Management and Production Engineering Review
Volume 16 • Number 4 • December 2025 • pp. 1–12
DOI: 10.24425/mper.2025.157218

Artificial Data Generation in the Implementation of Digital Twins: Techniques and Applications

Francisco ZENZA[1], Ana L. RAMOS[1], José V. FERREIRA[1], Luís P. FERREIRA[2], Ricardo RIBEIRO[3]

[1] *GOVCOPP, DEGEIT, University of Aveiro, Aveiro, Portugal*
[2] *LAETA/INEGI, ISEP, Polytechnic of Porto. Dr. António Bernardino de Almeida, 431. 4249-015, Porto. Portugal*
[3] *Efacec Energia – Máquinas e Equipamentos Eléctricos, S.A., 4466-925 S. Mamede de Infesta, Portugal*

Received: 20 July 2025
Accepted: 18 Oktober 2025

Abstract
The growing dependence on high quality data in industrial environments drives the adoption of artificial data generation techniques, especially in the development and implementation of Digital Twins (DTs). This article presents a critical review of the main approaches to creating synthetic data, with an emphasis on their application in Industry 4.0 intelligent cyber-physical systems. Initially, traditional techniques such as Random Oversampling (ROS), SMOTE and its variants are analyzed, as well as statistical models such as the Gaussian Mixture Model (GMM). Next, Deep Learning-based methods are explored, namely Autoencoders, Variational Autoencoders and Generative Adversarial Networks (GANs), highlighting their ability to produce realistic and diverse data. The study also includes the analysis of practical cases in which DTs have been developed using synthetic data, covering domains such as wind energy, aviation and urban infrastructure. In this way, the aim of this study is to critically explore the different techniques of artificial data generation with an integration with the technology of DTs. The results suggest that the appropriate use of synthetic data can not only overcome limitations related to privacy and the scarcity of real data, but also improve the robustness and effectiveness of Digital Twins models. The article concludes by discussing the current challenges and future opportunities in integrating these techniques into smart industrial environments.

Keywords
Synthetic Data, Digital Twins, Artificial Data Generation, Machine Learning, Industry 4.0.

Introduction

Data is present everywhere and can be a source of great value. However, to generate this value, the data needs to be of high quality. Furthermore, when working with sensitive data, such as financial and medical databases, the privacy of this data must be safeguarded without sacrificing quality. The lack of high-quality data and the need for privacy-preserving data has become increasingly evident in recent years, as companies and researchers use this data more and more (Figueira & Vaz, 2022).

Initially proposed by Rubin (1993) to solve these issues of privacy and the availability of sensitive data, synthetic data consists of artificially generated data (El Emam, Mosquera, & Hoptroff, 2020) and its use is a very effective tool for solving the two problems mentioned above. As synthetic data is generated rather than collected or measured, it can be of higher quality than real data. In addition, privacy restrictions can be applied so that synthetic data does not reveal any important or sensitive information. In the context of integration with Industry 4.0 technologies, synthetic data also solves the problem of collecting enough real data to train supervised machine learning (ML) models. This type of data consists of an unlimited source of balanced and varied data and can simulate rare events and situations that have not yet been faced (Aranjuelo et al., 2021).

Recently, with the advent of the new industrial revolution, new manufacturing systems have emerged, such as Digital Twins (DTs) (Jaskó et al., 2020). This new era of intelligent manufacturing systems, based on cyber-physical systems, is disruptive in a variety of aspects of traditional manufacturing companies (Almada-Lobo, 2015). In a highly digitalized, globalized and

Corresponding author: *Luís Pinto Ferreira – LAETA/INEGI, ISEP, Polytechnic of Porto. Postal Address: Dr. António Bernardino de Almeida, 431. 4249-015, Porto. Portugal., phone: (+351) 22 83 40 500, e-mail: lpf@isep.ipp.pt*

volatile disruptive environment, companies seek to be increasingly competitive (Mourtzis, 2021). In this way, a DT can be considered an important tool for responding to this challenge, as a DT model can represent system changes over time (Lopes et al., 2024).

A DT is a virtual model, synchronized with a physical model through real-time data, in order to allow the simulation and analysis of the system's performance, behaviours and potential results (Segovia & Garcia-Alfaro, 2022). A DT is based on mathematical models that represent physical phenomena, making it possible to understand the behavior of a real asset at any given time. In addition, by using this physics-based model, it is possible to create synthetic data for events that have never occurred, acquiring knowledge about the behavior of the system under certain conditions that would otherwise not be possible to study. Data-based models can identify and prevent events that have been measured in the past. However, the training process for data-driven algorithms always depends on historical data. In contrast, DTs provide two new sources of information: first, physics-based models make it possible to understand their actual behavior, and second, physical simulation allows synthetic data to be generated for new potential scenarios, such as possible anomalies and failure conditions (Pujana et al., 2023). Furthermore, hybrid models, being those that combine physics-based models with data analysis, provide a powerful tool for diagnosis and prognosis. Hybrid models developed for this purpose have a good basis for creating a DT (Mishra et al., 2015).

