Generative Adversarial Networks for Spatio-Temporal Data:  
A Survey

NAN GAO, HAO XUE, WEI SHAO, SICHEN ZHAO, KYLE KAI QIN, ARIAN PRABOWO, 
MOHAMMAD SAIEDUR RAHAMAN, and FLORA D. SALIM, RMIT University, Australia

Generative Adversarial Networks (GANs) have shown remarkable success in the computer vision area for producing realistic-looking images. Recently, GAN-based techniques are shown to be promising for spatio-temporal-based applications such as trajectory prediction, events generation and time-series data imputation. While several reviews for GANs in computer vision been presented, nobody has considered addressing the practical applications and challenges relevant to spatio-temporal data. In this paper, we conduct a comprehensive review of the recent developments of GANs in spatio-temporal data. We summarise the popular GAN architectures in spatio-temporal data and common practices for evaluating the performance of spatio-temporal applications with GANs. In the end, we point out the future directions with the hope of benefiting researchers interested in this area.

CCS Concepts: • Computing methodologies → Machine learning; Artificial intelligence.

Additional Key Words and Phrases: Generative adversarial nets, spatio-temporal data, time series, trajectory data

ACM Reference Format:

1 INTRODUCTION

Spatio-temporal properties are commonly observed in various fields, such as transportation [134], social science [75] and criminology [125], among which that have been rapidly transformed by the proliferation of sensor and big data. The vast amount of spatio-temporal (ST) data requires appropriate processing techniques for building effective applications. Generally, traditional data mining methods dealing with transaction data or graph data could perform poorly when applied to ST datasets. The reasons are mainly two-fold [145]: (1) ST data are often in continuous space while traditional datasets (e.g., transactions, graphs) are usually discrete; (2) ST data usually have both spatial and temporal attributes where the data correlations are more complex to be captured by traditional techniques. Moreover, ST data tends to be highly self-correlated and data samples are usually not independently generated as in traditional data.

With the prevalence of deep learning, many neural networks (e.g., Convolutional Neural Network (CNN) [74], Recurrent Neural Network (RNN) [100], Autoencoder (AE) [57], Graph Convolutional

Authors’ address: Nan Gao, nan.gao@rmit.edu.au; Hao Xue, hao.xue@rmit.edu.au; Wei Shao, wei.shao@rmit.edu.au; Sichen Zhao, sichen.zhao@rmit.edu.au; Kyle Kai Qin, kai.qin2@rmit.edu.au; Arian Prabowo, arian.prabowo@rmit.edu.au; Mohammad Saiedur Rahaman, saiedur.rahaman@rmit.edu.au; Flora D. Salim, flora.salim@rmit.edu.au, RMIT University, Melbourne, Victoria, Australia, 3000.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2020 Association for Computing Machinery.
2157-6904/2020/12-ART111 $15.00
https://doi.org/10.1145/1122445.1122456

Network (GCN) [69]) have been proposed and achieved remarkable success for modelling ST data. The wide adoption of deep learning for ST data is due to its demonstrated potential for hierarchical feature engineering ability. In this survey, we focus on one of the most interesting breakthroughs in the deep learning field: Generative Adversarial Networks (GANs) [46] and their potential applications for ST data.

GAN is a generative model which learns to produce realistic data adversarially. It consists of two components [46]: the generator $G$ and discriminator $D$. $G$ captures the data distribution and produces realistic data from the latent variable $z$, and $D$ estimates the probability of the data coming from the real data space. GAN adopts the concept of the zero-sum non-cooperative game where $G$ and $D$ are trained to play against each other until reaching a Nash equilibrium. GANs have gained considerable attention in various fields, involving images (e.g., image translation [62], super-resolution [76], joint image generation [88], object detection [32], change facial attributes [29]), videos (e.g., video generation [142]), natural language processing (e.g., text generation [87], text to image [160]).

However, applying image or video generation directly are not applicable for modelling ST data such as traffic flow, regional rainfall, and pedestrian trajectory. On one hand, image generation usually takes the appearance between the input and output images into account, and fails to adequately handle spatial variations. On the other hand, video generation considers spatial dynamics between images, however, temporal changes are not adequately considered when the prediction of the next image is highly dependent on the previous image [130]. Hence, new approaches need to be explored for successfully applying GANs on ST data.

Recently, GANs have started being applied to ST data. The applications for GANs on ST data mainly include the generation of de-identified spatio-temporal events [64, 130], time series imputation [93, 94], trajectory prediction [53, 73], graph representation [14, 143], etc. Despite the success of GANs on computer vision area, applying GANs to ST data prediction is challenging [130]. For instance, leveraging additional information such as Places of Interest (Pol), weather information is still untouched in previous research. Besides, different from the images where researchers could rely on visual inspections of the generated instances, evaluation of GANs on ST data remains an unsolved problem. It is neither practical nor appropriate to adopt the traditional evaluation metrics for GAN on ST data [33, 130].

A few research have reviewed recent literature on the problems in ST data or GAN applications in different fields. For ST data modelling, Atluri et al. [8] have reviewed the popular problems and methods for modelling ST data. A taxonomy of the different types of ST data instances has been provided to identify the relevant problems for ST data in real-world applications. Then, Wang et al. [145] have reviewed the recent progress in applying deep learning to ST data mining tasks and proposed a pipeline of the utilisation of deep learning models for ST data modelling problems. For GAN based applications, Hong et al. [60] explained the GANs from various perspectives and enumerate popular GAN variants applied to multiple tasks. Recent progress of GANs was discussed in [112] and Wang et al. [147] proposed a taxonomy of GANs for computer vision area. Particularly, Yi et al. [154] reviewed the recent advances of GANs in medical imaging.

Nevertheless, all the above works reviewed either ST data modelling problems or the recent progress of GANs in the computer vision area. Though many researchers [33, 53, 93, 94, 130] have modelled ST data with GANs, there is no related survey in this area to address the potential of using GANs for ST data applications. The lack of a comprehensive review makes it more difficult for researchers to identify the set of problems and choose an appropriate method (e.g., architecture, loss function, evaluation metric) when applying GAN techniques for ST applications. For the first time, this paper presents a comprehensive overview of GANs in ST data, describes promising
applications of GANs, and identifies some remaining challenges needed to be solved for enabling successful applications in different ST related tasks.

To present a comprehensive overview of all the relevant research on GANs for ST data, we use Google Scholar ¹ to conduct automated keyword-based search [123]. According to [27], Google Scholar provides coverage and accessibility, and digital libraries such as IEEE Explore ², Science Direct ³, ACM Digital Library ⁴. The search period is limited from 2014 to 2020 (inclusive) as the GAN has first appeared in 2014 [46]. However, papers that introduce novel concepts or approaches for spatio-temporal data mining can be predated 2014. To ensure that our survey covers all relevant primary literature, we have included such seminal papers regardless of their publication date.

The remainder of the paper is organised as follows. In Section 2, we discuss the properties, characteristics and common research problems of ST data. We also present the popular deep learning methods with non-GAN frameworks for ST data, including the Convolutional Neural Networks, Recurrent Neural Networks, Long Short-term Memory and Gated Recurrent Units. Section 3 reviews the definition of GANs and its popular variants with different architecture and loss functions. Section 4 lists the recent research progress for GANs in different categories of ST applications. Section 5 summarises the challenges on processing ST data with GANs, including the adapted architectures, loss functions and evaluation metrics. Finally, we conclude the paper and discuss future research directions.

2 PRELIMINARY

2.1 Spatio-temporal Data

The existence of time and space introduces a rich variety of spatio-temporal data types, leading to different ways of formulating spatio-temporal data mining problems and techniques. In this part, we will first introduce the general properties of spatio-temporal data, then briefly describe the common types of spatio-temporal data in different applications using generative adversarial nets techniques.

