Num2vec: Pre-Training Numeric Representations for Time Series Forecasting in the Sensing System

JINXIAO FAN, PENGFEI WANG, YU FAN, LIANG LIU, and HUADONG MA, Beijing University Posts and Telecommunications, China

Time series forecasting in the sensing system aims to predict future values based on historical records that sensors have collected. Previous works, however, usually focus on improving model structure or algorithm for better performance but the perspective of learning proper numeric representations is overlooked. The inappropriate and coarse numeric representations are not expressive enough to capture the intrinsic characteristics of numbers, which will obviously degrade the prediction performance.

In this article, we propose Num2vec, an algorithmic framework to learn numeric representations. Specifically, Num2vec lists three main logic characteristics of numbers: arithmetic, direction, and periodicity. By representing numbers into a transition space, Num2vec can translates numbers agilely to different Internet of Things tasks through selecting the corresponding characteristics. According to such a design, Num2vec enjoys flexible numeric representations to fit different Internet of Things time series tasks. Extensive experiments on four real-world datasets show that the approach achieves the best performance when compared with state-of-the-art baselines.

CCS Concepts: • **Information systems** → **Sensor networks**; • **Computing methodologies** → **Artificial intelligence;**

Additional Key Words and Phrases: Time series forecasting, representation learning, IoT, sensing system

ACM Reference format:
Jinxiao Fan, Pengfei Wang, Yu Fan, Liang Liu, and Huadong Ma. 2023. Num2vec: Pre-Training Numeric Representations for Time Series Forecasting in the Sensing System. *ACM Trans. Sensor Netw.* 19, 4, Article 94 (July 2023), 23 pages.
https://doi.org/10.1145/3599728

1 INTRODUCTION

Benefiting from the explosive deployment of **Internet of Things (IoT)** devices, a massive amount of time series data now is continuously generated and in turn contributes tremendous practical value to different sensing system, such as air quality [12, 32], intelligent transportation [10], and human mobility [33], among others. In this specific scenario, single-value data collected from sensors account for a large proportion. Considering their unique status, it is necessary to capture the

This work was supported in part by the A3 Foresight Program of NSFC (grant 62061146002), and the Funds for Creative Research Groups of China (grant 61921003).
Authors' address: J. Fan, P. Wang, Y. Fan, L. Liu (corresponding author), and H. Ma, Beijing University Posts and Telecommunications, Beijing, China; emails: jinxiaofan@bupt.edu.cn, wangpengfei@bupt.edu.cn, fany@bupt.edu.cn, liangliu@bupt.edu.cn, mhd@bupt.edu.cn.

1550-4859/2023/07-ART94 $15.00
https://doi.org/10.1145/3599728

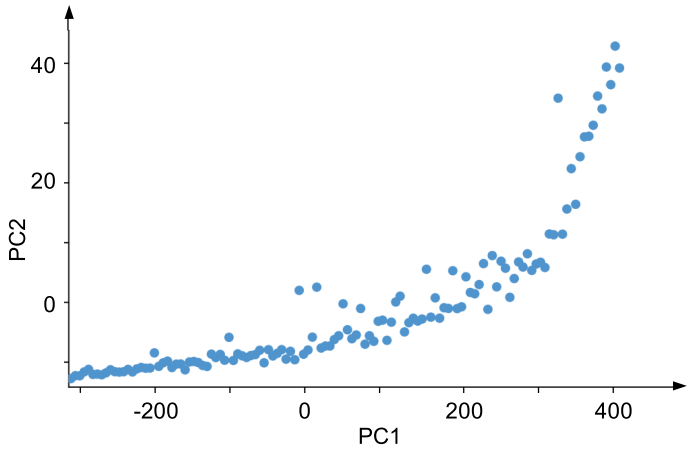

Fig. 1. Arithmetic. The blue dots show that the arithmetic relationships actually weaken in the intermediate process of model.

characteristics of numeric data for effective time series forecasting. However, previous works usually overlooked this perspective but focused on improving the model structure or algorithm for better performance. The numeric data is thus lacking in-depth modeling to capture their intrinsic characteristics, and making the prediction performance inevitably degraded.

Consequently, in this article, we try to answer the following question: *How to model these characteristics over numbers with representation learning?* This has rarely been noticed before, because different numeric data is arising from different scenarios where their intrinsic characteristics vary widely. For instance, the essential implications of temperature value "20 °C" and direction value "20°" are not equivalent. This means that multiplex feature relationships cannot be modeled uniformly.

But, it is surprising that traditional methods ignore these inherent numerical characteristics since the oversimplified numbers are always ideally thought to be recognized and accepted perfectly by the model [17, 22, 26, 30]. However, numbers lack sufficiently clear structure and distinct correlation, in contrast to text data in neural linguistic programming [9, 23–25].

Given the variety of numerical data in IoT time series applications, in this work we define three different characteristics (i.e., arithmetic, direction, and periodicity), which cover almost all numeric characteristics among time series measured by sensors.

However, it remains a great challenge to capture corresponding characteristics based on the different target sequences—specifically, we take air quality monitoring as an example to illustrate. First, the *arithmetic*, as a basic property, can reflect the magnitude relationship of numbers. But the neglected arithmetic becomes utterly absent after undergoing the high-dimensional transformation in models. Figure 1 reflects the nonlinear relationships of vectors in model's intermediate process by PCA [35]. Second, the *direction* is generally rendered as angle values, which carry orientation but no magnitude relationship. For instance, "359°" and "0°" differ by 359 in numerical values, but 1 in angular values. So, they have almost the same influence on the target variable (e.g., the influence of wind direction on $PM_{2.5}$ concentration as shown in Figure 2. Third, the *periodicity* involves an obvious seasonal pattern or day-periodic characteristic. Traditional temporal properties capturing methods generally manipulate the timestamps [41], which compels various variables to share equivalent periodic constraints. Again, the behavior of periodicity is not globally the same (i.e., month-periodic or even day-periodic), as shown in Figure 3. It can clearly be seen that data obtained from different sensors traditionally only requires us to pay attention to the

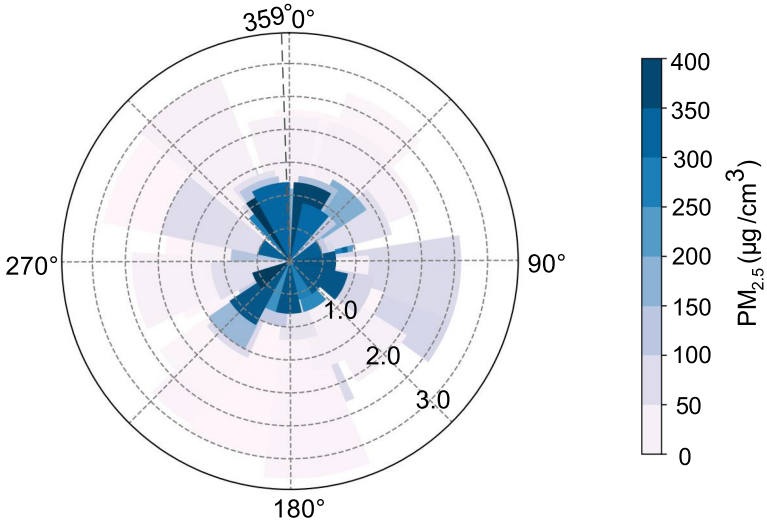

Fig. 2. Direction. The radar chart describes the impact of wind on $PM_{2.5}$ concentration in March, where angle and radius indicate wind direction and wind speed, respectively, and the depth of color indicates the concentration of $PM_{2.5}$. It can be observed that adjacent angles have similar effects.

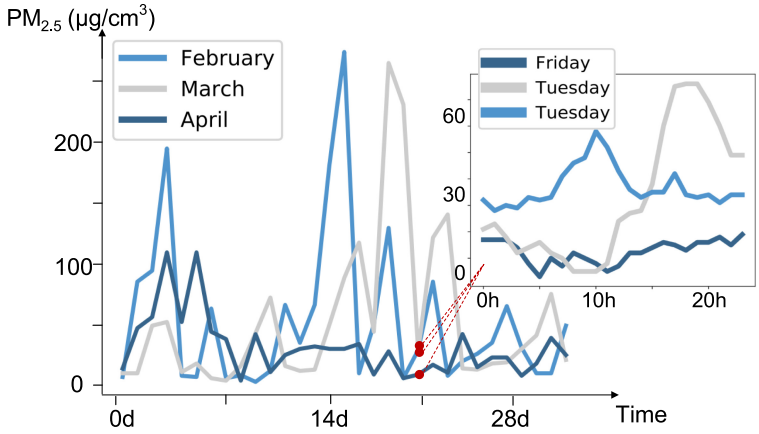

Fig. 3. Periodicity. The main part shows month-periodic of $PM_{2.5}$ in February, March, and April, and the part indicated by three red dots shows day-periodic on the last day of the third week. Although the average value of $PM_{2.5}$ for the 3 days is close, the change at each moment is not regular.

magnitude relationship between values. However, during data preprocessing, numbers with the same numerical value but different implications tend to be the same after standardization. Again, during model training, the logical characteristics of intra-sequence from temperature or wind direction values may be corrupted after high-dimensional transformation. Therefore, it is necessary to learn representations of different numerical values and constrain model training based on different types of values in IoT time series data.

