighted $\mathcal{L}_2$ space defined on circles to guarantee periodicity. This allows us to deviate from the trigonometric functions and use potentially other periodic functions to encode periodicity.

[0057] With $f(x, t)$ represented by Equation (10), the solution of the problem in Equation (2) is reduced to learn $a_n(x)$ and $b_n(x)$, i.e., the Fourier coefficients of $f(x, t)$ that are now independent of t and only dependent on x . In addition, the sine and cosine components are dependent on t . Since Equation (10) takes the form of a partial Fourier expansion of $f(x, t)$, this learning method may be referred to as “Fourier learning”.

[0058] Fourier learning allows the designer to retain his original model design, but at the same time mixes the last hidden layer's expert's advice in a time-dependent fashion. This explicit role of t in the prediction model circumvents the laborious feature engineering required when the feature t is implicitly added to the model in the form of $f(x, t)$.

[0059] The goal now shifts towards learning the coefficient functions in the frequency domain. For tractable learning, a cutoff frequency N/T may be introduced and thus a truncated Fourier expansion of $f(x, t)$ may instead be represented as follows:

$$f_N(x, t) = \sum_{n=0}^N \left[a_n(x) \sin \left[\frac{2\pi nt}{T} \right] + b_n(x) \cos \left[\frac{2\pi nt}{T} \right] \right], \quad (11)$$

where N is a predetermined number, which is an integer larger than one.

[0060] The truncated Fourier expansion in Equation (11) is an approximation to the Fourier expansion in Equation (10). The approximation error for all $f \in \mathcal{H}$ in Equation (11) may be denoted as $E_N(f)$, which may be determined as follows:

$$E_N(f) = \left\| f(x, t) - f_N(x, t) \right\|_{\mathcal{H}}^2. \quad (12)$$

In practical systems, it can be commonly assumed that $\lim_{N \rightarrow \infty} E_N(f) = 0$ for all $f \in \mathcal{H}$. Hence, with a properly selected N , the approximation error may be limited as well as the amount of model parameters.

[0061] In the Fourier expansion for Fourier learning in Equation (11), Fourier coefficients $a_n(x)$ and $b_n(x)$ are needed to be determined so as to generate a prediction result of the model $f(x, t)$. The Fourier coefficients $a_n(x)$ and $b_n(x)$ may be considered as coefficient functions dependent on x , which can be learned under a variety of regimes. For example, they can be learned non-parametrically using function optimization algorithms.

[0062] In some embodiments, if the Fourier coefficients $a_n(x)$ and $b_n(x)$ have a parametric form, such as a neural network, stochastic gradient descent is known to converge to stationary point at a certain rate under standard assumptions, which may be introduced in detail below. In some embodiments, apart from the above parametric framework, Fourier learning also fits into the non-parametric regime, which may be introduced in detail below.

Machine Learning System based on Fourier Learning

[0063] In the following, it is discussed parameterizing of $a_n(x)$ and $b_n(x)$ with neural networks, so as to apply Fourier learning to large-scale machine learning scenarios.

[0064] The above theory analysis indicates that a model constructed based on a Fourier learning can intuitively exploit the periodicity of training data and can be expressed as a periodic function with the periodicity. Thus, the Fourier learning can be adapted to a machine learning-based prediction model. In embodiments of the present disclosure, according to the Fourier expansion of $f(x, t)$ in Equation (11), it is proposed to view x as information related to an input data sample generated at a certain point of time t . The input data sample may be of a data sample of periodic data. A Fourier expansion result can be determined based on the Fourier expansion and a prediction result for the input data sample is then determined based on the Fourier expansion result.

[0065] Reference is now made to Fig. 2, which illustrates a block diagram of a machine

learning system 200 with Fourier learning in accordance with some example embodiments of the present disclosure. The machine learning system 200 may be implemented as the machine learning model 105 in the environment 100. As illustrated, the machine learning system 200 comprises a prediction model 210 and a Fourier layer 220.

5 **[0066]** The prediction model 210 may be configured with any model architectures to implement a prediction task. In embodiments of the present disclosure, input data to be processed by the prediction model 210 are periodic data with a certain periodicity (represented as T). The input to the prediction model 210 is an input data sample generated at a certain point of time t . The Fourier layer 220 is introduced to allow generating more
10 accurate prediction results by considering the periodicity within the input data. It is noted that the prediction model 210 may be constructed in any manner which may or may not exploit the periodicity of the input data because in either case, the addition of the Fourier layer 220 can further exploit the periodicity.

[0067] The Fourier layer 220 is designed by the following intuition: if x is considered as
15 the output of an original prediction model's last hidden layer, then Equation (11) can be viewed as the network's output layer with an architecture shown in Fig. 2. Specifically, the Fourier layer 220 first transforms x into $a_n(x)$ and $b_n(x)$, and then element-wise multiplies them with basis vectors SIN and COS, yielding a $(2N+1)$ -dimension result. This result may be then added up, yielding a scalar output. Notably, when $a_n(x) = b_n(x)$
20 $=0$ for all $n \geq 1$, the final output equals $b_0(x)$, which, by itself, can be interpreted as the original model's output. This implies that replacing the original model's output layer with the Fourier layer 220 increases its capacity, avoiding the need for laborious feature engineering.

[0068] In particular, the Fourier layer 220 receives a feature representation of the input
25 data sample extracted by the prediction model 210. The prediction model 210 may generally be considered as consisting of two parts, one is to extract hidden features within the input data sample and the other one is to determine a model output based on the final hidden feature. In some embodiments, the prediction model 210 may comprise a plurality of layers, including an input layer to receive the input data sample, one or more hidden layers to
30 process the input data sample and generate a feature representation to characterize hidden features within the input data sample, and an output layer to generate the model output. The

layers of the prediction model 210 are connected layer-by-layer and an output from a layer being provided to a next layer as an input.