2.1.1 Properties. There are several general properties for spatio-temporal data (i.e., spatial reference, time reference, auto-correlation and heterogeneity [8]) described as below.

Spatial Reference. The spatial reference describes whether the objects are associated with the fixed location or dynamic locations [71]. Traditionally, when the data is collected from stationary sensors (e.g., weather stations), we consider the spatial dimension of the data is fixed. Recently, with the boost of mobile computing and location-based services, the dynamic locations of moving objects have been recorded where the collected data comes from sensors attached to different objects, e.g., GPS trajectories from road vehicles [116].

Temporal Reference. The temporal reference describes to what extent the objects evolve [71]. The simplest context includes objects do not evolve at all where only the static snapshots of objects available. In a slightly more complicated situation, objects can change status but only the most recent update snapshot remains where the full history of status is unknown. The extreme context consists of moving objects where the full history of moving is kept, therefore generating time series where all the status have been traversed.

Auto-correlation. The observations of spatio-temporal data are not independent and usually have spatial and temporal correlations between near measurements. For example, in the transportation area, sensors in each parking lot with the unique spatial location can record the temporal

¹https://scholar.google.com/
²https://ieeexplore.ieee.org/
³https://www.sciencedirect.com/
⁴https://dl.acm.org/
Heterogeneity. Spatio-temporal dataset can show heterogeneity in spatial or temporal information on different levels. For instance, traffic flow in a city can show similar patterns between different weeks. During a week, the traffic data on Monday may be different from data on Friday. There can also be inter-week changes due to public events or extreme weather, which can affect the traffic patterns in a city. To deal with the heterogeneity of spatial and temporal information, it is necessary to learn different models for different spatio-temporal regions.

2.1.2 Data Types. There are various spatio-temporal data types in real-world applications, differing in the representation of space and time context [8]. We describe the four common types of spatio-temporal data which have been studied with GAN recently: (1) time series [17, 19, 33, 55, 72, 80, 93, 94, 102, 167]; (2) spatio-temporal events [130, 134]; (3) spatio-temporal graphs [77, 143, 153]; (4) trajectory data [53]. In this part, we provide a taxonomy of the data types available in the spatio-temporal domain, then briefly discuss the properties of those data types and potential difficulties when facing with GANs.

Time Series. A time series can be represented as a sequence of data points $X = \{X_1, X_2, ..., X_n\}$ listed in an order of time (i.e., sequence of discrete-time data [140]). Examples of time series include the values of indoor temperature during a day [38, 119], the changes of accelerometer readings in the IoT devices [37, 39], fluctuations of the stock price in a month [167], etc. Time series analysis consists of techniques to analyse time series for extracting useful statistic information and other characteristics of data. The common questions that used for dealing with time series include but not limited to: Can we predict the future values for time series based on the historical values [72, 106, 148]? Can we cluster groups of time series with similar temporal and spatial patterns [4, 86]? Can we impute the missing values automatically in multi-variate time series [94, 103]? Can we split time series into different segments with its own characteristic properties [28, 63]?

Spatio-temporal Events. An spatio-temporal event represents a tuple containing temporal, spatial information as well as an additional observed value [83]. Generally, it is denoted as $x_i = \{m_i, t_i, l_i\}$, where $t_i$ and $l_i$ indicates the time and location of the event, $m_i$ means the value to describe the event. Typically, the locations are recorded in three dimensions (i.e., latitude, longitude, and
altitude or depth), although sometimes only 1 or 2 spatial coordinates are available. Spatio-temporal events (see Fig. 1(a)) are frequently used in real-world applications such as the taxi demand [118], traffic flow [130], urban crimes [124], forest fires [26], etc. In some cases, spatio-temporal events may even have duration like parking or heliophysics [114]. Usually, an ordered set of spatio-temporal events can also be considered as an trajectory where the spatial locations visited by moving objects. Some common questions that used for analysing spatio-temporal events includes: Can we predict the future spatio-temporal events based on the previous observations [130]? How are spatio-temporal events clustered based on time and space [135]? Can we identify the anomalous spatio-temporal events that do not follow the common patterns of other events [10]?

Trajectory data. A trajectory represents the recordings of locations of a moving object at certain times and it is usually defined as a function mapped from the temporal domain to the spatial domain [13, 35]. Trajectories of moving points can be denoted as a sequence of tuples $P = \{(x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n)\}$, where $(x_i, y_i, t_i)$ indicates the location $(x_i, y_i)$ at time $t_i$. Several research have been conducted in the field of trajectory data mining and there are four major categories [164]: mobility of people [120], mobility of transportation [130], mobility of natural phenomena and mobility of animals [84]. Fig. 1(b) shows an example of two trajectories of object $A$ and object $B$. The common questions for processing trajectory data include: Can we predict the future trajectory based on the historical trajectory traces [53, 127, 128]? Can we divide a collection of trajectories into small representative groups [133]? Can we detect the abnormal behaviours from trajectories [90]?

Spatio-temporal Graph. Spatio-temporal graph structure provides the representation of the relations between different nodes in different time. A sequence of spatio-temporal graphs [153] can be represented as $G = (G_1, G_2, \ldots, G_n)$ where $G_i = \{V_i, E_i, W_i\}$ indicates the graph snapshot at time $T_i (i \in \{1, 2, \ldots, n\})$. Spatio-temporal graphs have been applied in various domains such as commerce (e.g., trades between countries [95]), transportation (e.g., route planning algorithms [42], traffic forecasting [156]) and social science (e.g., studying geo-spatial relations of different social phenomena [51]). Fig. 2 is an example of spatio-temporal graphs in $T_1, T_2, T_3$. Some common questions for processing spatio-temporal graph includes: Can we forecast the status of graph based on the historical graph representations [143, 156]? Can we predict the links based on the previous graph networks [77]?

2.2 Spatio-Temporal Deep Learning with Non-GAN Networks

This section introduces the traditional deep learning approaches for spatio-temporal data mining with Non-GAN networks, including Convolutional Neural Network, Recurrent Neural Network, Autoencoder, Graph Convolutional Network etc.
2.2.1 CNN. Convolutional Neural Network (CNN) [74] is a type of deep, feed-forward neural network commonly used to analyse visual imagery. Similar to other neural networks, a typical CNN model is composed of an input layer, an output layer and some hidden layers as shown in Fig. 3. Several commonly used hidden layers are convolution, Rectified Linear Unit (ReLU) [131] activation, pooling and fully connected layers. Convolutional layer put the previous layer through a series of convolutional filters and each filter activates certain features from the input. ReLU is a non-linear operation used after each convolutional layer, which replaces all negative values in the feature maps by zero while maintaining positive pixel values. Pooling layer simplifies the output from the previous rectified feature map through nonlinear down-sampling and parameter reduction. We take the maximal number of the input area when using the max-pooling layer and mean the number of the input area when using average pooling layer. After several convolutional, ReLU, and pooling layers, there are will be fully connected layers for high-level reasoning classification. Fully connected layers connect all neurons in the previous layer to every single neuron in the next layer, which is similar to the traditional multilayer perceptron (MLP). Compared to MLPs, CNNs can develop internal representations of two-dimensional images, which allows CNNs to be used more generally on other types of data with spatial correlations. Though CNNs are not specifically developed for non-image data, it has been widely used in spatio-temporal data mining problem for trajectory and spatio-temporal raster data [116].