In this article, the aim of the authors is to draw up a review of the main methods and techniques for generating synthetic data, and how these can be implemented in DTs methodologies, analyzing relevant techniques and case studies. To this end, this article is organized as follows: the "Artificial Data Generation Techniques" section presents and analyses various artificial data generation techniques, from classic approaches such as SMOTE and ROS to advanced methods based on Deep Learning, such as Autoencoders and Generative Adversarial Networks. The "Digital Twins with Artificial Data Generation" Section explores the application of these techniques in the context of DTs, highlighting use cases in real industrial domains and integration methodologies with synthetic data. The "Discussion" section critically discusses the impact, limitations and challenges associated with adopting these techniques in cyber-physical environments, with a special focus on their practical validity and relevance to Industry 4.0. Finally, the "Conclusions" section presents the main findings of the work and points to directions for future research.

Artificial Data Generation Techniques

According to El Emam et al. (2020), synthetic data can be generated from real data, from existing models, using expert knowledge of a given domain, or through a mixture of these options. Synthetic samples generated from real data can be obtained by building a model that captures the properties (distribution, correlation between variables, etc.) of the real data. Once the model has been created, it can then be used to deliver synthetic data.

Synthetic data generated from existing models consists of instances generated through statistical models (mathematical models that have statistical assumptions about how the data should be generated) or through simulations (video game engines that create images from objects). The use of specific knowledge about a given domain can also be applied to the generation of synthetic data. For example, knowledge about how the financial market works can be used to create an artificial database of stock prices. However, this action involves a great deal of knowledge about the domain in question, so that the synthetic data is similar to real data (Figueira & Vaz, 2022).

Synthetic data is usually generated from raw data, i.e. unprocessed, and the most appropriate approach is for the new data to be modeled from its original form (Patki et al., 2016), before any modifications are applied, and for missing values to be represented, as these provide information about the database (Ping et al., 2017). In this way, synthetic data must be generated properly, so that the data is plausible and follows the underlying distribution of the original data (Figueira & Vaz, 2022).

Many of the current problems with artificial intelligence (AI) are the result of insufficient data, poor quality data or unlabeled data. This phenomenon is practically widespread, as many areas of study suffer from the same difficulties, such as physics (Alanazi et al., 2021), finance (Assefa et al., 2020), health (Lan et al., 2020), sports (Chen & Little, 2019), and agriculture (Barth et al., 2017). As a result, there is increasing interest in the usefulness of synthetic data and the challenges that this technology can overcome.

Synthetic data is artificially generated from real data, and has the same statistical properties as the original data. However, unlike real data, which is measured or collected in the real world, synthetic data is generated using algorithms hat Is Synthetic Data? |NVIDIA Blogs, n.d.). Through these, the artificial generation methods must produce this new data in a robust way, in order to capture the patterns of the real databases. The evolution of these tools has been marked by a se-

ries of techniques intended to come as close as possible to recreating the data as faithfully as possible. One of the oldest algorithms, SMOTE (Chawla et al., 2002) (Synthetic Minority Oversampling Technique), was proposed in 2002 as a technique that attempts to replicate the distribution of data, in addition to others such as ROS (Random OverSampling) and rotation or scaling algorithms. The technique has evolved over time, and over the years, other variants were proposed (Figueira & Vaz, 2022). However, it was only with the advent of Deep Learning that the most promising ideas emerged, such as VAEs (Kingma & Welling, 2013) (Variational AutoEncoders) in 2013 and, more importantly, GANs (Goodfellow et al., 2014) (Generative Adversarial Networks) the following year.

In his book, Nikolenko (2021) argues that synthetic data is essential for the development of Deep Learning, specifically in the potential application of many case studies. The author also proposes three main directions one can take when looking to apply synthetic data in ML: use synthetic data to train ML models and use them to make predictions about real-world data; use synthetic data to extend existing real-world databases, mainly to cover under-represented parts of the data distribution; and to solve legal and ethical problems by generating artificial data, which is thus anonymized. The following sections will therefore give a more detailed account, based on the work of Figueira and Vaz (2022), of each of the artificial generation techniques and algorithms mentioned above, as well as other algorithms and their variants.