With regard to this, we are supposed to model a pre-training framework for explicitly extracting three characteristics respectively by numeric representations. Here we propose Num2vec, a pre-training numeric representation learning method, which imposes potential arithmetic, direction,

and periodicity in virtue of the numbers' realistic characteristics. In short, our method explicitly (i) represents numbers as points in a (high-dimensional) "transition space" and helps these existing characteristics reduce the damage caused by intermediate transformation, and (ii) three logic characteristics' numeric representation results cooperated with the original values are applied to the downstream predictor. Ultimately, our method helps extend potential characteristics to models' parameter space and preserve essential knowledge of the numeric sequences in high-dimensional transition vectors. As a result, we improve the model prediction accuracy and make these numeric representations interpretable. In summary, the contribution of this article lies in three aspects:

- Compared with the traditional IoT time series forecast task, we focus on learning potential logic characteristics. To the best of our knowledge, this is the first study to improve prediction performance from the perspective of modeling numeric representations.
- To meet the preceding goal, we propose Num2vec, a pre-training numeric representations of learning intrinsic arithmetic, direction, and periodicity. It is clearly beneficial to extract and maintain these potential characteristics in the connected downstream models.
- We conduct extensive experiments on four real-world datasets, and the proposed method helps the model achieve better forecast performance. Moreover, the additional representation processing increases the interpretability of task.

The rest of the article is organized as follows. Section 2 reviews the related work about IoT time series forecast and representation learning. We present an overview in Section 3. Section 4 proposes three intuitions and presents corresponding numeric representations to improve forecasting performance. Section 5 evaluates the proposed method by extensive experiments over datasets and analyzes representation results quantitatively. Section 6 concludes this article.

2 RELATED WORK

2.1 Time Series Forecasting

Time series forecasting has been an emerging topic in machine learning, which can be broadly divided into two categories from the perspective of methodology. The first category is model-driven approaches. It is also well known as parametric approaches, which are generally predetermined based on strong theoretical assumptions [14]. However, these assumptions are hardly satisfied in practice, thus limiting their forecasting performance. The previous methods, such as ARIMA [40], can only learn linear relationships among different timesteps, which have an inherent deficiency in fitting many real-world time series data that are highly nonlinear. The second category is data-driven approaches, which requires us to start from various data perspectives to fill the knowledge gaps in the model. Traditional machine learning methods, such as KNN [42, 45], and PCA [43, 44] heavily rely on handcrafted features with expert experience. Consequently, these methods are not well-suited to analyzing real-world high-dimensional time series datasets as they require significant amounts of prior knowledge and are difficult to learn features autonomously from raw data. Recently, deep learning has made it possible to effectively model high-dimensional data, which helps us automatically discover hierarchical features but comes at the expense of memory. For example, LSTnet [17] combines CNNs and RNNs to capture short-term local dependencies and long-term trends simultaneously. TPA-LSTM [26] employs a novel attention mechanism named *temporal pattern attention* to capture temporal patterns across multiple timesteps. Informer [41] tackles the efficiency problem of high-complexity self-attention and shows remarkable performance on forecasting tasks with long sequences. Differently, our work predicts time series tasks from the perspective of the data driven approach. However, unlike the preceding approaches, we pay more attention to the underlying logic-characteristic representation capacity of raw numeric data.

2.2 Representation Learning

Representation learning is the effective operating of constructing explanatory features that can be used for classification or prediction problems, which is generally classified into two categories based on learning methods. The first is end-to-end methods. They learn for a single task as supervised learning, which is supposed to model complex and diverse features through some deep learning approaches [13]. The second is pre-training methods. They obtain general embedding vectors or representations from the unsupervised or self-supervised objects for different downstream tasks. These methods not only help data establish more complete feature representation but also contribute to a variety of downstream tasks and practical applications [39]. Some word embedding methods in neural linguistic programming [8, 9, 21, 24], node embedding methods in graph representations [4], and entity/relationship embedding methods in knowledge graph [2, 31] can be commonly applied in succession. In recent years, this method has also attracted much attention in spatial-temporal data processing [20], so as to help model the spatiotemporal properties of data. This indirectly explains that representation learning began to enter the pure digital data processing. Benefiting from in-depth research on knowledge graph representation, our work applies pre-training representations to time series in the sensing system.

But note that the application of representation learning in time series does not reach deeper to the logic characteristics. Although traditionally most of the actual machine learning research has focused on feature engineering or the design of pre-processing data transformation [1], the intrinsic characteristics of numeric data are not fully represented into models [34]. Additionally, the performance of the model heavily depends on data representation or features. Therefore, it strongly motivates the adoption of a recent trend in time series forecasting toward utilization of numeric representation—Num2vec proposed in this article. It helps the original numeric sequence learn potential logic representations so that data with more knowledge can better suit the downstream forecast work to improve performance.

3 OVERVIEW

To model the potential characteristics of original numerical dataand simultaneously enjoy the generalization benefits of IoT time series forecasting, in this article we do not choose to design a targeted end-to-end predictive model but shift to learning generic pre-training numeric representations. Starting from a logic characteristics point of view, this work extracts arithmetic, direction, and periodicity, which are sufficient to represent numerical characteristics of most IoT time series. Therefore, our Num2vec helps the downstream predictor incorporate prior knowledge of numerical intrinsic characteristics and capture pivotal features of potential relationships.

The intuitions are as follows:

- *Intuition 1—Arithmetic extraction*: To enhance the arithmetic properties for time series, we consider assigning addition and subtraction operations as relationships, thus capturing the magnitude similarity between numbers.
- *Intuition 2—Direction extraction*: To append the directional attribute to the angle values in the sequence, we design interval and opposite relationship for 361 discrete values to distinguish them from arithmetic.
- *Intuition 3—Periodicity extraction*: Due to the uncertainty of the subsequence (e.g., month-periodic and day-periodic as shown in Figure 3), we capture periodic characteristic through constraints on periodic functions dynamically.

3.1 Problem Formulation

Time series in sensing systems generally consist of multiple nonpredictive time series and a target series. The time series data $\mathcal{X} = \{x_1, x_2, \ldots, x_\tau\}$ denotes historical observations in τ time window

size, where each x_i represents multivariate variables. The forecasting goal is to predict the target observation at the next moment $\mathcal{Y} = \{x_{\tau+1}\}$. In general, traditional methods are usually abstracted as a mapping from input to output $\mathcal{X} \rightarrow \mathcal{Y}$, which requires us to learn the mapping function better to realize performance benefits. Different from this, this work is designed to learn numeric representations based on arithmetic, direction, and periodicity, which help each original data x_i append corresponding prior characteristic knowledge. Therefore, we further learn numerical potential representation $\mathcal{R}$ to assist in learning the preceding mapping in the pre-training phase of the prediction model. The new mapping is described as follows:

$$\{\mathcal{X}; \mathcal{R}\} \rightarrow \mathcal{Y}.$$

3.2 Framework Overview

To realize the preceding pre-training numeric representations, we build a module in two steps. The first step is naturally capturing potential characteristics according to the logical relationship of values. Concretely, we analyze the input data $\mathcal{X}$ in view of three numerical characteristics, and we model the underlying arithmetic, direction, and periodicity by numeric representations, as shown in module 2 of Figure 4. This step helps extract the underlying three logic characteristics and convert them into high-dimension transition vectors as additional constraints of original sequences. The second step is easily fusing the three representations and original data to fit various downstream models, as shown in module 3 of Figure 4. We integrate standardized raw data as well as learned numeric representations and then merge them into the forecaster. It not only benefits data to maintain their intrinsic relationships in higher-dimensional space but also provides better performance for the time series forecasting task. Furthermore, appending these more interpretable characteristics to numbers can scale to large, real-world datasets, which is compatible with different IoT general tasks.

4 METHODOLOGY

In this section, we describe our implementation method Num2vec in the numeric representations module in Figure 4. Due to the overlapping of periodic data with arithmetic and direction, we design two approaches. We first discuss how to model the first two characteristics in Section 4.1. Then, we explore how to extract periodicity in Section 4.2. Finally, we discuss how to combine the learned multi-characteristic with original data for downstream prediction in Section 4.3.

4.1 Translation-Based Numeric Representations

To attach underlying arithmetic and direction characteristics (relationships) between numbers (entities), we exploit translation-based methods to model numeric representations. It applies these numbers to a metric space with a novel translation-based structure, which enjoys the benefits of using a single, interpretable component.