2.2.2 RNN, LSTM and GRU. Recurrent Neural Network (RNN) [100] is a type of neural networks where the previous outputs are fed as the input to the current step, which are widely used in Natural Language Processing (NLP) problems. The advantage of RNN is the hidden state (internal memory) which captures information that has been calculated so far in a sequence. Fig. 4 shows the basic architecture of a RNN, where X is the input data, y is the output data, h is the hidden state and U, V, W indicates the parameters of the RNN. The current state $h_t$ is calculated by the current input $X_t$ and previous state $h_{t-1}$.
Though the RNNs work effectively in many application domains, it may suffer from a problem called vanishing gradients [82]. To cope with this problem, two variants of RNN has been developed: Long Short-Term Memory (LSTM) [58] and Gated Recurrent Units (GRU) networks [22]. LSTM is capable of learning long-term dependencies with a special memory unit as shown in Fig 5. An LSTM cell has three gates (forget gate, input gate, and output gate) to regulate the information flow. Forget gate decides which information we’re going to remember in the cell state. Input gate decides what new information we’re going to store and output gate decides what information we’re going to output. Compared with standard LSTM models, GRU has fewer parameters which combines the input gate and the forget gate into an ‘update gate’ and merges the cell state and hidden state. RNN, LSTM and GRU are widely used to learn the temporal correlations of time series and spatio-temporal data.

2.2.3 Autoencoder (AE). Autoencoder (AE) [57] is a neural network that is trained to copy its input to its output by learning data codings in an unsupervised manner [45]. The network is composed of two parts: encoder and decoder as shown in Fig. 6. Encoder function $h = f(x)$ compresses the input into a latent-space representation and decoder $r = g(h)$ reconstructs the input through the representation. Autoencoder can learn the useful properties of the input data and is commonly used for dimensionality reduction, feature learning, and generative modelling. As a commonly used unsupervised representation learning method, AE is popular for classification and prediction tasks in trajectories [107, 165], time series [61] and other spatio-temporal data [31].

2.2.4 Graph Convolutional Network (GCN). The Graph Convolutional Network (GCN) is capable of extracting representations from hidden layers which encode both local graph structure and node features and it is claimed to be linearly scalable with the size of graph [70]. Traffic prediction becomes popular in recent years and several spatio-temporal deep learning models based on GCNs
have been proposed. Yu et al. introduced Spatio-Temporal Graph Convolutional Networks (STGCN) [156] to solve the prediction problem in traffic networks. Normal deep learning models have some issues in dealing with spatio-temporal forecasting tasks, such as heavy computation of training in RNN-based networks and the normal convolutional operation is limited on grid structures. To solve these problems, STGCN converts traffic network into the graph-structured format and use several spatio-temporal convolutional blocks to learn spatial and temporal dependencies. Each of the blocks consists of graph convolutional layers and convolutional sequence learning layers which reduce the cost of computation via approximation strategies such as Chebyshev Polynomials Approximation or First order Approximation. To seize the spatial and temporal dependencies simultaneously, Zhao et al. also model traffic network via graph and solve traffic forecasting tasks with a novel model called the Temporal Graph Convolutional Network (T-GCN) [161]. In this model, the GCN is applied to learn complex topological structures for extracting spatial dependence and the gated recurrent unit (GRU) is responsible to learn dynamic dependence at temporal aspect.

Recently, the Graph Multi-Attention Network (GMAN) [163] and the Attention-based Spatial-Temporal Graph Convolutional Network (ASTGCN) [52] all adapt attention mechanisms with GCN models to learn the impact of the spatio-temporal factors on traffic conditions. GMAN has an encoder to extract the traffic features as input and predicts the output sequence by the decoder. Several ST-Attention blocks are deployed in both encoder and decoder, and each block contains a spatial attention, a temporal attention and a gated fusion for modelling the correlations between vertices and time frames. Besides, one transform attention layer is used as an intermediate component to reduce the error propagation effect. Differently, ASTGCN uses three independent components to model hourly-periodic, daily-periodic and weekly-periodic dependencies from traffic flows, respectively. However, it is similar to GMAN, each component has an attention to effectively capture the dynamic spatial-temporal correlations in traffic flow and then conduct convolution on the constructed network with GCN.

GCN further demonstrates its capability for human action recognition. Yan et al. model dynamics of human body skeletons via graphs to retain information for human actions [151]. In this work, they propose a novel variant of Spatial-Temporal Graph Convolutional Networks (ST-GCN), which automatically learns both the spatial and temporal patterns from human actions data. The skeleton sequences of human actions are represented by a spatial-temporal graph in a hierarchical way, which contains $N$ human joints and $T$ frames and features not only intra-skeleton connection but also the links between same joints between consecutive frames. To construct the convolutional networks on the defined skeleton graph, the CNN filters are designed for the convolution operation on both the neighbours of one node within one single frame and those across consecutive time frames.

3 GAN AND ITS POPULAR VARIANTS

In this section, we will introduce the basic idea of generative adversarial nets. Then, we will discuss the popular GAN variants and loss functions especially used in the spatio-temporal data modelling applications.

3.1 Basic Idea of GANs

The original concept of GANs is to create two neural networks and let them compete against each other. As shown in Figure 7, the basic architecture of GAN comprises two components: a generator and a discriminator. On the one hand, the generator’s task is to synthesis fake images which can fool the discriminator. On the other hand, the discriminator, as to its name, learns to distinguish if its input is a fake image or not [46].
If we left the images generation task aside, the underlying idea of generative adversarial nets is more general, which is to create one fake distribution \( p_g \) and make it as close as possible to a data distribution \( p_r \). The reason we use such an approach is that \( p_r \) could be hard to get directly and by doing in this manner, we get a good approximation of it and then we can sample from this approximate distribution instead [6]. The advantages of this approach are that since the generator is learning to approximate the real distribution directly, there is no need to introduce the Markov Chain and no inference is required due to the isolation between the generator and the real data distribution. Besides, its simple structure makes it easier to incorporate with other techniques [101].

The Generator \( G(z; \theta_g) \), a neural network that parameterized by theta takes a sample \( z \sim p_z \) as input and mapping that to a sample \( x \sim p_g \). And its rival, the Discriminator \( D(x; \theta_d) \) outputs a single binary value to indicate its prediction of the input’s origin. During the training session, both parts are trained simultaneously and based on their opponent’s result, which forms a minimax game with the overall objective function [46]:

\[
\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]
\]

Despite all the aforementioned advantages, the original generative adversarial network is still inadequate in some places. The practical results show that the training is particularly delicate and the generators may suffer from vanishing gradient for optimizing the generator [6]. To address all those problems might occur, many variants of the vanilla GAN are proposed.

In this paper, we divided those popular variants of GAN into two different categories: modification of its structure and changes on the loss function. The former one is focusing on improving its performance by increasing the overall complexity of its structure while the latter one pays attention to the deficiency of the Jensen–Shannon divergence. Since the plain MLP (multi-layer perceptrons) generator and discriminators in the vanilla GAN [46] can be seamlessly replaced by other types of neural networks such as CNN [117], we do not consider them as the modification of the structure. And it should also be noted that adding auxiliary parts normally leads to a change in the loss function but as long as it still keeps the JSD, it would not be treated as a new type of loss function in this paper.
3.2 Architecture of GANs

Although the vanilla GAN shows its potential for data generation [46], and the discriminator in this structure is proved to be effective on classification task [117]. But it still suffers from blurry and possible mode dropping-collapse. Besides, there is no control in the generation process since its unsupervised manner [109]. To this end, some researches introduce other machine learning techniques into the original GAN structure, and some results are promising. In this section, we describe three types of the variant in that direction.