Standard techniques

This subsection presents some of the main methods and techniques for generating synthetic data. This first part differs from the subsequent ones in that it will present the algorithms most commonly used before the success of *Deep Learning* generative models, hence the label "standard techniques". This presentation will be guided by the level of sophistication of the algorithm, so ROS will be analyzed, followed by SMOTE and its variants, then *cluster-based* oversampling algorithms, and finally GMM.

- *Random Oversampling (ROS)*

The ROS algorithm adds additional observations to the database by randomly sampling minority classes. This is probably the simplest and most straightforward algorithm for expanding the data set. Even so, this approach can change the distribution of the data. Thus, if a classifier is fed such data, it could learn from an incorrect distribution. Furthermore, because ROS duplicates observations, it implies that this algorithm does not create new synthetic samples, but only replicas of existing data, which has led to the development of more advanced techniques such as SMOTE.

In their work, the authors Batuwita and Palade (2010) compared the use of ROS with RUS (Random Undersampling), this algorithm consisting of the reverse of ROS, where instances of the original data are removed from the majority classes. In the end, it was revealed that ROS returns more satisfactory classification results than RUS. In another paper, authors (Drummond and Holte (2003) also compare the two methods and, among other conclusions drawn, come to the conclusion that ROS, in most cases, proves to be ineffective, producing little or no change in classification performance.

- *Synthetic Minority Oversampling Technique (SMOTE)*

SMOTE is also an oversampling algorithm where synthetic observations are generated (and not duplicated, as in ROS) from the minority classes. It works by selecting a data point from the minority class and its nearest neighbor. The distance between the two is then calculated and multiplied by a random number between 0 and 1. This value is then added to the originally selected data point, so that the new sample is on the same line as the original point and its neighbor. This process is repeated until there are the desired number of samples.

This algorithm is very popular in the literature. For example, in the work by Blagus and Lusa (2012), the authors evaluate the use of SMOTE on a high-dimensional dataset, showing that this technique does not mitigate the bias towards majority classes for most classifiers. However, for K-th nearest neighbor classifiers based on Euclidean distance, SMOTE can be beneficial if the number of variables is reduced by variable selection.

Despite being a more advanced technique than ROS, SMOTE still has certain problems, such as focusing on instances of minority classes, thus ignoring majority classes, or altering the authentic distribution of data. Based on this, some variants of this algorithm have been developed to overcome these problems, namely Borderline-SMOTE, Safe-Level-SMOTE, and ADASYN (Adaptive Synthetic Sampling Approach).

- *Borderline-SMOTE*

In their work, Han et al. (2005) proposed two algorithms as a variation of SMOTE: Borderline-SMOTE1, which only generates samples from the minority classes near the edges, and *Borderline-SMOTE2*, which also takes into account the observations of the majority classes. The first Borderline-SMOTE only considers the data points of the minority classes that have a number of neighbors of the minority classes in the interval

$[m/2, m]$, where m *is* defined by the user. These are the data points that could be incorrectly classified. After detecting these observations, SMOTE is applied to create synthetic samples. The second Borderline-SMOTE is similar, with the difference that it also considers the neighbors of the majority classes. According to the authors, these variations of SMOTE offer improvements over the original SMOTE and ROS, in terms of the true positive rate and the F-value.

- *Safe-Level-SMOTE*

SMOTE synthesizes the minority class samples along a line connecting the minority class instance to its nearest neighbors, ignoring the majority class instances. Instead, Safe-Level-SMOTE (Qian et al., 2009), defines safe regions to prevent oversampling in overlapping and noisy regions with majority classes, providing better accuracy performance than the original SMOTE and Borderline-SMOTE.

Each minority class instance is assigned a security level defined by the number of minority class instances in the K nearest neighbors, K being user-defined. Each synthetic instance is positioned near the highest security level so that all synthetic instances are created in the secure regions. Intuitively, when the algorithm is given a data point, p, from the minority class and its nearest neighbor, n (from the same class), Safe-Level-SMOTE will generate a synthetic sample closer to p if its security level is higher than the security level of n, and vice versa. In other words, the synthetic sample will be closer to the data point that has the most neighbors close to the minority class. Hence, this algorithm offers a more insightful solution than the original SMOTE, in that it does not only generate a random instance on the line segment between two data points of the minority class, but also considers their neighborhoods.

- *Adaptive Synthetic Sampling Approach (ADASYN)*

The ADASYN technique is an oversampling algorithm that improves the learning performance of its classifiers (He et al., 2008). It uses a weighted distribution for different instances of the minority class, taking into account their level of difficulty for a classifier to learn, i.e. samples of the minority class that have a smaller number of neighbors of the class are more difficult to learn than those with more neighbors of the same class. Thus, more synthetic samples are generated for the minority class examples that are harder to learn and fewer synthetic samples are generated for the minority class examples that are easier to learn.