4.1.1 Model Numeric Characteristics. Recall that the time series is composed of a sequence of scalar numbers that carry different relationships and characteristics naturally, and we decide to model the relationships between the values structurally. In the field of knowledge graph embedding [6, 7, 29], they generally employ triples (head, relation, tail) to represent knowledge. By constructing a graph where nodes are entities (e.g., head, tail) and edges are relations, it learns a representation of structured relational information. Specifically, transE [2] and transH [31] are the translation-based knowledge graph embedding models that use distance-based measures to generate the similarity score for a pair of entities and their relationships. Additionally, they aim to find a vector representation of entities in relation to the translation of the entities. Inspired by

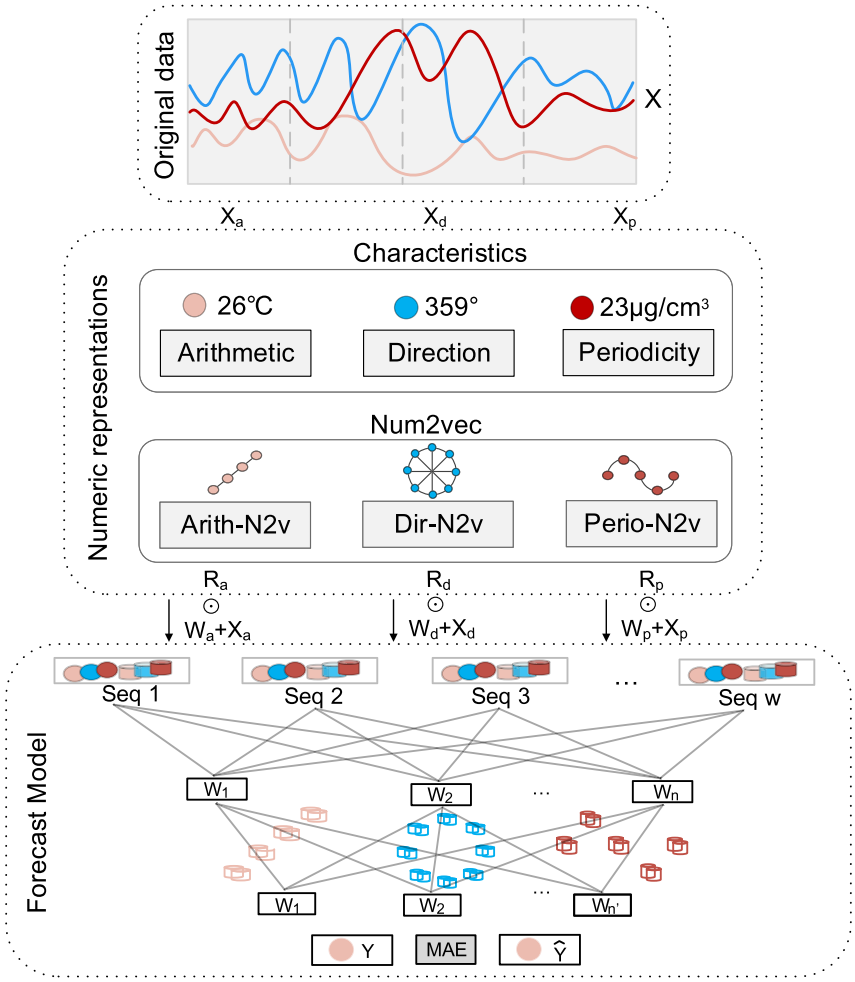

Fig. 4. Time series forecast framework with numeric representation. Module 1: Multivariate sequences as inputs. Module 2: Model potential arithmetic, direction, and periodicity characteristics over corresponding numbers and map to transition vectors $\mathcal{R}_a, \mathcal{R}_d, \mathcal{R}_p$ by additional numeric representation. Module 3: Fusing three numeric representations and original data into the downstream forecast model.

them, Num2vec designed two targeted algorithm modules, Arith-N2v and Dir-N2v, by interpreting multi-relation numbers as high-dimensional vectors to model the potential numeric features.

Arithmetic Characteristic. In view of the arithmetic characteristic, Num2vec converts directly to addition and subtraction operations between numbers, which not only can extract the relationship between numbers but also helps the sequence learn the difference between two timesteps. Methodologically, we learn a transition space $\Psi_a = R^k$, where each of the numbers $i \in E_a$ is represented with a vector $\boldsymbol{i} \in \Psi_a$. To model the arithmetic characteristic, we represent addition relationships r_+ and subtraction r_- relationships with two translation vectors $\boldsymbol{r}_+, \boldsymbol{r}_- \in \Psi_a$ to capture the number's inherent correlation. For instance, after we determine the two values i, j and the relationship r, it should be guaranteed that there is a unique value m matching it, and what we want is

$$\boldsymbol{i} + \boldsymbol{r}_+ + \boldsymbol{m} = \boldsymbol{j}, \tag{1}$$

ALGORITHM 1: Arith-N2v

Input:
Numbers and relations sets $E_a \supset \mathcal{X}_a, L_a$, correct quads $S_a = \{s|s = (i, r, m, j)\}$, incorrect quads $S'_a = \{s'|s' = (i', r, m', j')\}$, margin η, representation dimension k_a.

Output:
Arithmetic numeric representations on number sets $\mathcal{R}_a$.

initialize $r, x \leftarrow \text{uniform}(-\frac{10}{\sqrt{k_a}}, \frac{10}{\sqrt{k_a}}),\ r \leftarrow \frac{r}{\|r\|}$, where $r \in L_a,\ x \in E_a$.
loop
  $x \leftarrow \frac{x}{\|x\|}$ for each $x \in E_a$.
  $S_{batch} \leftarrow$ sample a mini batch from S_a.
  $Q_{batch} \leftarrow$ initialize the set of quads as empty set.
  for $s \in S_{batch}$ **do**
    $s' \leftarrow$ sample an incorrect quad from S'_a.
    $Q_{batch} \leftarrow Q_{batch} \cup \{s, s'\}$.
  end for
  $\mathcal{R}_a \leftarrow$ Update representations w.r.t. $\sum [\eta + \Delta D_q]_+$
end loop
return $\mathcal{R}_a \leftarrow x\mathcal{R}_a$ to enhance linear correlation.

which means that $\boldsymbol{j}$ should be the nearest neighbor of $\boldsymbol{i} + \boldsymbol{r}_+ + \boldsymbol{m}$ in Ψ according to some distance metric (e.g., squared Euclidean distance). Additionally, the metric formula applied for a quad $(\boldsymbol{i}, \boldsymbol{r}, \boldsymbol{m}, \boldsymbol{j})$ is denoted as $d(\boldsymbol{i} + \boldsymbol{r} + \boldsymbol{m}, \boldsymbol{j})$ for convenience, and the comparison with traditional triples is shown in Section 5.4.1. In this article, we optimize the following margin-based ranking loss to train our methods:

$$\mathcal{L}_a = \sum [\eta + \Delta D_q]_+, \tag{2}$$

where margin η makes the summation always positive (i.e., $[x]_+ = max(0, x)$), which is equivalent to the interval correction between the two quads. ΔD_q denotes $d(\boldsymbol{i} + \boldsymbol{r} + \boldsymbol{m}, \boldsymbol{j}) - d(\boldsymbol{i'} + \boldsymbol{r} + \boldsymbol{m'}, \boldsymbol{j'})$, whose former and latter items respectively sample from correct quads S_a and incorrect quads S'_a. The incorrect quads are formed by replacing one of the numbers with another random number and fixed other letters, so S'_a is constructed as follows:

$$\begin{aligned} S'_a = &\{(i', r, m, j) \mid i' \in E_a\} \cup \\ &\{(i, r, m', j) \mid m' \in E_a\} \cup \\ &\{(i, r, m, j') \mid j' \in E_a\}. \end{aligned} \tag{3}$$

Note that we uncover such a metric space where (i) the quads capture a numeric magnitude concept of similarity by Equation (1), and (ii) this translation that interprets numbers as high-dimensional vectors encapsulates increase/decrease relationships. The detailed training procedure for modeling arithmetic numeric representations is described in Algorithm 1. After training and learning such numeric representations, the corresponding sequence $\mathcal{X}_a$ is attached to arithmetic characteristic $\mathcal{R}_a$. Moreover, we multiply the original value for vectors of each dimension to enhance linear correlation as the final representation results.

Direction Characteristic. Different from the arithmetic characteristic, direction depicts the orientation property of the values rather than the magnitude. Here, Dir-N2v captures the interval and opposite relationship among numbers to model directional characteristics.

On the one hand, the two relations are involved in reflexive problems due to the particularity of direction angles. First, interval relationship r_i can be regarded as the arithmetic characteristic with the addition of cycle properties and the subtraction of magnitude attributes. Likewise, we employ quads to denote the interval relationship while (i, r_i, m, j) is equivalent to (j, r_i, m, i). Again, angular numbers are constrained by the opposite relationship r_o, which refers to the inverse orientation between pairs of directions. Therefore, we employ general triangle inequality to extract this relationship, whereas the triples are appended symmetry properties (i.e., (h, r_o, t) is equivalent to (t, r_o, h)). On the other hand, we are supposed to overcome many-to-one/one-to-many problems in relations because "0" and "360" indicate the same significance.