Support Info. Mirza et al. [101] proposed CGAN (Conditional GAN) which introduces a support info vector $y$. In the generator, each input $z$ gets its corresponding $y$, and it is also available the discriminator which can help it for better judgement. Since this vector is a controlled parameter rather than another random sample, we gain some level of control of the samples generated. Chen et al. [18], on the other hand, is also focused on providing support info to the generator, and proposed the InfoGAN. A latent code $c$ is adding to the input of the generator, and after the images go through the discriminator, another module $Q$ is introduced to approximate the distribution of $P(c|x)$ and calculate the variational mutual information $I(c; G(z, c))$ which indicates the level of info remains after the generation process. By maximising this regularisation term, the result generator can be controlled according to the latent code $c$. Odena et al. [109] introduced a supervised task into the original GAN and proposed ACGAN (Auxiliary Classifier GAN). Every sample from the real data belongs to a predefined class, and an expected label $c \sim p_c$ along with noise $z \sim p_z$ is used as input to generate a data sample of that class. Besides the real/fake discrimination task, an auxiliary classifier is created to classify every sample. This enables the ability for the generator to synthesis sample for a particular class.

Hierarchical Structure. Zhang et al. [160] proposed StackGAN (Stacked GAN) to cope with the problems that the generated images are blurry and hard to scale up. In this structure, two GANs are created, and each of them having different tasks. The task of its first layer, or also called 'Stage-I' GAN is to generate low-resolution images with primitive shapes and colours, while the 'Stage-II' GAN is used as a refiner to increase the resolution to the desired level and corrects possible defects. It also incorporates the CGAN into it, since its task is to synthesis images based on a given sentence. On the base of that idea, Juefei-Xu et al. makes a further improvement and proposed GoGAN (Gang of GANs) [66]. In this method, the whole structure is divided into multiple ranking stage, and each stage has its own GAN model. However, unlike StackGAN [160], which gives different tasks to generators at different stages, all generators in the GoGAN have the same task and same input. A unique constraint is applied to enforce that the images generated by the later stages should be closer to the real data compared to their ancestors. This enables competitions in more dimensions except for the generator versus discriminator one described in the vanilla GAN [46], and analysis shows that it provides faster convergence than WGAN [7].

Except getting multiple sets of generator and discriminator, Karras et al. proposed ProGAN (Progressive growing of GANs) [68] which utilise the same GAN structure and creating more detailed images by incrementally adding more layers to the existing generator. To avoid the damage that could backpropagate from the newly added layer, it uses a weight addition between the upsampling result from the last layer and the image from the new layer. And this new layer will smoothly fade in by introducing a monotone increasing hyperparameter. Although it has the ability to generate images with larger pixels and finer details, this progressive training does consume a lot of computational power and the deeper it goes, the more consuming it will be.

3.2.1 Loss Function. In traditional generative modelling approaches, the performance of a model is indicated by the reverse Kullback-Leibler (KL) divergence between our desire distribution $p_r$ and our generator’s distribution $p_g$ [6].
Generative Adversarial Networks for Spatio-Temporal Data: A Survey

\[ D_{KL}(P_g \parallel P_r) = \int_X P_r(x) \log \frac{P_g(x)}{P_r(x)} \, dx \]

Minimising this term means making those two distributions closer, and it would get to zero once \( p_r = p_g \). However, the generator might still generate fake-looking data due to the imbalanced nature \[6\] of this function. It could heavily penalise the generator for the part that is in the real distribution but not covered by the generator while paying less attention to the extra part covered by the generator. In order to avoid this weakness, another option that is discussed in the original GAN paper is called Jensen-Shannon (JS) divergence \[46\].

\[ D_{JS}(P_r \parallel P_g) = \frac{1}{2} D_{KL}(P_r \parallel \frac{1}{2}(P_r + P_g)) + \frac{1}{2} D_{KL}(P_g \parallel \frac{1}{2}(P_r + P_g)) \]

Although it shows some promising results, JS divergence is not the ultimate choice since it still suffers from issues like gradient vanishing. Some late research shows that those can be resolved by using other types of loss function \[7, 98, 108\]. In this subsection, we describe two major types of loss functions and the GAN method based on them.

**f-divergence.** As mentioned in the previous section, aside from the competitive structure, one difference between GAN and traditional generative modelling methods is the use of JS divergence instead of KL divergence. Its balanced characteristic makes it more suitable for machine learning models to optimise, but it still prunes to mode dropping/collapse empirically \[6, 108\].

Instead of settling down on just a single divergence metric, Nowozin et al. \[108\] proposed \( f \)-GAN, which allows us to choose from many other metrics by introducing a term called \( f \)-divergence.

\[ D_f(P_g \parallel P_r) = \int_X P_r(x) f \left( \frac{P_g(x)}{P_r(x)} \right) \, dx \]

where \( f \) is a convex function such as \( f(x) = x \log x \). Choosing different \( f \) function can lead to different divergence metrics such as KL, reverse KL and Squared-Hellinger divergence. And by using a convex conjugate function, also known as Fenchel conjugate, we can switch the original GAN’s objective function by the following equation:

\[ F(\theta, \omega) = \mathbb{E}_{x \sim P} [T_\omega(x)] - \mathbb{E}_{x \sim Q_\theta} [f^*(T_\omega(x))] \]

where, \( P \) is the real distribution, \( Q_\theta \) is the approximate distribution controlled by parameter \( \theta \), and \( T_\omega \) is the variational function that serves as our discriminator.

Mao et al. \[98\] proposed LSGAN and in that paper, the author dully discussed the relation between their choice of the objective function and \( f \)-divergence and showed that it could be equivalently minimising the Pearson \( \chi^2 \) divergence which makes LSGAN a special form of \( f \)-GAN.

**Integral Probability Metric.** Integral Probability Metric (IPM) is another family of metrics that could be used to measure the distance between two certain distributions \[105\]. Some metrics in this type show nicer properties compared to the original JS Divergence used in vanilla GAN such as less oscillation in the generator training. We mainly cover Wasserstein GAN (WGAN) \[7\] and its improved version \[50\] in this paper since they are the most commonly-used ones.

Arjovsky et al. \[7\] proposed Wasserstein GAN, which including the Earth-Mover (EM) distance or Wasserstein-1 distance shows below:

\[ W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x, y) \sim \gamma} [\|x - y\|] \]

This metric indicates the amount of 'dirt' needs to be moved in order to transform a distribution \( P \) into another \( Q \). This is much suitable than the Jensen-Shannon divergence since it provides some
excellent characteristics to counter the weakness of original GAN. Firstly, since the Wasserstein-1 distance is changing continuously with less huge variations, the gradients of the generator would not be constant zero with an optimal or sub-optimal discriminator. This solves the vanishing gradients problem in training the generator. Secondly, it could also help to cope with mode dropping problem since the Wasserstein-1 distance only reaches zero if those two distributions are the same.

However, there are still some vulnerabilities in this method. The infimum in the original Wasserstein-1 distance equation is highly intractable and have to be converted to the following form:

\[
W(P_r, P_g) = \sup_{\|f\|_{L} \leq 1} E_{x \sim P_r}[f(x)] - E_{x \sim P_g}[f(x)]
\]

where the \(\|f\|_{L} \leq 1\) means that the function \(f\) which indicates the generator function has to be 1-Lipschitz (or \(K\)-Lipschitz for some constant \(K\)). To enforce this constraint, the author uses weight clipping as a preliminary approach [7] and improved to gradient penalty, which shows faster convergence and allows the use of momentum optimiser like Adam optimiser [50].