The ADASYN algorithm is similar to SMOTE in that it generates synthetic samples on line segments between two data points from the minority class. The difference is that ADASYN uses the density distribution as a criterion to automatically determine the number of synthetic samples to generate for each instance of the minority class. From there, the expanded database provides a balanced representation of the data distribution, and forces the classifier to pay more attention to the most difficult-to-learn examples.

While the ADASYN, *Safe-Level*, and *Borderline* algorithms are all variants of SMOTE, it is also possible to not modify the SMOTE algorithm and instead use an unsupervised algorithm before applying SMOTE. Clustering algorithms are a type of unsupervised algorithm that can be very useful in detecting structures in the data (also in dividing the data into classes). When applied well, *clustering* algorithms can reveal hidden patterns in the database that were previously undetectable.

- *K-Means SMOTE*

The K-Means SMOTE algorithm was proposed in a paper by Douzas, Bacao and Last (2018), where they combine the popular K-means clustering algorithm MacQueen, 1967) with SMOTE, thus avoiding the generation of noise and efficiently overcoming imbalances between and within classes.

K-Means SMOTE consists of three steps. First, the observations are grouped (*clustered*) using the K-means algorithm. This step is followed by the filtering stage, where *clusters* with a small proportion of minority class instances are discarded. The number of synthetic samples to be created also depends on the *cluster*, i.e. *clusters* with a low proportion of minority class samples will have more synthesized instances. Finally, the SMOTE algorithm is applied to each of the *clusters*.

- *Cluster-Based Oversampling*

In their work, Jo and Japkowicz (2004) respond to the presence of small disjuncts during data training. Their work showed that performance loss in standard classifiers is not caused by class imbalance, but that this imbalance can lead to small disjuncts which, in turn, cause performance loss.

The Cluster-Based Oversampling algorithm consists of grouping the data for each class, i.e. each class is grouped separately in their work (Jo & Japkowicz, 2004), the authors used K-Means, but theoretically any clustering algorithm can be used), and then ROS is applied to each cluster. For the majority class clusters, more specifically, all the clusters except the largest, the ROS is applied to them until they have the same number of observations as the majority class cluster. The minority class clusters are randomly oversampled until each cluster has $[m/N]$ samples, where m is the number of instances of the majority class, and N *is* the number of minority class clusters. This algorithm is similar to K-Means SMOTE in that both use clustering followed by oversampling, but they differ in some

respects. For example, K-Means SMOTE uses a specific clustering algorithm, K-means, and the classes are not grouped separately, while *Cluster-Based Oversampling* allows the user to freely choose the clustering algorithm, and the classes are grouped separately. In addition, K-Means Clustering uses the SMOTE oversampling technique, while Cluster-Based Oversampling uses the ROS algorithm.

The methods mentioned so far, with the exception of ADASYN, tend to neglect the distribution of the original data. Therefore, the logical but different approach would be to model the underlying distribution of the data and draw a sample from that set. However, estimating such a distribution is an extremely laborious problem, especially as the number of features in the data increases and simplifications need to be made.

- *Gaussian Mixture Model (GMM)*

As the latest technique in the standard algorithms for generating artificial data, GMM is a probabilistic model that assumes that the data can be modeled by a weighted sum of a finite number of Gaussian distributions (Gaussian mixture models, 2025). Hence, the resulting model is given by the function (1):

$$p(x) = p_1 p_1(x) + p_2 p_2(x) + ... + p_n p_n(x) \qquad (1)$$

In this equation, in the univariate case, $p_i(x)$ is the probability density function of a univariate normal distribution with a mean and standard deviation i, $p_i(x)$ is the weight assigned to each $p_i(x)$, and n is the number of components. The number of components, n, is determined by the user, and the parameters of the means, standard deviations and weights are estimated, typically through an expectation-maximization algorithm, which is an iterative, statistics-based algorithm that calculates the probability with which each point is generated by each component, and then changes the parameters in order to maximize the probability of the data. For the multivariate case, $p_i(x)$ is replaced by a multivariate normal distribution, $N_k(\mu_\mathrm{i} \sum_\mathrm{i})$, where k is the dimension of the multivariate normal distribution, μ_i is a vector of means, and $\sum_i$ is a covariance matrix. With the model determined, synthetic data can be generated by taking random samples from it.