[0069] In some embodiments, the feature representation extracted at a last hidden layer 212 of the prediction model 210 is provided to the Fourier layer 220 as its input. This feature representation is represented as $\mathcal{X}$. Typically, the input data sample may comprise redundant information and may be of a higher dimension. Through the feature extraction in the prediction model, the feature representation may be able to characterize useful feature information within the input data sample with a relatively small dimension. The Fourier layer 220 may be able to further process the feature representation $\mathcal{X}$ to generate a prediction result for the input data sample.

[0070] It is assumed that the feature representation $\mathcal{X}$ is of a dimension d_1 , and the prediction result for the input data sample is of a dimension d_2 . The dimension of the feature representation $\mathcal{X}$ and the dimension of the prediction result (d_2) may depend on the configuration of the prediction model 210. Generally, d_1 is larger than one, and d_2 may be equal to or larger than one. For example, the prediction result may be a single-dimensional output to indicate, for example, a probability of a user being interest of a target item, or may be a multi-dimensional output to indicate, for example, respective probabilities of a user being interest of a plurality of items.

[0071] Given the input (i.e., the feature representation $\mathcal{X}$) and the output (i.e., the prediction result) of the Fourier layer 220, the processing of the Fourier layer 220 may be considered as mapping the input with a dimension of d_1 to the output with a dimension of d_2 . The model structure of the Fourier layer 220 may be designed to implement such mapping based on the Fourier expansion.

[0072] As illustrated, the Fourier layer 220 comprises a mapping model 230 to generate Fourier coefficients $a_n(x)$ in a Fourier expansion, and a mapping model 240 to generate Fourier coefficients $b_n(x)$ in the Fourier expansion. Following the truncated Fourier expansion in Equation (11) which has a predetermined number $(N+1)$ of terms, the mapping model 230 may be configured to transform the feature representation $\mathcal{X}$ with the dimension d_1 into an output with a dimension N , and the mapping model 240 may be configured to transform the feature representation $\mathcal{X}$ with the dimension d_1 into an output with a dimension of $(N+1)$. The N elements in the output of the mapping model 230 may be

determined as N Fourier coefficients $\{a_n(x)\}_{n=1}^N$. The $(N+1)$ elements in the output of the mapping model 240 may be determined as $(N+1)$ Fourier coefficients $\{b_n(x)\}_{n=0}^N$.

[0073] The mapping model 230 and the mapping model 240 may be constructed based on any machine learning architecture. In some embodiments, the mapping model 230 and the mapping model 240 may be constructed without activation functions. Generally, an activation function applied in a machine learning model (e.g., sigmoid function, tanh function, ReLU function) may restrict the amplitude of the model output into a certain range. In the Fourier expansion, since there is no explicit restriction on the amplitude of the Fourier coefficients, the mapping model 230 and the mapping model 240 may be configured with no activations.

[0074] A Fourier expansion generally comprises a sine function-based component and a cosine function-based component. Accordingly, as illustrated in Fig. 2, the Fourier layer 220 further comprises a sine function unit 232 to determine values for the sine component in the Fourier expansion, and a cosine function unit 242 to determine values for the cosine component in the Fourier expansion. As shown in Equation (11), the sine component is based on a sine function dependent on the point of time t which is a periodic function with the periodicity of T , and the cosine component is based on a cosine function dependent on the point of time t which is a periodic function with the periodicity of T .

[0075] The sine function unit 232 may provide a set of sine component values as a column vector $[\sin(2\pi t/T), \dots, \sin(2\pi Nt/T)]^\top$, and the cosine function unit 242 may provide a set of cosine component values as a column vector $[1, \cos(2\pi t/T), \dots, \cos(2\pi Nt/T)]^\top$. In some embodiments, the time is variable with $t \in [0, T]$. That is, the actual point of time when the input data sample is generated is transformed to a point within a period of T , for example, through the *mod* operation.

[0076] The N sine component values $[\sin(2\pi t/T), \dots, \sin(2\pi Nt/T)]^\top$ may be generated by shifting a frequency of the sine function for N times, and the $(N+1)$ cosine component values $[1, \cos(2\pi t/T), \dots, \cos(2\pi Nt/T)]^\top$ may be generated by shifting a frequency of the sine function for $(N+1)$ times. Starting from a phase of zero, for each time of phase shifting, a phase shift of $2\pi t/T$ is applied to the sine function and the cosine functions. It is noted that the sine function and the cosine function may be phase-shifted for

a same number of $(N+1)$ times, but at the initial time the sine component value at the zero phrase is zero.

5 [0077] At the Fourier layer 220, the Fourier coefficients $\{a_n(x)\}_{n=1}^N$ and $\{b_n(x)\}_{n=0}^N$ are determined in real time in response to the input data sample generated at each point of time. In some embodiments, the sine and cosine component values $\sin(2\pi t/T), \dots, \sin(2\pi Nt/T)$ and $1, \cos(2\pi t/T), \dots, \cos(2\pi Nt/T)$ may be pre-calculated and stored in memory for use.

10 [0078] The N sine component values and the N Fourier coefficients are provided to a multiplier 234. The multiplier 234 is configured to perform element-wise multiplication on the N sine component values and the N Fourier coefficients to generate N products. The $(N+1)$ cosine component values and the $(N+1)$ Fourier coefficients are provided to a multiplier 244. The multiplier 244 is configured to perform element-wise multiplication on the $(N+1)$ cosine component values and the $(N+1)$ Fourier coefficients to generate $(N+1)$ products. The $(2N+1)$ products are corresponding to the individual terms involved in the Fourier expansion.

15 [0079] In order to obtain a prediction result with the dimension of d_2 , the N products from the multiplier 234 may be input into a mapping model 236, and the $(N+1)$ products from the multiplier 244 may be input into a mapping model 246. The mapping model 236 may be configured to transform the N products from the multiplier 234 into a first intermediate expansion result with a dimension of d_2 , and the mapping model 246 may be configured to transform the $(N+1)$ products from the multiplier 244 into a second intermediate expansion result with a dimension of d_2 . The first and second intermediate expansion results may be provided to an aggregator 250, which is configured to perform an element-wise summation on the first and second intermediate expansion results to provide a Fourier expansion result, which may be determined as a prediction result for the input data sample with a dimension of d_2 .