Therefore, Dir-N2v enables a number to have distributed representations based on their involvement in different relationships. For the purpose of avoiding the proximity of two/three numbers and ensuring the realization of multi-relation during model training, the angular number $i \in E_d$ is first projected in a relation-specific hyperplane w_r, which is denoted as $\boldsymbol{i}_\perp$, and the corresponding relationship is denoted as $\boldsymbol{r}^w$. So, this projection is defined as

$$\boldsymbol{i}_\perp = \boldsymbol{i} - w_r^\top \boldsymbol{i} w_r.$$

Moreover, the objective function consists of the following margin-based hinge loss:

$$\mathcal{L}_d = \sum [\eta + \Delta D_q + \Delta D_t]_+ + \gamma f. \tag{4}$$

Here, the first part is similar to Equation (2) but supplements the measurement computing of triples, of which ΔD_q and ΔD_t calculate separately the loss about the quads extracting interval relationship and triples extracting opposite relationship. So, it is measured in two tuples according to direction relationships (i.e., interval-quad set S_i and opposite-triple set S_o). Specifically, ΔD_t denotes $d(\boldsymbol{h}_\perp + \boldsymbol{r}^w, \boldsymbol{t}_\perp) - d(\boldsymbol{h}'_\perp + \boldsymbol{r}^w, \boldsymbol{t}'_\perp)$ where the corresponding incorrect triples $(h', r_o, t') \in S'_o$. S'_o constructs as followed:

$$\begin{aligned} S'_o = &\{(h', r_o, t) \mid h' \in E_d\} \cup \\ &\{(h, r_o, t') \mid t' \in E_d\}, \end{aligned} \tag{5}$$

where $h'_\perp, t'_\perp$ are angle entities h', t' projected to the hyperplane. Different from Equation (2), ΔD_q denotes $d(\boldsymbol{i}_\perp + \boldsymbol{r}^w + \boldsymbol{m}_\perp, \boldsymbol{j}_\perp) - d(\boldsymbol{i}'_\perp + \boldsymbol{r}^w + \boldsymbol{m}'_\perp, \boldsymbol{j}'_\perp)$, where the corresponding incorrect quads are $(i', r_i, m', j') \in S'_i$. Additionally, the incorrect quads S'_i can be obtained in the same way according to Equation (2). Again, the second part of Equation (2) is soft constraints for the score function [31]. γ is a hyper-parameter to weigh the importance of this part. As well, f consists of two parts as a constraint item: one part f_1 restricts that all numbers are normalized, and another part f_2 ensures that r_w exists on the hyperplane. Specifically, f_1 and f_2 are designed as follows:

$$\begin{cases} f_1 = \sum\limits_{x \in E_d} [\|x\|_2^2 - 1]_+, \\ f_2 = \sum\limits_{r_i, r_o} \left[\dfrac{(w_r^\top \boldsymbol{r}^w)^2}{\|\boldsymbol{r}^w\|_2^2} - \varepsilon \right]_+. \end{cases} \tag{6}$$

In both of the preceding cases, the quads and triples play an important role in helping the model generalize vectors that better represent the realistic direction attributes, as it does in canonical metric learning scenarios. For instance, the gap between "359°" and "0°" is very small, and Dir-N2v will also put "359°" close to "0°". In contrast, all angular numbers with an opposite relationship have the largest gap. Other training details are described in Algorithm 2. In addition to employing

ALGORITHM 2: Dir-N2v

Input:

Numbers and relations sets $E_d \supset \mathcal{X}_d$, L_d, training set $S_i = \{s_i | s_i = (i, r_i, m, j)\}$, $S_o = \{s_o | s_o = (h, r_o, t)\}$, incorrect set $S_i' = \{s_i' | s_i' = (i', r_i, m', j')\}, S_o' = \{s_o' | s_o' = (h', r_o, t')\}$, margin η, weight γ, representation dimension k_d.

Output:

Direction numeric representations on angular number sets $\mathcal{R}_d$.

initialize $r, x \leftarrow \text{uniform}(-\frac{10}{\sqrt{k_d}}, \frac{10}{\sqrt{k_d}})$, $r \leftarrow \frac{r}{\|r\|}$,
where $r \in L_d$, $x \in E_d$.
loop
  $x \leftarrow \frac{x}{\|x\|}$ for each $x \in E_d$.
  $S_{batch} \leftarrow$ sample a batch from S_i, S_o respectively.
  $Q_{batch} \leftarrow$ initialize the set of quads and triples.
  for $s_i, s_o \in S_{batch}$ **do**
    $s_i', s_r' \leftarrow$ sample two corrupted sets respectively.
    $s_i, s_i', s_o, s_o' \leftarrow$ update sets by parameter w_r w.r.t.
      $$\boldsymbol{i}_\perp = \boldsymbol{i} - w_r^\top \boldsymbol{i} w_r$$
    $Q_{batch} \leftarrow Q_{batch} \cup \{s_i, s_i', s_o, s_o'\}$.
  end for
  $\mathcal{R}_d \leftarrow$ Update representation w.r.t.
    $$\mathcal{L}_d = \sum [\eta + \Delta D_q + \Delta D_t]_+ + \gamma f$$
end loop

categorical orientations to indicate the surrounding direction [11, 18, 38] (e.g., [E, S, W, N, SE, SW, NE, NW], representing eight directions of east, south, west, north, southeast, southwest, northeast, and northwest, respectively), we deal with it similarly applying Dir-N2v. Each orientation value is regarded as a discrete value, and the interval and opposite relationships are used to constrain it. Especially, Dir-N2v can also apply to other directional representations by changing the training entity set and the settings of some parameter variables.

Note that we guarantee that all numbers within the training range serve for pre-training sets. Therefore, all datasets should belong to training as well as testing, which ensures not only the accuracy of representation performance but also the correspondence between pre-training output and downstream input. Two algorithms iterate the training process continuously by minimizing margin-based ranking loss according to Equations (2) and (4). Certainly, the quality of final pre-training representations is measured by the downstream prediction performance.

4.1.2 Extract Numeric Knowledge. Modeling arithmetic and direction characteristics is inspired by knowledge graph embedding [2, 19, 28, 31, 36], where the optimization objective is to learn multiple relations between pairs of entities. Here, we use the idea of knowledge extraction to facilitate learning diverse numerical representations in time-series data.

By abstracting and converting the information according to the sequences in given time series applications, the potentially carried characteristics are captured by the corresponding high-dimension translation operation. Therefore, it cannot only embody in enriching numeric characteristics but maintain these features in the intermediate high-dimension space.

On the one hand, we exploit Arith-N2v to capture the close association between time series within a fixed range. For instance, extracting additive relation among "2," "23," and "25," which

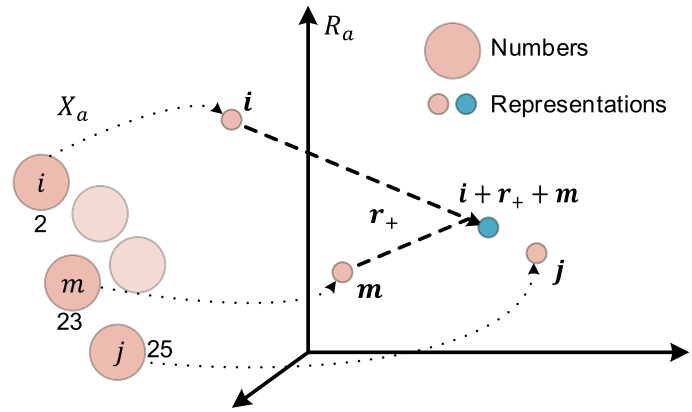

Fig. 5. Extracting addition relation of arithmetic knowledge.

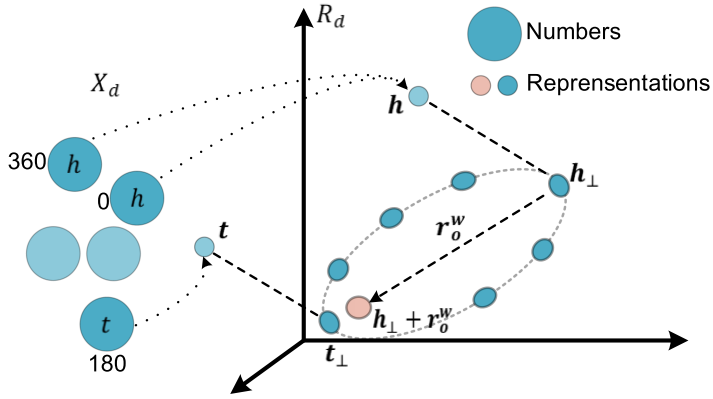

Fig. 6. Extracting opposite relation of direction knowledge.

translates them into high-dimensional vectors and measures the distance between converted "2+23" and "25" in this relation, as shown in Figure 5. It guarantees that arbitrary two numbers with one relationship (r_+ or r_-) are determined, and there is another unique number corresponding to them, which is also in line with the uniqueness of arithmetic operations. Up to this point, arithmetic knowledge between numbers can be learned and preserved explicitly in downstream models. On the other hand, we exploit Dir-N2v to capture a directional relationship among angular numbers ranging from 0 to 360. "0°" and "360°" denote the same direction, so they should learn similar representation after inputing the model in order to carry related information and perform the corresponding functions. Specifically, the opposite relationship r_o takes along symmetry properties (e.g., "0°/180°" is opposite to "180°/0°"), as shown in Figure 6. Dir-N2v regards angular numbers as discrete values for the purpose of breaking invalid arithmetic properties of original numbers. Additionally, it extracts direction knowledge from high-dimensional translation vectors R_d, thus preserving for downstream tasks.

4.2 Periodicity Numeric Representations

In the IoT time series forecasting task, some variables can occur periodically (e.g., the weather in different seasons), but this regular pattern only reflects the rough trend. Other variables possess

ALGORITHM 3: Perio-N2v

Input:
Training set $\mathcal{X}_p = \{x_j\}_{j=1}^{n}$, sequence length q, representation dimension k_p.

Output:
Time representations $\mathcal{R}_p$.

initialize parameters $\mathcal{W}, b \leftarrow \text{uniform}(-\frac{2}{\sqrt{k_p}}, \frac{2}{\sqrt{k_p}})$. Periodic activation function $\mathcal{T} = cos$.
loop
  $S_{batch} \leftarrow$ sample a mini batch from $\mathbf{X}_p$.
  $T_{batch} \leftarrow$ initialize the sequence with q time points.
  for $x \in T_{batch}$ **do**
    $\mathcal{R}_p = \mathcal{T}(\mathcal{W}_i x + b_i)$, if $0 \leq i < k_p$
    Input features $T_{batch} \leftarrow T_{batch} \cup \mathcal{R}_p$
  end for
  $\mathcal{R}_p \leftarrow$ Update representations according to MAE loss.
end loop

unstable regularity (e.g., the weekly air quality is periodic in the long term, but there are locally significant short-term fluctuations due to rain, holiday, or other reasons). So, the periodicity characteristic among input variables has difficulty in being learned uniformly. Our strategy to learn periodicity representations requests that we should not depend on timestamp variables, as they only allow us to learn temporal properties from fixed timesteps. Instead, we design the sliding training window (sequence length ω) to include part or all of the time data within a specific time interval. In this work, we design a targeted algorithm, Perio-N2v, that draws on the ideas of Time2vec [16]. By this method, we pay attention to the periodic characteristic of numeric sequence in a given window.