Deep learning techniques

Artificial data generation methods with *Deep Learning* are so called because they use techniques from this area to create new instances. Unlike standard methods, *Deep Learning* models are more difficult to understand due to their greater level of complexity and are generally not able to be interpreted. Despite this, this subsection will explain the three main classes of generative models: *Bayesian Networks* (BNs), *Autoencoders* (AEs) and Generative *Adversarial* Networks (GANs). There are countless variations of these algorithms with a wide range of architectures for specific application sectors. The following is a brief overview and description of the main advanced data generation techniques.

- *Bayesian Networks (BNs)*

Although Bayesian networks may not be considered Deep Learning, they are easily considered to be Bayesian neural networks, which are Deep Learning structures (Baan, 2021). A Bayesian network is a type of probabilistic graphical model that uses Bayesian inference for probabilistic computations on a directed acyclic graph. This model is used to represent the dependence between variables, so that any joint probability distribution can be represented, and in many cases succinctly (Russell & Norvig, 2016). In a Bayesian network, each node corresponds to a random variable (which can be discrete or continuous) and contains probability information that quantifies the effect of the node's parents. If there is a connection between the node x_i and x_j, then x_i has a direct impact on x_j. Furthermore, if there is a path between the nodes x_i and x_j (such as at least one node in between them), then x_i also has an influence on x_j.

- *Autoencoders (AEs)*

An autoencoder is a special type of direct neural network that consists of two parts: an encoding network that learns to compress the data from high dimension to low dimension, resulting in a latent spatial representation (the code), and a decoding network that decompresses this compressed representation into the original domain (Foster, 2022). Formally, an encoder can be seen as a function, $c = E(x)$, that produces a low-dimensional representation of the data, and a decoder as a function, $r = D(c)$, that produces the reconstruction of the code. The goal is not for the AE to learn to implement $D\left(E\left(x\right)\right) = x$ for every input of x, but rather for it to learn to approximately copy the original data, and only those inputs that approximate the original data. By constraining and forcing the algorithm to learn which aspects of the data are priorities, autoencoders can learn useful properties about the data (Goodfellow et al., 2017).

In terms of generating synthetic samples, EAs have a few problems. First, the learned distribution of latent space points is undefined, meaning that when a data point is sampled from latent space and decoded to generate a new example, there is no guarantee that it will be a plausible example. Secondly, there is a lack of diversity in the samples generated. Furthermore, data points belonging to the same class may have larger gaps

in the latent space, which can lead to faultily generated instances when samples are drawn from their neighborhood. To overcome these challenges, VAEs can be used.

Variational autoencoders were first proposed in the work of Kingma and Welling (2013), and consist of a natural expansion of autoencoders with a focus on solving the aforementioned problems. VAEs improve on the base models with changes in the coding function and the loss function. The coding process of a VAE maps each point of the original data into a multivariate normal distribution in latent space, represented by the mean and variance vectors. VAEs assume that there is no correlation between the two dimensions in the latent space, hence the covariance matrix does not need to be calculated because it is diagonal. This small change ensures that a point, a, sampled from the neighborhood of another point, b, in latent space, is similar to point b. Thus, a point in latent space that is completely new to the decoder can, with high probability, deliver a correct sample.

The loss function in the VAE model adds the Kullback-leibler (KL) divergence to the autoencoder reconstruction function. This loss function provides a well-defined distribution that can be used to sample data points in latent space. Sampling from this distribution guarantees, with greater probability, that the points sampled are in a region where the decoder must decompress. In addition, the intervals between points in the latent space will also be smaller. With this, the inclusion of KL in the loss function, and with the change in coding, a VAE thus has a better structure for generating digital samples.

- *Generative Adversarial Networks (GANs)*

Proposed by Goodfellow et al. (2014), generative adversarial networks are structures that use an adversarial process to estimate generative deep learning models. These structures have been adapted and improved in recent years, to the point where they are very powerful and versatile. Currently, GANs are capable of painting, writing and composing. A GAN consists of two models: a generator model (G) that tries to produce samples that follow an underlying distribution of the data (these observations are suitably different from the observations in the database, i.e. the model must not reproduce observations that have already occurred in the database). The second model is the discriminator (D) which, given an observation (from the original database or the data synthesized by the generator), classifies it as a false copy (produced by the generator) or true. An important point to consider is that the generator and the discriminator compete against each other. While the generator creates data points similar to the data in the original set in order to fool the discriminator, the discriminator tries to distinguish the generated value from the real observation.

In order to describe the training of these networks in more detail, this process is divided into training the discriminator and training the generator. Training the discriminator consists of creating a database with instances created by the generator and data points from the original data set. The discriminator delivers a probability, i.e. a continuous value between 0 and 1 that indicates whether a given observation comes from the original dataset (0 means that the discriminator is 100% sure that the observation has been synthesized, while 1 means the complete opposite).