25 [0080] In some embodiments, if the prediction result is a single-dimensional output, the mapping models 236 and 246 may be omitted from the Fourier layer 220. In this case, the products from the multiplier 234 and 244 are summed up to provide a Fourier expansion result, which may be determined as the prediction result.

[0081] In some embodiments, the mapping model 230 and the mapping model 240 may be constructed as multi-layer perceptron (MLP) models. In some embodiments, the mapping model 236 and the mapping model 246 may be constructed as MLP models. The Fourier layer 220 may thus be considered as a Fourier-MLP (F-MLP) layer.

5 **[0082]** The parameter values of the mapping models 220, 240, 236, and 246 in the Fourier layer 220 may be determined through a training process. In some embodiments, these mapping models may be trained with the prediction model 210. The training data may include input data samples to the prediction model 210 and labeling information indicating corresponding groundtruth labels for the input data samples. In some embodiments, the
10 mapping models in the Fourier layers may be trained in an end-to-end manner with the prediction model 210. In some embodiments, the prediction model 210 may be first trained and then retrained together with the mapping models in the Fourier layers.

[0083] In some embodiments, the Fourier layer 220 may be generalized as F-MLP $_{d_1 \rightarrow d_2}^{[N]}(x) \in \mathbb{R}^{d_2 \times 1}$, for an F-MLP with input dimension d_1 and output dimension
15 d_2 , its processing may be represented follows:

$$\text{F-MLP}_{d_1 \rightarrow d_2}^{[N]}(x) = (W_2 \odot \text{COS}) \cdot \text{MLP}_{d_1 \rightarrow (N+1)}(x) + (W_1 \odot \text{SIN}) \cdot \text{MLP}_{d_1 \rightarrow N}(x), \quad (13)$$

where $x \in \mathbb{R}^{d_1 \times 1}$ is the input to the Fourier layer 220, $\text{MLP}_{d_1 \rightarrow N}(x) \in \mathbb{R}^{N \times 1}$ is a regular MLP that maps x into a vector of dimension N , having no activations; $W_1 \in \mathbb{R}^{d_2 \times N}$ and $W_2 \in \mathbb{R}^{d_2 \times (N+1)}$ are the parameter values; while SIN and COS are matrices stacked up by row vectors $[\sin(2\pi t/T), \dots, \sin(2\pi Nt/T)]$ and
20 $[1, \cos(2\pi t/T), \dots, \cos(2\pi Nt/T)]$ a total of d_2 times. The operator $\odot$ is the Hadamard product. When $d_2=1$ (which means that the output is one-dimensional), W_1 and W_2 can be merged into $\text{MLP}_{d_1 \rightarrow N}$ and $\text{MLP}_{d_1 \rightarrow (N+1)}$, which serve the role of $a_n(x)$ and $b_n(x)$ in Equation (11), respectively.

[0084] It is noted that there are some approaches that propose to combine Fourier
25 analysis with deep learning systems. However, most of those approaches are aimed at learning the intrinsic high-frequency component within the distribution of the input data itself, rather than focusing on the periodicity of the distribution in time. In particular, the inventors have observed that the implementation of the Fourier layer of the present

disclosure into existing designs of prediction model fundamentally alters the physical meaning of each processing unit in the model: under a regular model, each processing unit is an expert that changes its decisions through time; under the Fourier layer of the present disclosure, each processing unit holds the frequency component of an expert that decides
 5 how drastically he alters his decisions through time at a given frequency. The former requires designing an online-learning algorithm to track a constantly varying optimal for each expert, while for the latter the future optimal can be predicted using the interpolation with the trigonometric functions. This offers advantage over the online learning approach.

[0085] In the example embodiments of Fig. 2, the Fourier layer 220 is introduced as an
 10 output layer for the prediction model 210, and thus its output is determined as the prediction result for the input data sample. In some embodiments, the Fourier layer 220 may operate with a complete prediction model 210 (comprising its own output layer), and the output from the Fourier layer 220 and the output from the prediction model 210 are aggregated to generate a final prediction result. Fig. 3 illustrates the machine learning system 200 in
 15 accordance with such embodiments.

[0086] As illustrated in Fig. 3, the prediction model 210 comprises, among other layers, an output layer 312 which receives the feature representation $\mathcal{X}$ from the last hidden layer 212. The output layer 312 in the prediction model 210 may process the feature representation $\mathcal{X}$ and generate an intermediate prediction result. The processing in the
 20 output layer 312 may depend on the configuration of the prediction model 210, which may be varied in different prediction tasks. The Fourier layer 220 may also receive the feature representation $\mathcal{X}$ from the last hidden layer 212 and generate an intermediate prediction result based on a Fourier expansion result, as discussed according to the embodiments with respect to Fig. 2.

[0087] The machine learning system 200 may further comprise an aggregator 330 which
 25 is configured to mix the two intermediate prediction results from the prediction model 210 and the Fourier layer 220. For example, the aggregator 330 may determine a weighted-sum of the two intermediate prediction results. The aggregation of the intermediate prediction results may be represented as follows:

$$f(x, t) = \lambda f_{\text{geo}}(x, t) + (1 - \lambda) f_{\text{ff}}(x, t) \quad (14)$$

where $f(x, t)$ represents the prediction result for the input data sample; $f_{\text{oco}}(x, t)$ represents the intermediate prediction result by the output layer 312 of the prediction model 210; and $f_{\text{fl}}(x, t)$ represents the intermediate prediction result generated by the Fourier layer 220; λ is a parameter used to weight the two intermediate prediction results. λ may be of a predetermined value, e.g., 0.5 or any other value.