4 GANS FOR SPATIO-TEMPORAL MODELLING

In this section, we propose a taxonomy of GANs for spatio-temporal data and modelling tasks. As illustrated in Table 1, the formation of our taxonomy mainly comes from several aspects, e.g., ST datatypes and tasks. GANs for spatio-temporal events prediction is firstly introduced. We then discuss tasks for sequence modelling with time series, including sequence generation, imputation and prediction. Recent GAN architectures for graph data are also reviewed. To be specific, we focus on two major tasks for graph data: link prediction and graph representation or embedding. Besides, we discuss some recent work on the trajectory prediction task which has become a popular topic in the research community. Based on this taxonomy, we review the recent progress of applying GANs for different types of spatio-temporal data in the following subsections. In addition, in Table 2, we summarise widely used datasets for each type of spatio-temporal data.

4.1 GANs for Spatio-temporal Events

In this subsection, we will mainly introduce how GANs are applied to predict the spatio-temporal events (e.g., taxi demand [130, 157], crime [64], fluid flows [20], anomaly detection [80]) in the future based on the previous events.

For the first time, Saxena et al. [130] proposed a generative adversarial network D-GAN for accurate spatio-temporal events prediction. In the model, GAN and VAE are combined to jointly learn generation and variational inference of ST data in an unsupervised manner. They also designed a general fusion module to fuse heterogeneous multiple data sources. Figure 8 shows the architecture for D-GAN, consisting of four components: Encoder, Generator/Decoder, Discriminator, and External feature fusion. G network is trained using the adversarial process in which decoder (i.e., generator) learns to approximate the distribution of real data, while the D network discriminate between samples generated by D and samples from real distributions. During the training process, D-GAN adopts a reconstruction loss and adversarial loss [130]. In addition, ConvLSTM [150] and 3D-ConvNet structures were exploited to model long-term patterns and spatial dependencies in ST data.
<table>
<thead>
<tr>
<th>ST data type</th>
<th>Reference</th>
<th>Year</th>
<th>Task</th>
<th>Values in task</th>
<th>Model category</th>
<th>Evaluation methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time series</td>
<td>C-RNN-GAN [102]</td>
<td>2016</td>
<td>Generation</td>
<td>Musical data</td>
<td>GAN and LSTM</td>
<td>Domain metrics (e.g., polyphony, scale consistency, repetitions, tone span)</td>
</tr>
<tr>
<td></td>
<td>RCGAN [33]</td>
<td>2017</td>
<td>Generation</td>
<td>Medical data</td>
<td>GAN and RNN</td>
<td>TSTR and TRTS</td>
</tr>
<tr>
<td></td>
<td>SSL-GAN [17]</td>
<td>2017</td>
<td>Generation</td>
<td>Electronic health records</td>
<td>GAN, CNN and AE</td>
<td>Prediction accuracy over multiple data combinations</td>
</tr>
<tr>
<td></td>
<td>OccuGAN [19]</td>
<td>2018</td>
<td>Generation</td>
<td>Occupancy data</td>
<td>GAN and DNN</td>
<td>Domain metrics (e.g., time of first arrival, number of occupied transitions)</td>
</tr>
<tr>
<td></td>
<td>Grid-GAN [159]</td>
<td>2018</td>
<td>Generation</td>
<td>Smart grid data</td>
<td>CGAN and CNN</td>
<td>TSTR and TRTS</td>
</tr>
<tr>
<td></td>
<td>EGG-GAN [55]</td>
<td>2018</td>
<td>Generation</td>
<td>EEG brain signals</td>
<td>WGAN and CNN</td>
<td>IS, FID and ED</td>
</tr>
<tr>
<td></td>
<td>StockGAN [167]</td>
<td>2018</td>
<td>Generation</td>
<td>Stock data</td>
<td>GAN, CNN and LSTM</td>
<td>Prediction accuracy (e.g., RMSRE, DPA)</td>
</tr>
<tr>
<td></td>
<td>ForGAN [72]</td>
<td>2019</td>
<td>Generation</td>
<td>Synthetic series and internet traffic</td>
<td>CGAN and LSTM</td>
<td>KL divergence</td>
</tr>
<tr>
<td></td>
<td>NAOMI [92]</td>
<td>2019</td>
<td>Imputation</td>
<td>Traffic flow, movement data</td>
<td>GAN and RNN</td>
<td>Imputation accuracy</td>
</tr>
<tr>
<td></td>
<td>TimeGAN [155]</td>
<td>2019</td>
<td>Generation</td>
<td>Sines, stocks, energy and events data</td>
<td>GAN and AE</td>
<td>Diversity, fidelity and usefulness (e.g., TSTR)</td>
</tr>
<tr>
<td></td>
<td>EzGAN [94]</td>
<td>2019</td>
<td>Imputation</td>
<td>Medical records, meteorologic data</td>
<td>GAN and GRU</td>
<td>Imputation accuracy</td>
</tr>
<tr>
<td></td>
<td>Ad-Attack [25]</td>
<td>2020</td>
<td>Generation</td>
<td>Stock prices and electricity data</td>
<td>GAN</td>
<td>Domain metrics (e.g., attack success rate, returned of perturbed portfolio)</td>
</tr>
<tr>
<td>Trajectory</td>
<td>GD-GAN [34]</td>
<td>2018</td>
<td>Prediction</td>
<td>Pedestrian trajectories</td>
<td>GAN and LSTM</td>
<td>Average displacement error (ADE) and final displacement error (FDE)</td>
</tr>
<tr>
<td></td>
<td>SocialGAN [33]</td>
<td>2018</td>
<td>Prediction</td>
<td>Socially acceptable trajectories</td>
<td>GAN and LSTM</td>
<td>Quantitative (e.g., ADE, FDE) and qualitative (e.g., group avoiding) metris</td>
</tr>
<tr>
<td></td>
<td>SoPhie [126]</td>
<td>2019</td>
<td>Prediction</td>
<td>Pedestrian trajectories</td>
<td>GAN and LSTM</td>
<td>ADE and FDE</td>
</tr>
<tr>
<td></td>
<td>APOIR [166]</td>
<td>2019</td>
<td>Prediction</td>
<td>Point-of-Interests</td>
<td>GAN and GRU</td>
<td>Precision, Recall, nDCG and MAP</td>
</tr>
<tr>
<td></td>
<td>CoL-GAN [89]</td>
<td>2020</td>
<td>Prediction</td>
<td>Pedestrian trajectories</td>
<td>GAN, CNN and LSTM</td>
<td>Average collision times (ACT), ADE and FDE</td>
</tr>
<tr>
<td></td>
<td>AdattTUL [40]</td>
<td>2020</td>
<td>Link prediction</td>
<td>Human mobility trajectories</td>
<td>GAN, GRU and LSTM</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td>ST events</td>
<td>D-GAN [120]</td>
<td>2017</td>
<td>Prediction</td>
<td>Taxi and bike data</td>
<td>GAN and VAE</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>Taxi-CGAN [157]</td>
<td>2020</td>
<td>Prediction</td>
<td>Taxi hotspots data</td>
<td>CGAN and LSTM</td>
<td>False identification test (FIT) and the section consistency test (SCT)</td>
</tr>
<tr>
<td></td>
<td>Crime-GAN [64]</td>
<td>2017</td>
<td>Prediction</td>
<td>Crime data</td>
<td>DCGAN, CNN and RNN</td>
<td>Prediction accuracy (e.g., JS divergence) over multiple models</td>
</tr>
<tr>
<td>Graphs</td>
<td>GraphGAN [143]</td>
<td>2018</td>
<td>Representation</td>
<td>Social networks</td>
<td>GAN and DNN</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>NetGAN [14]</td>
<td>2018</td>
<td>Representation</td>
<td>Citation and blogs networks</td>
<td>WGAN and LSTM</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>ANE [24]</td>
<td>2018</td>
<td>Representation</td>
<td>Citation and blogs networks</td>
<td>GAN and DNN</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>NetRA [158]</td>
<td>2018</td>
<td>Representation</td>
<td>Social and biological networks</td>
<td>GAN, LSTM and AE</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>GCN-GAN [77]</td>
<td>2019</td>
<td>Link Prediction</td>
<td>Mobility networks</td>
<td>GAN, GCN and LSTM</td>
<td>MSE, edge-wise KL divergence, mismatch rate</td>
</tr>
<tr>
<td></td>
<td>GANE [59]</td>
<td>2019</td>
<td>Representation</td>
<td>Coauthor networks</td>
<td>WGAN and DNN</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>ProGAN [36]</td>
<td>2019</td>
<td>Representation</td>
<td>Social and citation networks</td>
<td>GAN and DNN</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td></td>
<td>MEGAN [138]</td>
<td>2019</td>
<td>Representation</td>
<td>Social multi-view networks</td>
<td>GAN and MLP</td>
<td>Prediction accuracy over multiple models</td>
</tr>
<tr>
<td>ST data type</td>
<td>Dataset</td>
<td>Data source</td>
<td>Used references</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>-------------</td>
<td>-----------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Time series</strong></td>
<td>Philips eICU database [115]</td>
<td>Medical data</td>
<td>[33]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MNIST [149]</td>
<td>Hand-written digit images</td>
<td>[33]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Occupancy dataset [85]</td>
<td>Occupancy data in the building</td>
<td>[19]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pecan street dataset [65]</td>
<td>Energy consumption, solar generation</td>
<td>[159]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PhysioNet dataset [136]</td>
<td>Medical data (e.g., heart rate, glucose)</td>
<td>[44, 93, 94]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>KDD cup 2018 dataset [3]</td>
<td>Air quality data</td>
<td>[93, 94]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A5M dataset [23]</td>
<td>Transatlantic link data</td>
<td>[72]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PEMS-SF traffic dataset [30]</td>
<td>Freeway occupancy rate</td>
<td>[92]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Appliances energy dataset [16]</td>
<td>Environmental data</td>
<td>[155]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UCI electricity dataset [141]</td>
<td>Historical price data</td>
<td>[25]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yoochoose [12]</td>
<td>Clicking events from users</td>
<td>[162]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MovieLens [54]</td>
<td>Movie ratings data</td>
<td>[162]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Trajectory</strong></td>
<td>ETH [113]</td>
<td>Videos</td>
<td>[5, 34, 53, 73, 89, 126]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UCY [78]</td>
<td>Videos</td>
<td>[5, 34, 53, 73, 89, 126]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stanford drone dataset [121]</td>
<td>Videos</td>
<td>[126]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Vittorio emanuele II [9]</td>
<td>Videos</td>
<td>[34]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Foursquare [152]</td>
<td>Location-based social networks</td>
<td>[40, 81, 91, 97, 166]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gowalla [21]</td>
<td>Location-based social networks</td>
<td>[40, 91, 166]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Brightkite [21]</td>
<td>Location-based social networks</td>
<td>[40, 97]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Yelp [1]</td>
<td>Location-based social networks</td>
<td>[81, 97, 166]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ST events</strong></td>
<td>Yellow taxi dataset [2]</td>
<td>Taxi demand data</td>
<td>[120]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CitiBike trip dataset [104]</td>
<td>Bike demand data</td>
<td>[120]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SWaT dataset [43]</td>
<td>Attacked data in water system</td>
<td>[80]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Graphs</strong></td>
<td>ArXiv-AstroPh [79]</td>
<td>Scientific collaborations data</td>
<td>[143]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wikipedia [49]</td>
<td>Network of words</td>
<td>[24, 143, 158]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CORA [99]</td>
<td>Citation networks of publications</td>
<td>[14, 24, 36]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CiteSeer [132]</td>
<td>Citation networks of publications</td>
<td>[14, 24, 36]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DBLP [111]</td>
<td>Collaboration graph of authors</td>
<td>[14, 24, 153, 158]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Blogcatalog [139]</td>
<td>Social network for bloggers</td>
<td>[36, 143, 158]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UCI message dataset [110]</td>
<td>Message communication networks</td>
<td>[153, 158]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Flickr [122]</td>
<td>Social networks</td>
<td>[36, 138]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Recently, Yu et al. [157] applied a conditional generative adversarial network with long short-term structure (LSTM-CGAN) for taxi hotspot prediction, which captures the spatial and temporal variations of hotspots simultaneously. Jin et al. [64] developed a context-based generative model Crime-GAN to learn the spatio-temporal dynamics of crime situation. They aggregated Seq2Seq, VAE network and adversarial loss in the framework to better study ST data representation. Furthermore, the deep convolutional generative adversarial network (DCGAN) has been developed for spatio-temporal fluid flow prediction in a tsunami case in Japan [20].