Methodologically, we model periodic characteristics through regular patterns of learned offsets and wavelengths. This completion assigned to extract temporal properties makes that attaching periodic-numeric representation to time series, which helps learn this periodicity associated with the downstream forecast. Formally, we add periodic constraints explicitly over a numeric sequence by the following formula:

$$\mathcal{R}_p[i] = \mathcal{T}(\mathcal{W}_i x + b_i),\ 0 \leq i < k_p, \tag{7}$$

where $\mathcal{T}$ denotes a periodic function that can be specified as required (e.g., sinusoidal function). Additionally, k_p is the transformed representation dimension. Specifically, for $0 \leq i < k_p$, $\mathcal{W}_i$ and b_i are the frequency and the phase shift of the periodic function. For instance, the period of $\mathcal{T}(\mathcal{W}_i x + b_i)$ is $\frac{\pi}{\mathcal{W}_i}$—that is, the trends between sequences are consistent approximately at timestep τ and $\tau + \frac{\pi}{\mathcal{W}_i}$. Here, we append a relative order index according to sequence window ω so as to prevent the position from being misaligned. So, $\mathcal{R}_p[i]$ is the i^{th} element of periodic representation. Again, due to the existence of window ω, we can extract local temporal properties. Other training details are described in Algorithm 3. Through it, we add explicit transition vectors that extract periodicity characteristic with learnable frequency and phase shifts of a periodic function, which reflects the changing trends over time.

4.3 Multi-Numeric Representation Fusion

In this section, we describe original data and multi-numeric representations feature fusion design. For multi-numeric representation fusion, we need to consider the importance of three representations on the forecast results. Take forecasting the temperature of the next day based on the

weather conditions of the previous week as an example. Obviously, if the difference in historical temperature fluctuates greatly, the constraints of arithmetic characteristic will make a greater significance on prediction. Again, it is well known that the formation of wind direction depends on the temperature difference between the wind source and this locality, which can be reflected in the weight of direction characteristics. Additionally, week-periodic and day-periodic properties are extremely crucial since sequences exhibit strong periodicity characteristics. Consequently, we realize that the influence degrees of different representations for downstream forecast targets are different when fusing the preceding three representations. So, we exploit the weighted sum method to merge the multi-numeric representation for the purpose of obtaining the final inputs with explicit characteristics:

$$\begin{aligned} \mathcal{R} &= W_a \odot \mathcal{R}_a + W_d \odot \mathcal{R}_d + W_p \odot \mathcal{R}_p, \\ \mathcal{X} &= [\mathcal{X}; \mathcal{R}], \end{aligned} \tag{8}$$

where $\odot$ is the Hadamard product. W_a, W_d, and W_p are learned weight parameters, reflecting the influence degree of three numeric representations respectively on the forecasting target. Additionally, we concatenate original time series $\mathcal{X}$ with multi-numeric representations $\mathcal{R}$ to get the final inputs for the predictor.

5 EXPERIMENTS

5.1 Dataset Descriptions

This work conducts experiments with air quality and weather data to illustrate the feasibility and effectiveness of subjoining our pre-training numeric representation Num2vec. We selected four representative public world datasets, all of which are widely used in research works. The first is KDDCUP-B. This dataset comes from KDD CUP 2018,[1] ranging from January 31, 2017 to January 30, 2018 (365 days) in Beijing, China. Each record contains six air quality observations (i.e., $PM_{2.5}$, PM_{10}, NO_2, SO_2, O_3, and CO) and five meteorological observations (i.e., temperature, pressure, humidity, wind speed, and wind direction). The second is KDDCUP-L. This dataset is also from KDD CUP 2018, ranging from January 1, 2017 to December 31, 2017 (365 days) in London, England. Each record contains three quality observations (i.e., $PM_{2.5}$, PM_{10}, and NO^2) and five meteorological observations that are same as the KDDCUP-B dataset. The third is BJMEMC. This dataset is crawled from the Beijing Environmental Quality Monitoring Center[2] and the National Climatic Data Center (NCDC),[3] ranging from January 1, 2020 to December 31, 2020 (365 days) in Beijing, China. Compared with the KDDCUP-B dataset, it only lacks two meteorological observations (i.e., pressure and humidity). The fourth is ETT.[4] This dataset is from the work of Zhou et al. [41], collected over 2 years (July 1, 2016 to June 26, 2018) with data from two separated counties in China. Each record contains seven electricity observations (i.e., HUFL (High Useful Load), HULL (High Useless Load), MUFL (Middle Useful Load), MULL (Middle Useless Load), LUFL (Low Useful Load), LULL (Low Useless Load), and OT (Oil Temperature)).

For the first three datasets, we choose $PM_{2.5}$ as the target value, and for the ETT dataset, we choose OT as the target value. Note that all observations in the four datasets were recorded with equal hourly intervals. Additionally, we divide the training set and test set by 8:2.

[1] https://www.biendata.xyz/competition/kdd_2018/data/.
[2] http://www.bjmemc.com.cn/.
[3] https://www.ncdc.noaa.gov/.
[4] https://github.com/zhouhaoyi/ETDataset.

5.2 Experiments Details

5.2.1 Baselines. The numeric representations of Num2vec we presented are universally dedicated to improving the prediction results of multiple models. Next, we compare its performance based on the following four baseline models:

- *LSTM* is a type of recurrent neural network capable of learning order dependence in long sequence prediction problems [15].
- *GRU* is like LSTM with a forget gate, which also solves the problem of gradient disappearance and gradient explosion in long sequence training [5].
- *LSTnet* is a deep learning framework that utilizes both the convolutional layer to discover the local dependency patterns and the recurrent layer to capture complex long-term dependencies [17].
- *TPA-LSTM* is an attention model for multivariate time series forecasting, which can learn inter-dependencies among multiple variables [26].
- *Informer* is an efficient transformer-based model for long sequence time series forecasting, which achieves $O(LlogL)$ in time complexity and memory usage [41].

To verify the effectiveness of our Num2vec, we conducted experiments on five predictive models by plugging in our pre-trained Num2Vec, respectively, named N2v-∼ for short, whose "∼" is replaced with the five methods' names. Among them, we take the arithmetic, direction, and periodicity representations obtained from Section 4 as pre-training feature values to the input layer. As a result, it assists the original data for training better by adding prior knowledge of the logic characteristic.

5.2.2 Evaluation Metrics. We use RMSE (root mean squared error), MAE (mean absolute error), and MAPE (mean absolute percent error), which are three widely used metrics to compare different experiments' results. Furthermore, we introduce an additional error metric, PCV (percentage change in variance) [27], which is conducive to measuring the consistency of methods by calculating the difference between the variance in the actual data and the predicted data. As we all know, the lower these four metrics are, the better the prediction ability of this model is, and the better it can demonstrate the superiority of subjoining our pre-training numeric representations. We perform significant tests using the t-test and F-test. Differences are considered statistically significant when the p-value is lower than 0.05.

5.2.3 Parameter Settings. To make fair comparisons, the optimizer is determined from 0.1 to 0.00001, and we use a cosine annealing strategy to avoid getting stuck in local optima. The hidden size of models is tuned in the range of [32, 64, 128, 256, 512]. Here, we conduct fivefold cross validation on the training set to tune the best hyper-parameters of each baseline. Additionally, we use Microsoft NNI[5] to perform optimization of hyper-parameters of all methods. As well, we first initialize the weights using Kaiming initialization and Xavier initialization for all the Conv and FC layers of models. For RNN layers, we use the orthogonal initialization to prevent gradient explosion.

Especially, for our Arith-N2v, the hidden size of the model is fixed to 128, trained by the Adam optimizer with a learning rate of $lr = 0.01$. The arithmetic numeric representation is mapped to high-dimensional vectors of 128 dimensions. For our Dir-N2v, the hidden size of the model is fixed to 128, trained by the Adam optimizer with a learning rate of $lr = 0.005$. The direction-numeric representation is mapped to vectors of 16 dimensions. For our Perio-N2v, the hidden size of the model is fixed to 128. The periodic function is set to cos, and the numeric representation is mapped

[5]https://github.com/Microsoft/nni.