[0088] In some embodiments, the prediction model 210 may have a complicated structure, for example, may comprise a plurality of sub-models having different model structures. In this case, the input to the Fourier layer 220 may be carefully designed. Fig. 3 illustrates the machine learning system 200 in accordance with such embodiments.

[0089] As illustrated in Fig. 4, the prediction model 210 may comprise a plurality of sub-models (e.g., K sub-models), such as a sub-model 410-1, . . . , a sub-model 410- K (collectively or individually referred to as sub-models 410 for the purpose of discussion). K is an integer larger than one. The outputs of the sub-models may be added up at the output layer 312 in the prediction model 210 to provide an output of the model. In this case, the output layer 312 may comprise an aggregator to perform a summation on the respective outputs from the K sub-models 410. The prediction model 210 may thus be represented as $f_{\text{oco}}(x, t) = \sum_{m=1}^K f_m(x, t)$, where $f_m(x, t)$ represents the output of the m -th sub-model 410.

[0090] With the structure of the prediction model 210 illustrated in Fig. 4, a sub-model 410 may extract a feature representation from the input data sample at its last hidden layer and determine its own output at its output layer based on the feature representation. The feature representations from the sub-models 410 may be aggregated to generate the feature representation x to be input into the Fourier layer 220. In some embodiments, the feature representations from the sub-models 410 may be of different dimensions. In the embodiments of Fig. 4, to aggregate the feature representations from the sub-models 410, the machine learning model 200 may further comprise a dimension aligning layer 420, to transform respective feature representations with different dimensions from the sub-models 410 into feature representations with the same dimension. In some embodiments, the dimension aligning layer 420 may comprise a plurality of MLPs, each configured to transform the feature representation from one of the sub-models 410 to a feature representation with the same dimension.

[0091] Since the Fourier layer 220 performs liner transform on its input, the feature representations with the same dimension generated from the dimension aligning layer 420 may be added together to obtain the feature representation $\mathcal{X}$ to input into the Fourier layer 220. In this case, the dimension aligning layer 420 may transform the feature representations with different dimensions from the sub-models 410 into feature representations with the dimension of d_1 .

[0092] In Fig. 4, it is illustrated that the output from the Fourier layer 220 is aggregated with the output from the prediction model 210 by the aggregator 330 as in Fig. 3. It would be appreciated that in other embodiments, the processing on the feature representations may be integrated into the system 200 illustrated in Fig. 2.

Training of Machine Learning System

[0093] The training of the Fourier layer 220 is performed jointly with the prediction model, following the procedure of streaming-SGD. This procedure is different from the standard SGD, which in practice would need sample data $(x, y, t) \sim \mathcal{D}_t(x, y)p(t)$. However, sampling from $p(t)$ is difficult for many online applications due to the real-time update requirement, where data arrives sequentially. Here we show that using streaming-SGD can avoid this issue while still having good practical performances and convergence guarantees.

[0094] The training procedure is as follows. $a_n(x)$ and $b_n(x)$ may be parameterized by $a_n(x; \theta_n)$ and $b_n(x; \rho_n)$, respectively, with θ_n and ρ_n being the neural network parameters. For cyclical data, the τ -th mini-batch of data may be collected in the k -th cycle, and the model may be updated with the following update rule:

$$\begin{aligned}\theta_n^{(k, \tau+1)} &= \theta_n^{(k, \tau)} - \eta_{k, \tau} g_{\theta_n}^{(k, \tau)}, \\ \rho_n^{(k, \tau+1)} &= \rho_n^{(k, \tau)} - \eta_{k, \tau} g_{\rho_n}^{(k, \tau)}.\end{aligned}\tag{15}$$

Here, $g_{\theta_n}^{(k, \tau)}$ and $g_{\rho_n}^{(k, \tau)}$ are gradients calculated using the collected mini-batch of data:

$$\begin{aligned} g_{\theta_n}^{(k,\tau)} &= \nabla_{\theta_n} \widehat{L}_{k,\tau}(f_N^{(k,\tau)}), \\ g_{\rho_n}^{(k,\tau)} &= \nabla_{\rho_n} \widehat{L}_{k,\tau}(f_N^{(k,\tau)}), \end{aligned} \quad (16)$$

where $f_N^{(k,\tau)}$ is computed with Equation (11), while $\widehat{L}_{k,\tau}$ is the empirical version of the loss in Equation (2) over this mini-batch of data. The overall training procedure is summarized in Algorithm 500 as illustrated in Fig. 5. The convergence analysis of it is presented as below.

- 5 **[0095]** The convergence properties are discussed when training the machine learning system based on Fourier learning with streaming-SGD. Recall that, using a truncated $f_N(x, t)$, the problem in Equation (2) reduces into finding the optimal of

$$\min_{\{a_n\}_{n=1}^N, \{b_n\}_{n=0}^N} \mathbb{E}_{x,y,t \sim \mathcal{D}_t(x,y)p(t)} \left[\ell(f_N(x, t), y) \right] \quad (17)$$

in frequency domain. The optimal set of coefficient functions of Equation (11) is denoted as $a_n^*(x)$ and $b_n^*(x)$, for which the corresponding model $f_N^*(x, t)$ can be expressed as

$$f_N^*(x, t) = \sum_{n=0}^N \left[a_n^*(x) \sin \left[\frac{2\pi n t}{T} \right] + b_n^*(x) \cos \left[\frac{2\pi n t}{T} \right] \right]. \quad (18)$$

[0096] In the following, it is first shown a gradient norm convergence result for streaming-SGD under a general non-convex setting, and then introduce a global convergence result under the assumption of strong convexity. Prior to that, some additional assumptions are introduced.