GANs have also been used for anomaly detection for spatio-temporal events. Li et al. [80] proposed MAD-GAN, an unsupervised anomaly detection for multivariate time series based on GAN. They trained a GAN generator and discriminator with LSTM; Then, the GAN-trained generator and discriminator are employed to detect anomalies in the testing data with a combined Discrimination and Reconstruction Anomaly Score (DR-Score).
4.2 GANs for Sequence Modelling

The applications of GAN technique on sequence data mainly focus on two aspects: generation and imputation. We will discuss them separately as follows.

4.2.1 Generation. Data generation refers to creating data from the sampled data source. One of the main purposes of time series generation with GAN is to protect the privacy of sensitive data such as medical data [33], electroencephalographic (EEG) data [55], heart signal electrocardiogram (ECG) data [44], occupancy data [19], electronic health records (EHR) [17], etc.

Recently, GANs have been used to generate sequential data. Mogren et al. [102] proposed C-RNN-GAN (continuous RNN-GAN) to generate continuous-valued sequential data. They built the GAN with LSTM generator and discriminator, and the discriminator consists of a bidirectional layout which allows it to take context in both directions into account for its decisions. They trained the model on sequences of classical music and evaluated with metrics such as polyphony, scale consistency, repetitions and tone span.

Then, Esteban et al. [33] proposed a regular GAN where both the generator and the discriminator have been substituted by recurrent neural networks. They presented the Recurrent GAN (RGAN) and Recurrent Conditional GAN (RCGAN) to generate sequences of real-valued medical data or data subject to some conditional inputs. For evaluation, they proposed to use the capability of the generated synthetic data to train supervised models, i.e., TSTR (train on synthetic, test on real). They addressed that TSTR is more effective than TRTS (train on real, test on synthetic) because TRTS performance may not degrade when GAN suffers mode collapse.

GANs have been used for the generation of biological-physiological signals such as EEG and ECG. Hartmann et al. [55] proposed EEG-GAN to generate electroencephalographic (EEG) brain signals. With the modification of the improved WGAN training, they trained a GAN to produce artificial signals in a stable fashion which strongly resembles single-channel real EEG signals in the time and frequency domain. For evaluation metrics, they showed that the combination of Frechet inception distance (FID) and sliced Wasserstein distance (SWD), Euclidean distance (ED) can give
a good idea about its overall properties. Golany et al. [44] proposed a simulator-based GANs for ECG synthesis to improve a supervised classification. They incorporated ECG simulator equations into the generation networks, and then the generated ECG signals are used train a deep network.

Chen et al. [19] proposed GAN framework for building occupancy modelling. They first learned the discriminator and generator in the vanilla GAN with the training occupancy data. Then, the learned generator is the required occupancy model which can be used to generate occupancy data with random inputs. To evaluate, they defined five variables (i.e., mean occupancy, time of the first arrival, time of the last departure, cumulative occupied duration and the number of occupied/unoccupied transitions) with two criteria (i.e., normalised root mean squared error and total variation distance).