Training the generator is more complicated. The generator is given, as input, random noise (the term latent space is used to designate the generator's input space) from a multivariate normal distribution, and as output, it is given a data point with the same characteristics as the original database. However, there is no database to inform whether a particular point in the latent space is mapped by the generator into a useful or reasonable example. Thus, the generator is only given a value of the loss function. This is usually the binary cross entropy between the output of the discriminator.

Given the discriminator's response, i.e. the value of the loss function, the generator attempts to improve it so that it can better fool the discriminator. As training continues, the generator uses the output of the discriminator to generate better examples, i.e. examples that better represent the actual distribution of the data. As the data produced by the generator becomes more and more realistic, the discriminator also improves its ability to determine whether a sample is real or synthetic. With this, both networks improve each other and, ideally, the generator will be able to faithfully reproduce the distribution of the original data, and the discriminator will have the same probability of distinguishing a real observation from an artificially generated sample. In this ideal scenario, the generator has succeeded in recreating the distribution of the original data, completely fooling the discriminator. Since GANs were first introduced, many researchers have considered them to be a very powerful tool. As a result, this technique has been systematically modified and improved.

GANs are used in a wide range of applications, and successfully applied to imaging tasks. However, many databases have a tabular format, and the most popular GAN architectures cannot be used in such settings due to the particular properties of tabular data. Firstly, continuous and categorical features are present in most tabular databases. As image data only consists of numerical properties (pixels), GANs used for imaging tasks cannot accommodate different types of variables. Second, multimodal and non-Gaussian

distributions are common in tabular databases, which must be taken into account when synthetic data is to be generated. Thirdly, very unbalanced categorical variables are a common feature, which can lead to insufficient training for minority classes. Finally, it is easier for a generic discriminator to distinguish between false and original data when learning from one-hot-encoded spaced vectors, since it takes into account the density of the distribution, than it is to check the comprehensive authenticity of the sample. This is followed by a list of other variants of the GAN algorithm for tabular databases (Figueira & Vaz, 2022):

- Tabular Generative Adversarial Networks (TGAN): proposed in 2018 as an architecture for generating artificial tabular data. Given a data set, which is already divided into training and test sets, the aim of TGAN is, by providing a machine learning model, the accuracy of the test set when trained by the training set should be similar to its accuracy, but when it is trained with the synthetic data set, the mutual information between each pair of the original and synthetic columns should be similar (Xu & Veeramachaneni, 2018).
- Conditional Tabular Generative Adversarial Networks (CTGAN): also proposed in 2019 by the authors of the previous technique, CTGAN is an improvement on TGAN. The objectives of CTGAN are practically the same as those of TGAN, with the difference that this algorithm is more ambitious, as it preserves the joint distributions of all the columns, instead of only preserving the correlation of each pair of columns in the synthetic data (Xu et al., 2019).
- TabFairGAN: proposed in 2022, this is a WGAN (Wasserstein Generative Adversarial Networks) but with a gradient penalty. As with the TGAN and CTGAN algorithms, in TabFairGAN it is crucial to represent the data correctly before using it as input. To this end, one-hot-encoding is used to represent the categorical classes (Rajabi & Garibay, 2022).

Digital Twins with Artificial Data Generation

A manufacturing process is defined by the use of one or more physical mechanisms to transform the shape or properties of a material (Chryssolouris, 2013). In simpler terms, a manufacturing system connects elements for sounding, decision making and action taking (Beregi et al., 2021). Manufacturing systems, represented by a wide range of processes, can be considered complex manufacturing networks (Zhan et al., 2014). In Industry 4.0, an intelligent manufacturing system is a complex physical system that can be broken down into multiple digital models. This concept involves simulation, modeling capabilities, interoperability, IoT sensors, and computational infrastructure tools (Leng et al., 2021).

The behaviours and properties of the production system can be stored in databases, which can then be used to synchronize executable models and real systems (Friederich et al., 2022). These models can be called DTs. This context leads to an increased demand for larger amounts of data (Cochran et al., 2016). When the concept of DT is implemented, a system becomes a system with a life cycle. The main phases of this cycle are: design, development, use, support and retirement. A DT needs data synchronization, so it will need at least some dummy data for its initial phase to understand what kind of analyses might be appropriate and beneficial. However, collecting real data in all these phases is a demanding job that requires a lot of resources (Lopes et al., 2024).

Generating synthetic data is a useful and effective way of obtaining data. Some researchers fill this gap by generating data with large-scale models or simulation models (Friederich et al., 2022). However, there is still a need for solutions that solve problems specific to each system. The alternative is to approach these manufacturing systems as complex networks through the use of data generation strategies at scale.