- 15 **[0097]** Assumption 7. Suppose: (i) for all n, k, τ , the second moment of the update directions are bounded: $\max\{\mathbb{E}[\|g_{\rho_n}^{(k,\tau)}\|_2^2 | \mathcal{F}^{(k)}], \mathbb{E}[\|g_{\theta_n}^{(k,\tau)}\|_2^2 | \mathcal{F}^{(k)}]\} \leq G^2$ for some $G \in \mathbb{R}$, where $\mathcal{F}^{(k)}$ is the minimum σ -algebra generated by $a_n^{(\kappa,\tau)}$ and $b_n^{(\kappa,\tau)}$ for all $n, \kappa < k$ and $1 < \tau < \Gamma$. In addition, it is assumed there exists $\Lambda > 0$ such that $\|\nabla L(f_1) - \nabla L(f_2)\| \leq \Lambda \|f_1 - f_2\|$ for all $f_1, f_2 \in \mathcal{H}$.

- 20 **[0098]** Assumption 7 assumes bounded second moment of the update directions, and the

Lipschitzness of the gradient, which are usually required in the convergence analysis of SGD-type algorithms. The following result shows that streaming-SGD with a proper learning rate achieves convergence under both non-convex and strongly convex settings.

[0099] Theorem 8 (Convergence of Streaming-SGD). *Let (i), (ii) of Assumption 4 and Assumption 7 hold, and define $\|\nabla L(f_N^{(k,1)})\|$ as the gradient with respect to a joint parameter vector combining all $\theta_n^{(k,1)}$ and $\rho_n^{(k,1)}$.*

• Let $\eta_{k,\tau} \triangleq \eta_k = \Theta(1/\sqrt{T_{\max} + 1})$, then

$$\min_{0 \leq k \leq T_{\max}} \|\nabla L(f_N^{(k,1)})\|^2 = \mathcal{O}(1/\sqrt{T_{\max} + 1}).$$

• Let $\eta_{k,\tau} \triangleq \eta_k = \Theta(1/\sqrt{k})$, then

$$\min_{0 \leq k \leq T_{\max}} \|\nabla L(f_N^{(k,1)})\|^2 = \mathcal{O}\left(\frac{\log(T_{\max} + 1)}{\sqrt{T_{\max} + 1}}\right).$$

Moreover, if L is σ -strongly convex with respect to θ_n and ρ_n , we can take

$\eta_{k,\tau} = \psi/k$ with $\psi < (2\sigma^2)^{-1}$ and obtain

$$\mathbb{E}\|\theta_n^{(T_{\max},1)} - \theta_n^*\|_2^2 = \mathcal{O}(T_{\max}^{-1}),$$

$$\mathbb{E}\|\rho_n^{(T_{\max},1)} - \rho_n^*\|_2^2 = \mathcal{O}(T_{\max}^{-1}),$$

where it is assumed $a_n(x, \theta_n^*) = a_n^*(x)$, $b_n(x, \rho_n^*) = b_n^*(x)$.

[00100] Simply put, the learning framework offers a convergence rate of $\mathcal{O}(1/\sqrt{T_{\max}})$ under a general non-convex setting and $\mathcal{O}(1/T_{\max})$ under a strongly convex setting. If $N \rightarrow \infty$ is further derived, the overall learning error of $f_0(x, t)$ can be driven to arbitrarily small. Compared to the online learning benchmark whose dynamic regret is affected by both the changing speed of the data-generating distribution and the variance of the stochastic gradients, Fourier learning yield a much smaller learning error and hence offers a potentially much better performance in many practical scenarios.

[00101] Apart from the above parametric framework, as mentioned above, in some embodiments, the proposed Fourier learning also fits into the non-parametric regime, where

a_n and b_n are updated directly:

$$\begin{aligned} a_n^{(k,\tau+1)}(x) &= a_n^{(k,\tau)}(x) - \eta_{k,\tau} \cdot g_{a_n}^{(k,\tau)}(x), \\ b_n^{(k,\tau+1)}(x) &= b_n^{(k,\tau)}(x) - \eta_{k,\tau} \cdot g_{b_n}^{(k,\tau)}(x). \end{aligned} \quad (19)$$

[00102] As the functional gradients in $\mathcal{L}_2$ often contain Dirac's δ -functions, causing discontinuous updates, we substitute the functional gradients with their kernel embeddings instead. Specifically, with $K(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ being a positive definite kernel
5 whose minimum eigen-value is bounded away from 0, let

$$\begin{aligned} g_{a_n}^{(k,\tau)}(x) &= \langle \nabla_{a_n} \widehat{L}_{k,\tau}(f_N^{(k,\tau)})(\cdot), K(x, \cdot) \rangle, \\ g_{b_n}^{(k,\tau)}(x) &= \langle [\nabla_{b_n} \widehat{L}_{k,\tau}(f_N^{(k,\tau)})](\cdot), K(x, \cdot) \rangle. \end{aligned} \quad (20)$$

It is easy to verify that these kernel embeddings yield continuous updates of $a_n(x)$ and $b_n(x)$ at each iteration. At the same time, $g_{a_n}^{(k,\tau)}$ and $g_{b_n}^{(k,\tau)}$ are “close enough” to the exact gradients and retain the convergence guarantees (Yang et al., 2019). If we initialize $a_n^{(0,\tau)}$ and $b_n^{(0,\tau)}$ to be zeros, then $a_n^{(k,\tau)}$ and $b_n^{(k,\tau)}$ can be written as a linear
10 combination of a finite set of kernels. The convergence result for the non-parametric case is given in Theorem 9 below.

Theorem 9. *Let Assumption 4 and Assumption 7 hold. Let $g_{a_n}^{(k,\tau)}$ and $g_{b_n}^{(k,\tau)}$ be the kernel embeddings of functional gradient with $K(\cdot, \cdot)$ at iteration (k, τ) , as defined in Equation (20). Let $\eta_{k,\tau} = \sigma(k + 0.5)\lambda^{-1}(k + 1)^{-2}$. Then, $\mathbb{E}\|a_n^{(T_{\max}, 1)} - a_n^*\|_2^2 = \mathcal{O}(T_{\max}^{-1})$
15 and $\mathbb{E}\|b_n^{(T_{\max}, 1)} - b_n^*\|_2^2 = \mathcal{O}(T_{\max}^{-1})$, with a_n^* and b_n^* defined in Equation (18).*

Example Process

[00103] Fig. 6 illustrates a flowchart of a process 600 for Fourier learning in accordance with some example embodiments of the present disclosure. The process 600 may be implemented at the machine learning system 200, or may be implemented by the model
20 application system 120 which can apply input data to the machine learning system 200 to perform the corresponding prediction tasks. For the purpose of discussion, reference is made

to Fig. 1 to discuss the process 600.