Che et al. [17] used a modified GAN called ehrGAN to generate plausible labelled EHR data. The generator is a modified encoder-decoder CNN network and the generated EHR data mimics the real patient records which augments the training dataset in a semi-supervised learning manner. In this work, they used the generative networks with the CNN prediction model together to improve the performance of risk prediction.

Koochali et al. [72] proposed ForGAN to predict the next-step time series value $X_{t+1}$ by learning the full conditional probability distribution. They applied a conditional GAN and the condition windows are the previous $t$ values $(X_0, X_1, ..., X_t)$. With the input of noise vector, the generator predicts the values at $t + 1$ step and then the discriminator compared this value to the true value at $t + 1$ step with the same condition windows. LSTM network is used in both generator and discriminator. Zhou et al. [167] predicted the stock price at next time step $y_{t+1}$ based on the features in previous $t$ time step $X_1, X_2, ..., X_t$ and previous stock price $y_1, y_2, ..., y_t$ using generative adversarial nets.

Instead of generating a sequence of single values, Dang et al. [25] developed an approach for the generation of adversarial attacks where the output is a sequence of probability distributions. The proposed approaches are demonstrated on two challenging tasks including the prediction of electricity consumption and stock market trading. Besides, AOSeRec [162] were proposed to generate a sequence of items consistent with user preferences rather than the next-item prediction. The model integrated the sequence-level oracle and adversarial learning into the seq2seq auto-regressive learning.

Generally, a good time-series generative model should preserve temporal dynamics, and the generated sequences should follow the original patterns between variables across time. Therefore, Yoon et al. [155] proposed a framework TimeGAN for producing realistic multivariate time-series, combining the flexibility of the unsupervised GAN approach with the control afforded by supervised learning. In addition to the traditional unsupervised adversarial loss on both real and fake data, they presented a stepwise supervised loss with the original data as supervision, which helps learn from the transition dynamics in real sequences.

4.2.2 Imputation. In real-world applications, time series are usually incomplete due to various reasons, and the time intervals of observations are usually not fixed [93]. The missing values in time series make it hard for effective analysis [41]. One of the popular ways to handle the missing values of time series is to impute the missing values to get the complete dataset. Generally, there are three different ways for time series imputation: case deletion methods [67], statistical imputation methods [47], and machine learning based imputation methods [11]. However, all the existing approaches hardly consider the temporal relations between two observations. In recent years, researchers have started to take advantages of GANs to learn latent representations between observations for time series imputation [92–94].
Fig. 9. An overview of the time series imputation framework proposed by Luo et al. [93]

Luo et al. [93] applied the adversarial model to generate and impute the original incomplete time series. To learn the latent relationships between observations with non-fixed time lags, a novel RNN cell called GRUI was proposed which takes into account the non-fixed time lags and fades the influence of the past observations determined by the time lags. They proposed a two-stage model (see Figure 9) for time series imputation: In the first stage, they adopted the GRUI in the discriminator and generator in GAN to learn the distribution and temporal information of the dataset. In the second stage, for each sample, they tried to optimise the 'noise' input vector and find the best-matched input vector of the generator. The noise was trained with a two-part loss function: masked reconstruction loss and discriminative loss. Masked reconstruction loss is the masked squared errors of the non-missing part between the original and generated sample. It means that the generated time series should be close enough to the original incomplete time series. The discriminative loss forces the generated sample as real as possible. However, this two-stage model needs a huge time to find the best-matched input vector which is not always the best especially when the initial value of the 'noise' is not properly set.

Then, Luo et al. [94] proposed an end-to-end GAN-based imputation model E^2GAN which not only simplifies the process of time series imputation but also generates more reasonable values for the filling of missing values. E^2GAN takes a compressing and reconstructing strategy to avoid the 'noise' optimisation stage in [93]. As seen in Fig. 10, in the generator (a denoising auto-encoder), they added a random vector to the original sample and map it into a low-dimensional vector. Then they reconstructed it from the low-dimensional vector. The generator seeks to find a network structure that can not only best compress and reconstruct the multivariate time series but also fools the discriminator. Then they used the reconstructed sample to impute the missing values.

Non-Autoregressive Multiresolution Imputation (NAOMI) [92] is a new model for the imputation of spatio-temporal sequences like traffic flow data and movement trajectories when arbitrary missing observations are given. NAOMI impute missing values for spatio-temporal sequences recursively from coarse to fine-grained resolutions with a non-autoregressive decoding procedure, and it further employs a generative adversarial learning process to reduce variance for improving the performance.

4.3 GANs for Spatio-temporal Graph Modelling

In this subsection, we will introduce the application of GAN on the graph data analysis which mainly focus on two areas: temporal link prediction and graph representation.
4.3.1 Temporal Link Prediction. Temporal link prediction refers to the dynamics prediction problem in network systems (e.g., mobility and traffic prediction) where system behaviours are described by the abstract graphs [77]. Given the snapshots of a graph in previous timestamps, the temporal link prediction task aims to construct the graph topology at the next timestamp. Lei et al. [77] proposed GCN-GAN to predict links in weighted dynamic networks. They combined graph convolutional network (GCN), long short-term memory (LSTM) as well as generative adversarial network (GAN). The generator consists of a GCN hidden layer, LSTM hidden layer and a fully-connected layer. Discriminator contains a fully-connected feed-forward network. For evaluation, they used edge-wise KL divergence and mismatch rate besides mean square error (MSE). Then, Yang et al. [153] designed an attentive GCN model for temporal link prediction in graphs using GAN. Compared to [77], attentive GCN allows for assigning different importance to the vertices to learn the spatial features of the dynamic network. Then, temporal matrix factorisation (TMF) LSTM was employed to capture the temporal dependencies and evolutionary patterns of dynamic networks. GAN framework was then proposed to improve the performance of temporal link prediction.

4.3.2 Graph Representation. Wang et al. [143] proposed GraphGAN unifying two types of graph representation methods: discriminative methods and generative methods via adversarial training. They found that the traditional softmax function and its variants are not suitable for the generator for two reasons: 1) softmax treats all vertices equally in the graph for a given vertex and does not consider the graph structure and proximity information; 2) the calculation of softmax involves all vertices in the graph which is time-consuming and computationally inefficient. Therefore, they introduced graph softmax as the implementation of the generator and proved that it satisfies the desirable properties of normalisation, computational efficiency and graph structure awareness.

Aiming at better capturing the essential properties and preserving the patterns of real graphs, Bojchevski et al. introduced NetGAN [14] to learn a distribution of network via the random walks. The merits of using random walks is their invariance under node reordering and efficiency in exploring the sparse graphs by merely traversing the nonzero entries. The results confirmed that the combination of longer random walks and LSTM is advantageous for the model to learn the topology and general patterns in the data.
Adversarial Network Embedding (ANE) [24] also considers random walk mechanism to learn network representation with the adversarial learning principle. It consisted of two components: 1) the structure-preserving component is developed to extract network structural properties via the usage of either Inductive DeepWalk or Denoising Autoencoder; 2) the adversarial learning component contributes to learning network representations by matching the posterior distribution of the latent representations to given priors. However, using DeepWalk for learning graph embedding could lead to overfitting issue due to sparsity is common in networks or increasing computational burden when more sampled walks are considered [158]. Therefore, NetRA [158] was proposed to further minimise network locality-preserving loss and global reconstruction error with a discrete LSTM Autoencoder and continuous space generator, in such that the mapping from input sequences into vertex representations could be improved.