Data Synchronization in Digital Twins

Today, industry is increasingly interested in the concepts of digital shadows and digital twins in its production lines, especially in the planning, monitoring and optimization processes (Shao, 2021). DTs are rigorous executable virtual models of physical assets or systems (Wright & Davidson, 2020). Similarly, shop floor operations can be described through the use of simulation models. By integrating both concepts in a manufacturing context, a simulation model can be used as a basic digital model for the development of a DT (Shao et al., 2019).

Multiple and different DTs can be generated for one or more physical assets, based on the requirements defined by the system. the mirrored object is used as the basis for creating the DT. In principle, the development of the DT based on the physical object can be a data-driven process or carried out manually. The latter would be considered a reverse engineering process (Tekinerdogan & Verdouw, 2020).

Geometric, physical, behavioral and collaborative models are descriptive in nature, while decision-making models are intelligent and data-based (Camarinha-Matos et

al., 2019). Thus, DTs are used to estimate the response of the physical system before an unexpected event is triggered (Schleich et al., 2017). The data that supports the modeling of smart factories can be grouped by state data, i.e. the records of the operational states of the system, event data, consisting of the chronological records of the most relevant events, and condition monitoring data, with the most relevant records of sensory data. These three categories are in a time series format, so each record consists of an observation and an indication of time (Friederich et al., 2022).

Three levels of abstraction can be distinguished in a DT: components, systems, and systems of systems. For example, a component can be a part, a system can be an engine, and a system of systems can be the entire factory floor. Developing a DT requires the presence of: a physical entity, a digital model of that entity, a data extraction and communication protocol between the physical and virtual systems, and data analysis techniques (van Dinter, Tekinerdogan, & Catal, 2022). Manual and data-driven processes offer distinct advantages in developing DTs for manufacturing systems. Data-driven methods can be cost-efficient by using existing data and reducing effort in the development phase. While the manual reverse engineering process enables more accurate models and requires more resources in the development phase. However, to make the most appropriate choice, you need to consider factors such as the availability and quality of the data, the desired level of detail, time constraints and budget (Lopes et al., 2024).

Application cases and their methodologies

The following case studies were selected to represent a diverse set of domains where DTs are being applied (i.e., energy, aviation, civil infrastructure, and urban systems). The selection criteria included recency of publication, representativeness of different industries, and explicit use of synthetic data in DT development. However, it should be noted that, at present, there are still few practical cases that explicitly integrate DTs with synthetic data generation.

This subsection will present some case studies carried out by a number of researchers, who have looked into the topics of creating DT models applied to a real asset, with the additional integration of artificial data generation mechanisms, tools and techniques. With this in mind, the purpose of this subchapter is to give a brief overview of some projects that have integrated the two topics mentioned so far in this work, in order to analyze the DT models built, the methodologies adopted for their data generators, and the main results obtained.

- *DT of a Wind Turbine Engine*

In the work by Pujana et al. (2023), the authors present a methodology for developing a DT based on a hybrid model of a wind turbine energy conversion system. This DT allows knowledge to be acquired from real operating data, preserving physical design relationships, generates synthetic data from events that have never happened and helps in the detection and classification of different fault conditions. Starting from an initial physics-based model of a wind turbine transmission system, which is trained with real data, the proposed methodology has two main innovative results. The first innovative aspect is the application of generative stochastic models coupled with a DT based on a hybrid model to create synthetic fault data based on real anomalies observed in SCADA (Supervisory Control and Data Acquisition) data. The second innovative aspect is the classification of faults based on automatic learning techniques, which make it possible to identify anomalous conditions in the operation of the wind turbine.

Firstly, the technique and methodology were contrasted and validated with operating data from a real wind farm owned by Engie, including labeled fault conditions. Although the selected use case technology is based on a double feeder induction generator (DFIG) and its corresponding part-scale power converter, the methodology could be applied to other wind conversion technologies.

- *DT of a Jet Engine*

In the work by Aghazadeh Ardebili et al. (2023), the authors focus on data magnification through the generation of synthetic data. They also concluded that this approach can facilitate the development of a DT if developers do not have enough data to train the ML algorithm. The current DT approach provides a prospective ideal state of the engine, in order to carry out proactive monitoring of its health, such as an anomaly detection service. In line with tracking unmanned aerial vehicles for urban air mobility in smart city applications, this article focuses specifically on the hybrid Turbo-Shaft engine common in drones and helicopters.