5 [00104] At block 610, the model application system 120 obtains a feature representation of an input data sample from a prediction model. The prediction model is configured to process input data with a periodicity. The input data sample is a sample of the input data generated at a point of time within a period.

10 [00105] At block 620, the model application system 120 determines first Fourier coefficients for a first component in a Fourier expansion by applying the feature representation into the first mapping model. The Fourier expansion is of the periodicity and dependent on the point of time and the feature representation. At block 620, the model application system 120 determines second Fourier coefficients for a second component in the Fourier expansion by applying the feature representation into a second mapping model.

15 [00106] At block 640, the model application system 120 determines a Fourier expansion result based on the first Fourier coefficients and the second Fourier coefficients in the Fourier expansion. At block 650, the model application system 120 determines a prediction result for the input data sample based on the Fourier expansion result.

[00107] In some embodiments, the Fourier expansion comprises a truncated Fourier expansion with a predetermined number of terms, and the number of the first Fourier coefficients and the number of the second Fourier coefficients are based on the predetermined number.

20 [00108] In some embodiments, the first component is based on a sine function dependent on the point of time and having the periodicity, and the second component is based on a cosine function with the periodicity.

25 [00109] In some embodiments, to determine the Fourier expansion result, the model application system 120 determines a set of first component values for the first component by shifting a frequency of the sine function for the predetermined number of times, and determines a set of second component values for the second component by shifting a frequency of the cosine function for the predetermined number of times. The model application system 120 determines the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the
30 second Fourier coefficients with the second component values.

[00110] In some embodiments, to determine the Fourier expansion result, the model application system 120 calculates first products by multiplying the first Fourier coefficients with the first component values and calculates second products by multiplying the second Fourier coefficients with the second component values. The model application system 120 maps the first products to a first intermediate expansion result using a third mapping model, and maps the second products to a second intermediate expansion result using a fourth mapping model. The model application system 120 determines the Fourier expansion result by aggregating the first intermediate expansion result and the second intermediate expansion result.

[00111] In some embodiments, to determine the prediction result, the model application system 120 determines a first intermediate prediction result from the Fourier expansion result, and obtains a second intermediate prediction result generated from an output layer of the prediction model based on the feature representation. The model application system 120 determines the prediction result by aggregating the first intermediate prediction result and the second intermediate prediction result.

[00112] In some embodiments, the prediction model comprises a plurality of sub-models configured to extract a plurality of feature representations from the input data sample. In some embodiments, the model application system 120 obtains the plurality of feature representations from the plurality of sub-models, and generates the feature representation by aggregating the plurality of feature representations.

[00113] In some embodiments, the first mapping model and the second mapping model are constructed without activation functions. In some embodiments, the third mapping model and the fourth mapping model are constructed without activation functions. In some embodiments, the mapping models are trained jointly with the prediction model.

Example System/Device

[00114] Fig. 7 illustrates a block diagram of an example computing system/device 700 suitable for implementing example embodiments of the present disclosure. The model application system 120 and/or the model training system 110 may be implemented as or included in the system/device 700. The system/device 700 may be a general-purpose computer or computer system, a physical computing system/device, or a portable electronic device, or may be practiced in distributed cloud computing environments where tasks are

performed by remote processing devices that are linked through a communication network. The system/device 700 can be used to implement the process 600 of Fig. 6.

[00115] As depicted, the system/device 700 includes a processor 701 which is capable of performing various processes according to a program stored in a read only memory (ROM) 702 or a program loaded from a storage unit 708 to a random access memory (RAM) 703. In the RAM 703, data required when the processor 701 performs the various processes or the like is also stored as required. The processor 701, the ROM 702 and the RAM 703 are connected to one another via a bus 704. An input/output (I/O) interface 705 is also connected to the bus 704.

[00116] The processor 701 may be of any type suitable to the local technical network and may include one or more of the following: general purpose computers, special purpose computers, microprocessors, digital signal processors (DSPs), graphic processing unit (GPU), co-processors, and processors based on multicore processor architecture, as non-limiting examples. The system/device 700 may have multiple processors, such as an application-specific integrated circuit chip that is slaved in time to a clock which synchronizes the main processor.

[00117] A plurality of components in the system/device 700 are connected to the I/O interface 705, including an input unit 707, such as a keyboard, a mouse, or the like; an output unit 707 including a display such as a cathode ray tube (CRT), a liquid crystal display (LCD), or the like, and a loudspeaker or the like; the storage unit 708, such as disk and optical disk, and the like; and a communication unit 709, such as a network card, a modem, a wireless transceiver, or the like. The communication unit 709 allows the system/device 700 to exchange information/data with other devices via a communication network, such as the Internet, various telecommunication networks, and/or the like.

[00118] The methods and processes described above, such as the process 600, can also be performed by the processor 701. In some embodiments, the process 600 can be implemented as a computer software program or a computer program product tangibly included in the computer readable medium, e.g., storage unit 708. In some embodiments, the computer program can be partially or fully loaded and/or embodied to the system/device 700 via ROM 702 and/or communication unit 709. The computer program includes computer executable instructions that are executed by the associated processor 701. When the computer program

is loaded to RAM 703 and executed by the processor 701, one or more acts of the process 600 described above can be implemented. Alternatively, processor 701 can be configured via any other suitable manners (e.g., by means of firmware) to execute the process 600 in other embodiments.

5 **[00119]** In some example embodiments of the present disclosure, there is provided a computer program product comprising instructions which, when executed by a processor of an apparatus, cause the apparatus to perform steps of any one of the methods described above.