Table 1. Number Representation Evaluation Results on Four Datasets

Dataset	Metric	LSTM		GRU		LSTnet		TPA-LSTM		Informer	
		~	N2v-~	~	N2v-~	~	N2v-~	~	N2v-~	~	N2v-~
KDDCUP-B	RMSE	11.28	**11.02**	11.34	**11.07**	11.14	**10.97**	11.09	**10.99**	11.08	**10.95**
	MAE	6.31	**6.31**	6.47	**6.29**	6.39	**6.09**	6.26	**6.07**	6.24	**6.20**
	MAPE	30.89	**30.89**	30.95	**30.45**	32.12	**28.32**	31.46	**29.45**	30.34	**28.62**
	PCV	4.61	**4.15**	4.61	**2.98**	2.98	**2.25**	5.41	**5.40**	5.64	**5.36**
KDDCUP-L	RMSE	3.34	**3.29**	3.40	**3.31**	3.33	**3.27**	3.32	**3.23**	3.30	**3.24**
	MAE	2.30	**2.25**	2.34	**2.18**	2.31	**2.21**	2.32	**2.10**	2.24	**2.19**
	MAPE	3.30	**3.12**	3.39	**2.82**	3.22	**3.00**	3.65	**2.85**	2.15	**2.12**
	PCV	9.54	**8.61**	10.29	**8.65**	7.60	**6.89**	6.90	**5.55**	4.17	**3.59**
BJMEMC	RMSE	7.79	**7.66**	7.82	**7.69**	7.75	**7.64**	7.77	**7.66**	7.72	**7.61**
	MAE	4.67	**4.68**	4.72	**4.48**	4.68	**4.51**	4.58	**4.55**	4.46	**4.41**
	MAPE	24.59	**23.25**	25.09	**23.16**	26.24	**22.16**	25.78	**23.72**	23.82	**22.90**
	PCV	4.87	**4.73**	6.48	**4.60**	2.87	**2.19**	5.95	**5.3**5	6.08	**4.79**
ETT	RMSE	0.641	**0.628**	0.651	**0.636**	0.636	**0.629**	0.634	**0.625**	0.625	**0.618**
	MAE	0.431	**0.422**	0.441	**0.433**	0.421	**0.414**	0.415	**0.411**	0.413	**0.411**
	MAPE	2.340	**2.302**	2.267	**2.258**	2.305	**2.289**	2.276	**2.247**	2.192	**2.178**
	PCV	0.106	**0.102**	1.573	**1.207**	0.123	**0.108**	0.114	**0.104**	0.102	**0.101**

Models with Num2vec are denoted as N2v-~, and the tilde omits the models' name. The best results are in bold.

to vectors of 128 dimensions. To consider the randomness of our experiment, we dynamically fine-tune these numeric representations by setting bias before integrating into the downstream task. For forecast models, we set the batch size to 128. Each original numerical data is normalized by the Z-score. We split the time series based on the sliding window approach, and make a simple single-step prediction according to the previous 48 hours (window size) of records. We implement Nume2vec and other baseline models in PyTorch. All methods are conducted on a Linux server with eight NVIDIA RTX2080ti GPUs.

5.3 Results and Discussion

Table 1 presents the evaluation results between baseline methods and basic models with Num2vec on four datasets. These results show that models subjoining our pre-training Nuw2vec module consistently outperform these baselines without Num2vec on the four datasets by achieving around 4% less MAE, 3% less RMSE, and 4% less MAPE, which verifies that modeling intrinsic arithmetic, direction, and periodicity improves the predictor's ability to effectively forecast future data. Specifically, in these basic models, Informer, LSTnet, and TPA-LSTM obtain more accurate forecasting results compared with the sample LSTM and GRU. But on the contrary, N2v-LSTM and N2v-GRU achieve better performance improvements compared with the baselines, which reveals that the slightly simple models (with certain characteristic learning ability but not enough explicitly) themselves are quite limited in learning numeric characteristics of sequences. Again, we observe that different datasets have different prediction effects (i.e., ETT > KDDCUP-L > BJMEMC > KDDCUP-B), which reflects the influence of environment and date on air quality and weather. Furthermore, the PCV values of N2v-LSTnet and N2v-Informer are closer to zero than other methods, which illustrates

that these methods are relatively stable. The PCV of the London17 dataset is the farthest from zero, but other metrics are smaller because of the smoothing of true values and the instability of predicted values. Nevertheless, KDDCUP-B and KDDCUP-L obtain better improvement results—that is, an average decrease of 2 percentage points compared with BJMEMC and ETT. This is related to the influence of natural environmental factors as well as unpredictably abnormal events, which further indicates the complexity of numerical forecasting.

5.4 Microbenchmarks

In this section, we conduct performance analysis separately on three numeric representations, which is dedicated to exploring the ability to extract potential arithmetic, direction, and periodicity characteristics, as well as to help understand the contribution of each part to our Num2vec methods.

5.4.1 Arithmetic Numeric Representations. To further verify the effectiveness of learning arithmetic characteristics, we design four variants of the basic method and conduct a controlled study on the KDDCUP-B dataset. Specifically, we choose LSTM as our basic forecast model and $PM_{2.5}$ concentration values as our metric numbers. So, the remaining four variants add a corresponding representation module based on LSTM to extract arithmetic relations. Each configuration of the models is as follows:

- *LSTM*: Only the basic forecast model without an arithmetic representations module.
- *LR-LSTM*: Using a normal linear module instead of a representation module. Intuitively, the linear operation can extract linear features of arithmetic relations, which projects numbers to higher-dimensional vectors by multiplication.
- *transLSTM*: Applying the traditional transE method acting as a representation module, which translates numbers to high-dimensional vectors to extract arithmetic relations.
- *Arith-N2v*: Our Num2vec method only contains the arithmetic representation module.
- *Arith*-N2v*: A method that generates one of the arithmetic representation results theoretically based on the idea of our Arith-N2v, which converts to vectors by mapping metric numbers to the diagonal line of high-dimensional transition space.

Table 2 summarizes the results of preceding five models. It shows that arithmetic numeric representation on sequences is profitable and indispensable. Specifically, LR-LSTM outperforms the basic LSTM method, which verifies that maintaining linear features of metric numbers into high-dimensional space can help improve the forecasting accuracy. However, it is defective if applying this normal linear module to all features, because the numbers do not have the ability to distinguish various characteristics. In other words, those numeric series that possess linear features are considered as a relational constraint of the unified scaled proportion (i.e., 0.1, 0.2, and 1,4). Additionally, only the linear relationship is not capable of explicitly demonstrating the significance of the difference between numerical values. Therefore, it is extremely crucial to attach the extraction of the arithmetic characteristic. For transLSTM, the performance is slightly better than the basic LSTM model, but there is little difference, so the improvement of representation is not persuasive. Our Arith-N2v model shows that analogous performance to LN-LSTM, which further proves the effectiveness of extracting the arithmetic characteristic. The difference is not significant as a result of only single-scaled data in the existing contents.

However, this method is nowhere near achieving the theoretically optimal result due to the deviation loss of training.

Our Arith*-N2v model generates better arithmetic representations in theory according to the idea of Arith-N2v. The results achieved a significant improvement among the control models. This

Table 2. Performance Comparison on Arithmetic Representation

Metric	RMSE	MAE
Arith*-N2v	**11.701**	**6.370**
Arith-N2v	11.772	6.431
transLSTM	11.827	6.458
LR-LSTM	11.783	6.453
LSTM	11.881	6.502

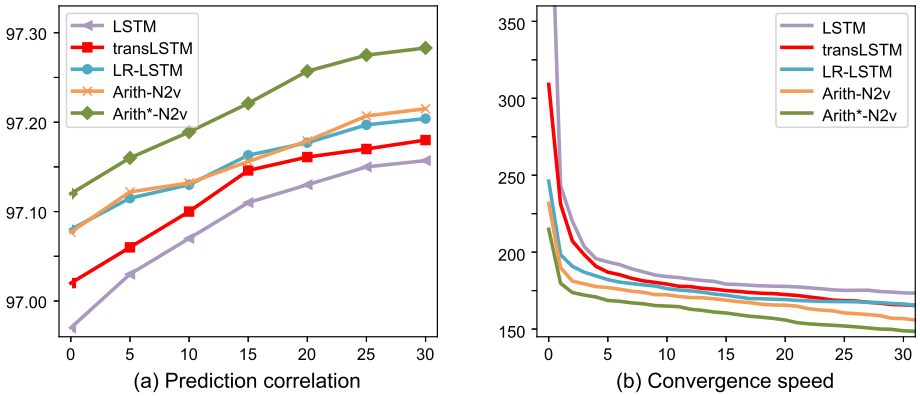

Fig. 7. Control study of arithmetic representation on the KDDCUP-B dataset.

proves that arithmetic numeric representation can adapt to the model better than the ordinary linear module and enhance forecast performance. Moreover, first, Figure 7(a) shows the correlation between ground truth and the predicted result on the preceding methods. The basic LSTM model presents the lowest prediction accuracy, and our Arith*-N2v model shows the highest prediction accuracy. This result further proves that it is essential to arithmetic constraints, and our representation can better maintain arithmetic dependencies between metric numbers in high-dimensional space. Second, Figure 7(b) shows the convergence speed of models during training. Obviously, compared with the basic LSTM method, our arithmetic representation has the benefit of reducing the training epochs of models and accelerating model convergence.

5.4.2 Direction Numeric Representations. Next, we analyze and compare our Dir-N2v with several traditional methods based on the KDDCUP-L dataset, so as to verify the availability of extracting the directional characteristic. We choose GRU as our basic forecast model, wind direction values as our angle numbers, and $PM_{2.5}$ concentration as our prediction object. The remaining three variants add a directional representation module based on GRU. Each configuration of models is as follows:

- *GRU*: Only the basic forecast model without a directional representations module.
- *Emd-GRU*: Adding a generic embedding layer to embed angle numbers into a low-dimensional space of two dimensions.
- *Polar-GRU*: Adding a directional feature processing module to map the angular numbers into the two-dimensional representation of polar coordinates.
- *Perio-N2v*: Our Num2vec method only contains directional numeric representation module.