Most recently, GAN embedding (GANE) [59] tries to gain the underlying graph distribution based on the probability distribution of edge existence which is similar to GraphGAN. The difference is that this model applies Wasserstein-1 distance as the overall objective function and intents to achieve link prediction and network embedding extraction simultaneously. As a novel network embedding method, the proximity generative adversarial network (ProGAN) [36] is proposed to capture the underlying proximity between different nodes by approximating the generated distribution of nodes in a triplet format to the underlying proximity in the model of GAN. Specifically, a triplet can encode the relationship among three nodes including similarity and dissimilarity. After the training of the generator and discriminator, the underlying proximities discovered are then used to build network embedding with an encoder.

The works mentioned above primarily focus on the single-view network in learning network embedding. However, numerous real-world data are represented by multi-view networks whose nodes have different types of relations. Sun et al. [138] introduced a new framework for multi-view network embedding called MEGAN, which can preserve the information from individual network views, while considering nodes connectivity within one relation and complex correlations among different views. During the training of MEGAN, a pair of nodes are chosen from the generator based on the fake connectivity pattern across views which is produced by multi-layer perceptron (MLP), and the discriminator is then executed to differentiate the real pair of nodes from the generated one.

### 4.4 GANs for Trajectory Prediction

Trajectory prediction refers to the problem of estimating the future trajectories of various agents based on the previous observations [96]. Gupta et al. [53] proposed SocialGAN to jointly predict trajectories avoiding collisions for all people. They introduced a variety loss encouraging the generative network of the GAN to spread its distribution and cover the space of possible paths while being consistent with the observed inputs. A new pooling mechanism was proposed to learn a “global” pooling vector which encodes the subtle cues for all people involved in a scene. In GD-GAN [34], Fernando et al. designed a GAN based pipeline to jointly learn features for both pedestrian trajectory prediction and social group detection. As the basic GAN structure used in SocialGAN is susceptible to mode collapsing and dropping issues, Amirian et al. [5] extended the SocialGAN by incorporating the Info-GAN [18] structure in their Social Ways trajectory prediction network.

SoPhie, proposed by Sadeghian et al. [126], is another GAN based trajectory prediction approach which can take both the information from the scene context and social interactions of the agents into consideration. Two separate attention modules are also used to better capture the scene context and the social interactions. More recently, based on BicycleGAN [168] framework, Social-BiGAT [73] develops the bijection function between the output trajectories and the latent space.
input to the trajectory generator. It also uses a Graph Attention Network in combination with a VGG network [137] to encode social influence from other pedestrians and semantic scene influence of the environment. In order to generate trajectories with fewer potential collisions, CoL-GAN [89], proposed by Liu et al., exploits a CNN-based network as the trajectory discriminator. Different from other GAN based trajectory prediction methods such as SocialGAN [53] and SoPhie [126], the proposed discriminator is able to classify whether each segment of a trajectory is real or fake.

Recently, Gao et al. [40] studied the trajectory user linking problem to identify user identities from mobility patterns. They combined autoencoder with GANs for jointly human mobility learning, which provides regularized latent space for mobility classification. APOIR [166] were developed to learn the distribution of underlying user preferences in the Point-of-interest (POI) recommendation. It consists of two components: the recommender and discriminator. The recommender approaches the true preference of users and the discriminator distinguishes the generated POIs from the truly visited ones.

5 DISCUSSION

5.1 Challenges and Future Directions

Alongside numerous advantages of GANs, there are still challenges needed to be solved for employing GANs in ST applications. The traditional architectures and loss functions of GANs may not be suitable due to the unique properties of ST data. Besides, evaluating ST data is more difficult compared to images where researchers could rely on the visual inspections. Therefore, we will mainly focus on: (1) how to modify architectures/loss functions of GANs to better capture the spatial and temporal relations for ST data? (2) how to evaluate the performance of GANs especially when visually inspecting the generated ST samples is not applicable? We will then address these two problems and indicate the future directions of investigating this area.

5.1.1 Architectures and loss functions of GANs. In the computer vision area, fully connected layers were initially used as building blocks in vanilla GAN, but later on were replaced by convolutional layers in DCGAN [117]. Compared with images with only spatial relations, modelling ST data is more complex due to the constraints from both spatial and temporal dimensions. Therefore, adapting architectures and loss functions of GANs for specific ST applications have become the mainstream recently.

Generally, original or adapted RNN [33, 93, 102], LSTM [14, 72, 77, 80, 158], VAE [17, 94, 130, 158], CNN [17], GNN [77] are usually used as the base model (i.e., the discriminator and generator) in the vanilla GAN, WGAN [55] or CGAN [72], which captures the spatio-temporal relations for the spatio-temporal data. What’s more, some new loss functions have been proposed to dealing with specific ST tasks, such as the stepped supervised loss in TimeGAN [155], masked reconstruction loss in GRU-GAN [93], the variety loss in SocialGAN [5]. With further developments of GANs for ST data, new architectures and loss functions can be designed based on the characteristics of ST tasks.

5.1.2 Evaluation Metrics. Though GANs have gained huge success in various fields, evaluating the performance of GANs is still an open question. As illustrated in [15] and [60], both quantitatively measures (e.g., Log-likelihood with Parzen Window Estimation [129], Fréchet Inception Distance [56], Maximum Mean Discrepancy [48]) and qualitative measures (e.g., Preference judgement [144], Analysing Internals of Models [117]) have strengths and limitations. The nebulous notion of quality can be best assessed by a human judge, which is neither practical nor appropriate for different types of ST data.
In most cases, it is not easy or even possible to visually evaluate the generated ST data. For instance, the Intense Care Unit (ICU) time series [33] or heart rate Electrocardiogram (ECG) [44] signals could look completely random to a non-medical expert. Usually, the evaluation of generated ST samples requires domain knowledge. For example, Mogren et al. [102] evaluated the generated music sequences using metrics in the field of music such as polyphony, repetitions, tone span and scale consistency. For future ST applications with GANs, some novel metrics based on the domain knowledge could be considered for the evaluation of generated ST data.

Especially, some researchers have proposed the general approach to evaluate the generated ST-data. Esteban et al. [33] developed a general method called Train on Synthetic, Test on Real (TSTR) to evaluate the generated samples of GANs when a supervised task defined on the training data. They used a dataset generated by GANs to train a classification model, which is then tested on a held-out set of true samples. This evaluation metric is ideal when employing GANs to share synthetic de-identified data because it demonstrates the ability of the generated synthetic data to be used for real applications. In the future, more practical metrics should be developed to evaluate the performance of generated ST samples.

5.2 Conclusions
In this survey, we conducted a comprehensive overview of Generative Adversarial Networks (GANs) for spatio-temporal (ST) data in recent years. Firstly, we discussed the properties of ST data and traditional ways for ST data modelling. Then, we have provided a thorough review and comparison of the popular variants of GANs, and its applications on ST data analysis, such as time series imputation, trajectory prediction, graph representation and link prediction. Besides, we summarised the challenges and future directions for employing GANs for ST applications.

Finally, we would like to point out, though there are many promising results in the literature, the adoption of GANs for ST data is still in its infancy. This survey can be used as the stepping stone for future research in this direction, which provides a detailed explanation of different ST applications with GANs. We wish this paper could help readers identify the set of problems and choose the relevant GAN techniques when given a new ST dataset.

6 ACKNOWLEDGMENTS
This research was supported by the Australian Government through the Australian Research Council’s Linkage Projects funding scheme (LP150100246) and Discovery Project (DP190101485).

REFERENCES


Generative Adversarial Networks for Spatio-Temporal Data: A Survey

111:23


[84] Zhenhui Li, Bolin Ding, Jiawei Han, Roland Kays, and Peter Nye. 2010. Mining periodic behaviors for moving objects. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1099–1108.


and Communications (PerCom). IEEE, 192–201.