10 **[00120]** In some example embodiments of the present disclosure, there is provided a computer readable medium comprising program instructions for causing an apparatus to perform at least steps of any one of the methods described above. The computer readable medium may be a non-transitory computer readable medium in some embodiments.

15 **[00121]** In an eighth aspect, example embodiments of the present disclosure provide a computer readable medium comprising program instructions for causing an apparatus to perform at least the method in the second aspect described above. The computer readable medium may be a non-transitory computer readable medium in some embodiments.

20 **[00122]** Generally, various example embodiments of the present disclosure may be implemented in hardware or special purpose circuits, software, logic or any combination thereof. Some aspects may be implemented in hardware, while other aspects may be implemented in firmware or software which may be executed by a controller, microprocessor or other computing device. While various aspects of the example embodiments of the present disclosure are illustrated and described as block diagrams, flowcharts, or using some other pictorial representations, it will be appreciated that the blocks, apparatuses, systems, techniques, or methods described herein may be implemented
25 in, as non-limiting examples, hardware, software, firmware, special purpose circuits or logic, general purpose hardware or controller or other computing devices, or some combination thereof.

30 **[00123]** The present disclosure also provides at least one computer program product tangibly stored on a non-transitory computer readable storage medium. The computer program product includes computer-executable instructions, such as those included in program modules, being executed in a device on a target real or virtual processor, to carry

out the methods/processes as described above. Generally, program modules include routines, programs, libraries, objects, classes, components, data structures, or the like that perform particular tasks or implement particular abstract types. The functionality of the program modules may be combined or split between program modules as desired in various embodiments. Computer-executable instructions for program modules may be executed within a local or distributed device. In a distributed device, program modules may be located in both local and remote storage media.

[00124] The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable medium may include but is not limited to an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. More specific examples of the computer readable storage medium would include an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing.

[00125] Computer program code for carrying out methods disclosed herein may be written in any combination of one or more programming languages. The program code may be provided to a processor or controller of a general purpose computer, special purpose computer, or other programmable data processing apparatus, such that the program codes, when executed by the processor or controller, cause the functions/operations specified in the flowcharts and/or block diagrams to be implemented. The program code may execute entirely on a computer, partly on the computer, as a stand-alone software package, partly on the computer and partly on a remote computer or entirely on the remote computer or server. The program code may be distributed on specially-programmed devices which may be generally referred to herein as “modules”. Software component portions of the modules may be written in any computer language and may be a portion of a monolithic code base, or may be developed in more discrete code portions, such as is typical in object-oriented computer languages. In addition, the modules may be distributed across a plurality of computer platforms, servers, terminals, mobile devices and the like. A given module may even be implemented such that the described functions are performed by separate processors

and/or computing hardware platforms.

5 [00126] While operations are depicted in a particular order, this should not be understood as requiring that such operations be performed in the particular order shown or in sequential order, or that all illustrated operations be performed, to achieve desirable results. In certain
circumstances, multitasking and parallel processing may be advantageous. Likewise, while
several specific implementation details are contained in the above discussions, these should
not be construed as limitations on the scope of the present disclosure, but rather as
descriptions of features that may be specific to particular embodiments. Certain features that
are described in the context of separate embodiments may also be implemented in
10 combination in a single embodiment. Conversely, various features that are described in the
context of a single embodiment may also be implemented in multiple embodiments
separately or in any suitable sub-combination.

[00127] Although the present disclosure has been described in languages specific to
structural features and/or methodological acts, it is to be understood that the present
15 disclosure defined in the appended claims is not necessarily limited to the specific features
or acts described above. Rather, the specific features and acts described above are disclosed
as example forms of implementing the claims.

WHAT IS CLAIMED IS:

1. A method comprising:
 - obtaining a feature representation of an input data sample from a prediction model,
5 the prediction model being configured to process input data with a periodicity, the input data sample being a sample of the input data generated at a point of time within a period;
 - determining first Fourier coefficients for a first component in a Fourier expansion by applying the feature representation into a first mapping model, the Fourier expansion being dependent on the point of time and the feature representation, and the Fourier
10 expansion being of the periodicity;
 - determining second Fourier coefficients for a second component in the Fourier expansion by applying the feature representation into a second mapping model;
 - determining a Fourier expansion result based on the first Fourier coefficients and the second Fourier coefficients in the Fourier expansion; and
15 determining a prediction result for the input data sample based on the Fourier expansion result.
2. The method of claim 1, wherein the Fourier expansion comprises a truncated Fourier expansion with a predetermined number of terms, and the number of the first Fourier
20 coefficients and the number of the second Fourier coefficients are based on the predetermined number.
3. The method of claim 2, wherein the first component is based on a sine function dependent on the point of time and having the periodicity, and the second component is
25 based on a cosine function with the periodicity, and wherein determining the Fourier expansion result comprises:
 - determining a set of first component values for the first component by shifting a frequency of the sine function for the predetermined number of times;
 - determining a set of second component values for the second component by shifting
30 a frequency of the cosine function for the predetermined number of times; and
 - determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values.

4. The method of claim 3, wherein determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values comprises:

5 calculating first products by multiplying the first Fourier coefficients with the first component values;

calculating second products by multiplying the second Fourier coefficients with the second component values;

10 mapping the first products to a first intermediate expansion result using a third mapping model, and mapping the second products to a second intermediate expansion result using a fourth mapping model; and

determining the Fourier expansion result by aggregating the first intermediate expansion result and the second intermediate expansion result.

15 5. The method of any of claims 1 to 4, wherein determining the prediction result for the input data sample based on the Fourier expansion result comprises:

determining a first intermediate prediction result from the Fourier expansion result;

obtaining a second intermediate prediction result generated from an output layer of the prediction model based on the feature representation; and

20 determining the prediction result by aggregating the first intermediate prediction result and the second intermediate prediction result.