Table 3. Performance Comparison on Direction Representation

Metric	RMSE	MAE
Dir-N2v	**7.830**	**4.626**
Polar-GRU	7.901	4.746
Emd-GRU	7.913	4.710
GRU	7.898	4.671

Table 4. Performance Comparison on Periodicity Representation

Metric	MAE	RMSE
Prio-N2v	**11.027**	**6.274**
Prio*-N2v	11.123	6.297
GRU	11.220	6.433

Table 3 summarizes the results of MAE and RMSE among the preceding methods. It can be observed that the performance of GRU is not much different from Polar-GRU and Embed-GRU. We can investigate this from the following aspects. First, Emd-GRU calculates the weight matrix to reduce dimensionality for 361 sparse angles, which treats them as discrete values simply but lacks the relevant constraints of a directional relationship. Again, although Polar-GRU converts into directional representation through the sine-cosine function, it is easily destroyed in high-dimensional space. Our Perio-N2v achieves better prediction results with about a percentage point increase, which proves the effectiveness of extracting dithe rection characteristic. Our method helps downstream forecast models learn the high-dimensional representation of the direction by Dir-N2v, which is conducive to being distinguished from numbers with other relationships, such as arithmetic. In view of this, each angle value is regarded as a discrete value, and the interval and opposite relationships are used to constrain it, thus improving prediction performance from the perspective of modeling direction representations.

5.4.3 Periodicity Numeric Representations. Here, we analyze the role of periodic representation and explore whether part of the time variables or the whole variables can help yield better performance. In allusion to all time data on the KDDCUP-B dataset, we expand and analyze this method configuration based on the following variants:

- *GRU*: Only the forecast model without the periodic representation module.
- *Prio-N2v*: Our Num2vec method only contains the periodic representation of predicted values.
- *Prio*-N2v*: Our Num2vec method only contains periodic representation of all time data.

With the designs mentioned previously, we perform a rigorous performance comparison on MAE and RMSE and give the results in Table 4. We can find that both Prio-N2v and Prio*-N2v achieve better results than the basic LSTM model, increasing by 2 percentage points and 1 percentage point, respectively, which also shows the availability of our periodic representation method. Moreover, Prio*-N2v only constrains that the periodic properties of predicted value is superior to Prio*-N2v. This shows a key insight—that is, despite Prio*-N2v capturing all time series time representation, it interferes with the periodic learning of predicting results. In other words, the effect to extract only the periodic characteristics of the predicted values is more targeted.

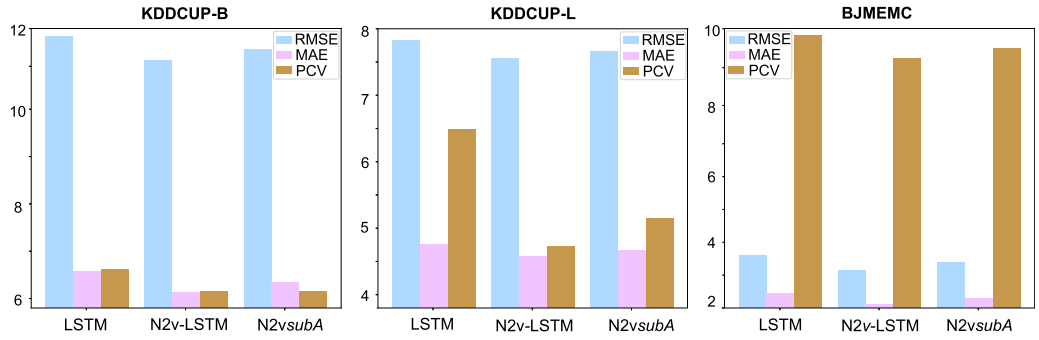

Fig. 8. Performance comparisons between N2v-LSTM and its two control models, LSTM and N2v*subA*, over three datasets.

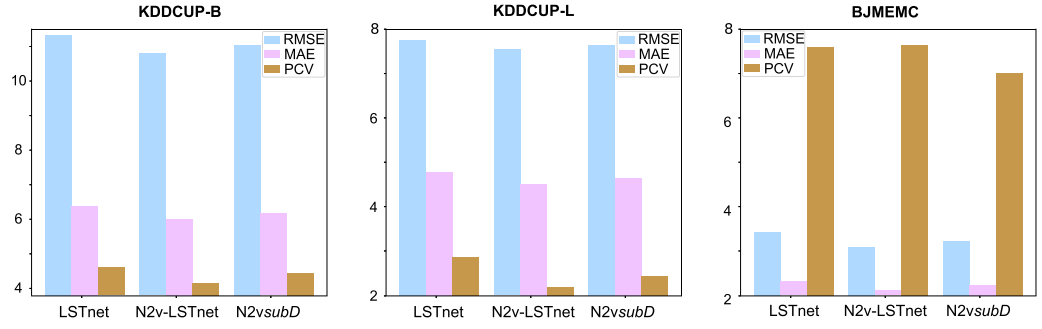

Fig. 9. Performance comparisons between N2v-LSTnet and its two control models, LSTnet and N2v*subD*, over three datasets.

5.5 Ablation Study

In this section, we conduct experiments to analyze variants of Num2Vec via an ablation study.

5.5.1 Analysis on the Arith-N2v Module. Recall that Num2Vec utilizes an Arith-N2v module to capture the arithmetic characteristic, and in this section we aim to analyze whether such a design can bring benefits. Here, we directly remove the Arith-N2v module as the final numeric representations. According to such a design, we denote the new variant as N2v*subA*. The results of Num2vec and N2v*subA* on three datasets are shown in Figure 8.

Specifically, we can see that N2v-LSTM performs obviously better than N2v*subA* with an improvement of about 2 percentage points in RMSE. This reveals that explicitly capturing the arithmetic characteristic indeed has a noticeable effect. Owing to arithmetic representations, the base model can directly learn this characteristic for better performance.

5.5.2 Analysis on the Dir-N2v Module. In Num2Vec, the Dir-N2v module helps models obtain better numerical direction recognition. To verify the effectiveness of Dir-N2v, we also make some degradation of Num2vec. Specifically, we choose LSTnet for verification, and we denote the new variant as N2v*subD* for LSTnet without using the Dir-N2v module. Figure 9 shows the performance comparisons of LSTnet, N2v-LSTnet, and N2v*subD* on three datasets.

We can see that compared with LSTnet, N2v*subD* and N2v-LSTnet achieve higher performance. It is easy to understand that a lack of dominant direction characteristic extraction is difficult to learn the exact features due to the random embedding space. Compared with N2vsubD,

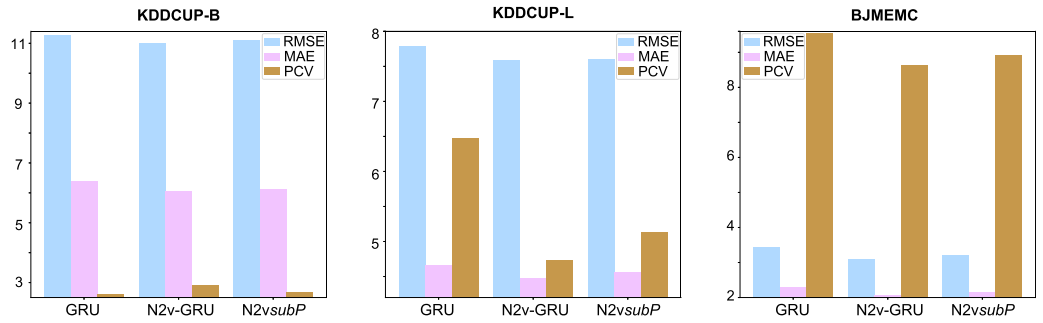

Fig. 10. Performance comparisons between N2v-GRU and its two control models, GRU and N2v*subP*, over three datasets.

N2v-LSTnet performs better with about 1 percentage point improvement in RMSE. This demonstrates the correctness and necessity of using direction numeric representations in Num2Vec, which helps the downstream task capture an explicit directional relationship.

5.5.3 Analysis on the Perio-N2v Module. Recall that we use Perio-N2v to facilitate the model learning periodicity representations. To validate effectiveness of our proposed Perio-N2v, we design N2v*subP*, a variant of Num2Vec that eliminates the Perio-N2v module. We choose GRU as our base model for validation.

Results are shown in Figure 10, in which it can be seen that models with periodicity characteristic extraction all contribute to improvement in three datasets. Compared to N2v*subP*, N2v-GRU achieves better performance with about 2% improvement in RMSE, indicating that our Num2Vec is influenced by the temporal characteristic extraction module and thus learning periodicity representations could better support prediction of the downstream model.

5.6 Analysis on the Hyper-Parameters

5.6.1 Analysis on the Representation Dimension. In this article, we use three numeric representations to facilitate the modeling learning for downstream models. Different values of k, the dimension of numeric representation in Num2Vec, has different effects on prediction performance. Here, we test different values of three representation dimensions k_a, k_d, k_p in {8, 16, 32, 64, 128, 256} to search for the best prediction accuracy. We use LSTM and LSTnet as our base model, and the results on the KDDCUP-B dataset are shown in Figure 11.

As can be seen, three numeric representations significantly improve prediction accuracy, respectively. The best representation dimensions of three modules are different, which reflects the structural differences in high-dimensional representations of individual characteristics. Specially, we mark the best performance with a rectangle. Based on these mark results, we set three representation dimensions $k_a = 128, k_d = 16, k_p = 128$ in our model, respectively.