6. The method of any of claims 1 to 5, wherein the prediction model comprises a plurality of sub-models configured to extract a plurality of feature representations from the input data sample, and wherein obtaining the feature representation comprises:

25 obtaining the plurality of feature representations from the plurality of sub-models;

generating the feature representation by aggregating the plurality of feature representations.

30 7. The method of any of claims 1 to 6, wherein the first mapping model and the second mapping model are constructed without activation functions.

8. A system, comprising:

at least one processor; and

at least one memory communicatively coupled to the at least one processor and comprising computer-readable instructions that upon execution by the at least one processor cause the at least one processor to perform acts comprising:

5 obtaining a feature representation of an input data sample from a prediction model, the prediction model being configured to process input data with a periodicity, the input data sample being a sample of the input data generated at a point of time within a period;

10 determining first Fourier coefficients for a first component in a Fourier expansion by applying the feature representation into a first mapping model, the Fourier expansion being dependent on the point of time and the feature representation, and the Fourier expansion being of the periodicity;

determining second Fourier coefficients for a second component in the Fourier expansion by applying the feature representation into a second mapping model;

15 determining a Fourier expansion result based on the first Fourier coefficients and the second Fourier coefficients in the Fourier expansion; and

determining a prediction result for the input data sample based on the Fourier expansion result.

20 9. The system of claim 8, wherein the Fourier expansion comprises a truncated Fourier expansion with a predetermined number of terms, and the number of the first Fourier coefficients and the number of the second Fourier coefficients are based on the predetermined number.

25 10. The system of claim 9, wherein the first component is based on a sine function dependent on the point of time and having the periodicity, and the second component is based on a cosine function with the periodicity, and wherein determining the Fourier expansion result comprises:

30 determining a set of first component values for the first component by shifting a frequency of the sine function for the predetermined number of times;

determining a set of second component values for the second component by shifting a frequency of the cosine function for the predetermined number of times; and

determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values.

5 11. The system of claim 10, wherein determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values comprises:

calculating first products by multiplying the first Fourier coefficients with the first component values;

10 calculating second products by multiplying the second Fourier coefficients with the second component values;

mapping the first products to a first intermediate expansion result using a third mapping model, and mapping the second products to a second intermediate expansion result using a fourth mapping model; and

15 determining the Fourier expansion result by aggregating the first intermediate expansion result and the second intermediate expansion result.

12. The system of any of claims 8 to 11, wherein determining the prediction result for the input data sample based on the Fourier expansion result comprises:

20 determining a first intermediate prediction result from the Fourier expansion result; obtaining a second intermediate prediction result generated from an output layer of the prediction model based on the feature representation; and

determining the prediction result by aggregating the first intermediate prediction result and the second intermediate prediction result.

25 13. The system of any of claims 8 to 12, wherein the prediction model comprises a plurality of sub-models configured to extract a plurality of feature representations from the input data sample, and wherein obtaining the feature representation comprises:

obtaining the plurality of feature representations from the plurality of sub-models;

30 generating the feature representation by aggregating the plurality of feature representations.

14. The system of any of claims 8 to 13, wherein the first mapping model and the second mapping model are constructed without activation functions.

5 15. A non-transitory computer-readable storage medium, storing computer-readable instructions that upon execution by a computing device cause the computing device to perform acts comprising:

obtaining a feature representation of an input data sample from a prediction model, the prediction model being configured to process input data with a periodicity, the input data sample being a sample of the input data generated at a point of time within a period;

10 determining first Fourier coefficients for a first component in a Fourier expansion by applying the feature representation into a first mapping model, the Fourier expansion being dependent on the point of time and the feature representation, and the Fourier expansion being of the periodicity;

15 determining second Fourier coefficients for a second component in the Fourier expansion by applying the feature representation into a second mapping model;

determining a Fourier expansion result based on the first Fourier coefficients and the second Fourier coefficients in the Fourier expansion; and

determining a prediction result for the input data sample based on the Fourier expansion result.

20 16. The non-transitory computer-readable storage medium of claim 15, wherein the Fourier expansion comprises a truncated Fourier expansion with a predetermined number of terms, and the number of the first Fourier coefficients and the number of the second Fourier coefficients are based on the predetermined number.

25 17. The non-transitory computer-readable storage medium of claim 16, wherein the first component is based on a sine function dependent on the point of time and having the periodicity, and the second component is based on a cosine function with the periodicity, and wherein determining the Fourier expansion result comprises:

30 determining a set of first component values for the first component by shifting a frequency of the sine function for the predetermined number of times;

determining a set of second component values for the second component by shifting a frequency of the cosine function for the predetermined number of times; and

determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values.

5 18. The non-transitory computer-readable storage medium of claim 17, wherein determining the Fourier expansion result by multiplying the first Fourier coefficients with the first component values, respectively, and multiplying the second Fourier coefficients with the second component values comprises:

10 calculating first products by multiplying the first Fourier coefficients with the first component values;

 calculating second products by multiplying the second Fourier coefficients with the second component values;

15 mapping the first products to a first intermediate expansion result using a third mapping model, and mapping the second products to a second intermediate expansion result using a fourth mapping model; and

 determining the Fourier expansion result by aggregating the first intermediate expansion result and the second intermediate expansion result.

20 19. The non-transitory computer-readable storage medium of any of claims 15 to 18, wherein determining the prediction result for the input data sample based on the Fourier expansion result comprises:

 determining a first intermediate prediction result from the Fourier expansion result;

 obtaining a second intermediate prediction result generated from an output layer of the prediction model based on the feature representation; and

25 determining the prediction result by aggregating the first intermediate prediction result and the second intermediate prediction result.

30 20. The non-transitory computer-readable storage medium of any of claims 15 to 19, wherein the prediction model comprises a plurality of sub-models configured to extract a plurality of feature representations from the input data sample, and wherein obtaining the feature representation comprises:

 obtaining the plurality of feature representations from the plurality of sub-models;

generating the feature representation by aggregating the plurality of feature representations.