5.6.2 Analysis on the Window Size. Last, we study the impact of different window sizes ω in the range of 12 to 72 when all other hyper-parameters remain the same. Here, we examine different choices of ω on the KDDCUP-B dataset, and analyze their impacts to the prediction performance. The results on four base models are shown in Figure 12.

As we can see, when ω is set too large or too small, the overall performance decreases. The reason is that a larger window size exacerbates the difficulty of model convergence and a smaller window size lacks structural information. Moreover, all models with our Num2Vec achieve optimal performance with different window sizes. We believe that since learning potential numeric

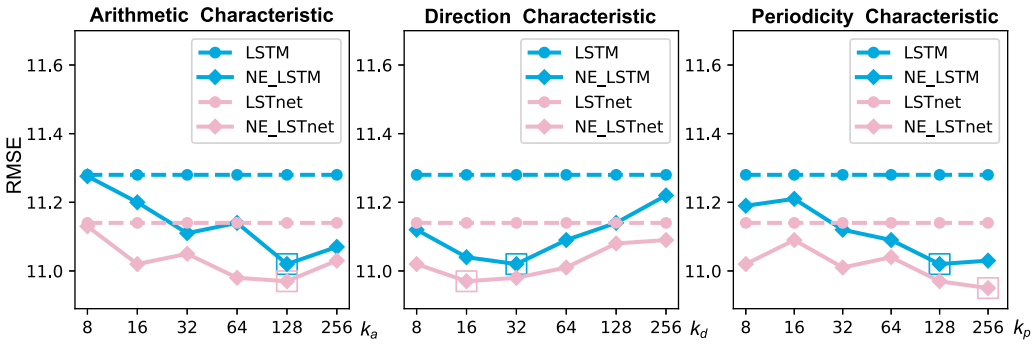

Fig. 11. The impact of the representation dimension in terms of RMSE on the KDDCUP-B dataset. The dimensionality is increased from 8 to 256. The rectangle indicates the best performance in each experiment.

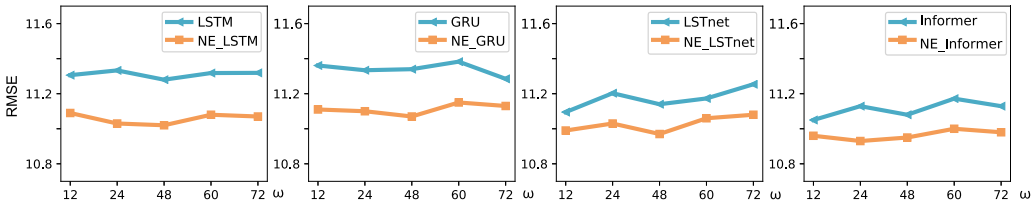

Fig. 12. The impact of the window size in terms of RMSE on the KDDCUP-B dataset. The window size is increased from 12 to 72.

representation helps extend potential characteristics to models' parameter space, it preserves the essential knowledge of the numeric sequences in high-dimensional transition vectors, thus achieving better prediction performance. Based on the results, we set a window size of $\omega = 48$ in our experiments.

6 CONCLUSION

In this article, we presented Num2vec, a pre-training numeric representation learning method to extract potential logic characteristics (i.e., arithmetic, direction, and periodicity). It attaches multi-numeric characteristics to sequences based on realistic constraints and converts them to high-dimensional transition vectors separately. This solves the problem that numerical characteristics and potential relationships are difficult to distinguish and maintain because of their slightly external difference and united high-dimensional transformation. Extensive experimental results on four real-world datasets demonstrated that the model appended to our method achieves better performance.

REFERENCES

[1] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 8 (2013), 1798–1828.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*, Vol. 2. 2787–2795.

[3] Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee. 2017. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691* (2017).

[4] Hongyun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. A comprehensive survey of graph embedding: Problems, techniques, and applications. *IEEE Transactions on Knowledge and Data Engineering* 30, 9 (2018), 1616–1637.

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[6] Shivani Choudhary, Tarun Luthra, Ashima Mittal, and Rajat Singh. 2021. A survey of knowledge graph embedding and their applications. *arXiv preprint arXiv:2107.07842* (2021).
[7] Yuanfei Dai, Shiping Wang, Neal N. Xiong, and Wenzhong Guo. 2020. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics* 9, 5 (2020), 750.
[8] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).
[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[10] Zulong Diao, Xin Wang, Dafang Zhang, Yingru Liu, Kun Xie, and Shaoyao He. 2019. Dynamic spatial-temporal graph convolutional neural networks for traffic forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 890–897.
[11] Yuyu Ding, Hai Zhou, Zhiping Tan, Ying Chen, and Jie Ding. 2012. The influence of wind direction on short-term wind power prediction: A case study in north China. In *Proceedings of IEEE PES Innovative Smart Grid Technologies*. IEEE, Los Alamitos, CA, 1–5.
[12] Josefine Gibson. 2015. Air pollution, climate change, and health. *Lancet Oncology* 16, 6 (2015), e269.
[13] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. 2019. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 922–929.
[14] Mohammad M. Hamed, Hashem R. Al-Masaeid, and Zahi M. Bani Said. 1995. Short-term prediction of traffic volume in urban arterials. *Journal of Transportation Engineering* 121, 3 (1995), 249–254.
[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (1997), 1735–1780.
[16] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. 2019. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321* (2019).
[17] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*. 95–104.
[18] Xinzhi Lin. 2021. The application of machine learning models in the prediction of PM2. 5/PM10 concentration. In *Proceedings of the 2021 4th International Conference on Computers in Management and Business*. 94–101.
[19] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*.
[20] Yan Lin, Huaiyu Wan, Shengnan Guo, and Youfang Lin. 2020. Pre-training context and time aware location embeddings from spatial-temporal trajectories for user next location prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
[21] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[22] Manfred Mudelsee. 2019. Trend analysis of climate time series: A review of methods. *Earth-Science Reviews* 190 (2019), 310–322.
[23] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP'14)*. 1532–1543.
[24] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).
[25] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Preprint.
[26] Shun-Yao Shih, Fan-Keng Sun, and Hung-Yi Lee. 2019. Temporal pattern attention for multivariate time series forecasting. *Machine Learning* 108, 8 (2019), 1421–1441.
[27] Sehyun Tak, Soomin Woo, and Hwasoo Yeo. 2016. Data-driven imputation method for traffic data in sectional units of road links. *IEEE Transactions on Intelligent Transportation Systems* 17, 6 (2016), 1762–1771.
[28] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the International Conference on Machine Learning*. 2071–2080.
[29] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.
[30] Shuo Wang, Yanran Li, Jiang Zhang, Qingye Meng, Lingwei Meng, and Fei Gao. 2020. PM2. 5-GNN: A domain knowledge enhanced graph neural network for PM2. 5 forecasting. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*. 163–166.

[31] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.
[32] Jing Wei, Zhanqing Li, Lin Sun, Wenhao Xue, Zongwei Ma, Lei Liu, Tianyi Fan, and Maureen Cribb. 2021. Extending the EOS long-term $PM_{2.5}$ data records since 2013 in China: Application to the VIIRS Deep Blue aerosol products. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2021), Article 4100412.
[33] Haomin Wen, Youfang Lin, Fan Wu, Huaiyu Wan, Shengnan Guo, Lixia Wu, Chao Song, and Yinghui Xu. 2021. Package pick-up route prediction via modeling couriers' spatial-temporal behaviors. In *Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE'21)*. IEEE, Los Alamitos, CA, 2141–2146.
[34] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. 2020. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478* (2020).
[35] Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems* 2, 1-3 (1987), 37–52.
[36] Bishan Yang, Wen-Tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).
[37] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. 2015. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*.
[38] Eddie Yatiyana, Sumedha Rajakaruna, and Arindam Ghosh. 2017. Wind speed and direction forecasting for wind power generation using ARIMA model. In *Proceedings of the 2017 Australasian Universities Power Engineering Conference (AUPEC'17)*. IEEE, Los Alamitos, CA, 1–6.
[39] Poorya Zaremoodi, Wray Buntine, and Gholamreza Haffari. 2018. Adaptive knowledge sharing in multi-task learning: Improving low-resource neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 656–661.
[40] G. Peter Zhang. 2003. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing* 50 (2003), 159–175.
[41] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI'21)*.
[42] Francisco Martínez, María Pilar Frías, María Dolores Pérez, and Antonio Jesús Rivera. 2019. A methodology for applying k-nearest neighbor to time series forecasting. *Artificial Intelligence Review* 52, 3 (2019), 2019–2037.
[43] Andrew D. Short and Arthur C. Trembanis. 2004. Decadal scale patterns in beach oscillation and rotation Narrabeen Beach, Australia—time series, PCA and wavelet analysis. *Journal of Coastal Research* 20, 2 (2004), 523–532.
[44] Kiyoung Yang and Cyrus Shahabi. 2004. A PCA-based similarity measure for multivariate time series. In *Proceedings of the 2nd ACM International Workshop on Multimedia Databases*, 65–74.
[45] Yen-Hsien Lee, Chih-Ping Wei, Tsang-Hsiang Cheng, and Ching-Ting Yang. 2012. Nearest-neighbor-based approach to time-series classification. *Decision Support Systems* 53, 1 (2012), 207–217.

Received 6 April 2022; revised 17 October 2022; accepted 20 May 2023