However, the authors found a significant gap in synthetic data generation in the literature of the unmanned aerial vehicle domain. Therefore, moving linear regression algorithms and the Kalman filter were implemented on noisy data, which simulates the data measured from the engine in a real operational life cycle. For the thermal and hybrid models, the corresponding DT model demonstrated high efficiency in filtering out noise and a certain amount of prediction with a lower error rate in all engine parameters except engine torque.

- *DT of City Facades*

Extracting data from building facades is integral to building an information infrastructure. Compared to semantic segmentation, instance segmentation can distinguish individual facades when acquiring and analyzing information about buildings. However, collecting and labeling a large amount of real-world data for training DCNNs (deep convolutional neural networks) to perform accurate segmentation of building façade instances is a labor-intensive process.

In the work by Zhang et al. (2022), the authors developed a system that can auto-generate synthetic datasets from a CDT (city digital twin) for the segmentation of building façade instances. Digital assets of buildings in an area of Tokyo were used as an example. The system that was proposed can produce synthetic images of street views from multiple viewpoints under different atmospheric effects. The system can also generate pixel-level annotations for synthetic building facades.

- *DT of a Bridge*

In their work, Rios et al. (2023) recognize the shift to a DT paradigm for all stakeholders in the engineering, architecture and construction industry. Since this new approach is currently at an early stage of development and adoption, new proposals to optimize its application need to be explored and developed. Such attempts require the development of prototypes that need to be validated against reference data. Overall, synthetic data used to create what-if scenarios for bridge DTs can provide valuable insights into bridge performance under different conditions, allowing professionals in the respective fields to identify potential problems and make informed decisions about maintenance and design challenges.

In this work, a methodology was proposed for the creation of a synthetic data generation tool. This methodology produces high-quality FAIR (Findable, Accessible, Interoperable, and Reusable) data that allows innovatively developed prototypes to be validated and, consequently, implemented in later stages of DT creation for real infrastructure assets. The main features of the proposed methodology are: the creation of multi-metric data, namely synthetic vibration, deformation, visual and mixed data in both undamaged and damaged scenarios; environmental and operational conditions can be adjusted and included in data generation; a synchronization module that ensures that all data can be correctly tracked over time; the consideration of both epistemic and random uncertainties for the proper generation of real-world-like scenarios; and that the data generated is suitable for use in the development and validation of model-based, data-driven components informed by the physics of a digital asset.

Discussion

Analysis of artificial data generation techniques reveals significant progress in dealing with structural limitations in access to and quality of real-world data. Traditional approaches, such as ROS and SMOTE, offer simple and computationally efficient solutions to imbalances in datasets, but prove limited in their ability to generalize, especially when applied to complex and multivariate scenarios. Their variants (e.g., Borderline-SMOTE, ADASYN) seek to mitigate these problems, but remain dependent on simplifying assumptions.

On the other hand, Deep Learning-based techniques, namely Autoencoders, VAEs and GANs, represent a considerable advance in the generation of realistic synthetic data, with greater fidelity to the distribution of real data. However, these techniques require substantial volumes of initial data, greater computing power and present added difficulties in validating the data generated, especially in tabular contexts with mixed variables.

A comparative analysis of the methods reviewed reveals important trade-offs. Classical techniques such as ROS and SMOTE are simple to implement and computationally efficient, but they often fail to capture complex multivariate relationships and may distort the original data distribution. In contrast, more advanced methods, such as VAEs and GANs, offer the ability to generate highly realistic and diverse datasets, making them better suited for complex scenarios. However, these approaches require larger volumes of training data, higher computational resources, and pose additional challenges in validation. Consequently, the choice of method should balance simplicity, representativeness, and feasibility according to the application domain.

The application of these techniques to the construction of DTs has shown promise, especially in the case studies analyzed. DTs benefit from artificial data generation to simulate rare failures, speed up development cycles and ensure real-time data synchronization. Even so, practical challenges remain, particularly in calibrating hybrid models and integrating synthetic data with sensory data in real industrial environments.

It is important to highlight that the performance of artificial data generation techniques is not uniform across different application domains. In industrial production processes, where data tends to be more structured and repetitive (often with distributions close to Gaussian), oversampling methods or hybrid models prove particularly effective. In contrast, in areas such as finance or healthcare, data is more irregular, temporally variable, and less predictable, which may compromise the effectiveness of classical models

and require more robust approaches, such as GANs or privacy-preserving methods. Therefore, the practical utility of these techniques strongly depends on the nature of the domain considered, making it necessary to validate them separately in production and non-production contexts.

Conclusions

This study has shown that artificial data generation is a strategic tool for overcoming the limitations inherent in collecting and using real data, especially in critical areas such as health, manufacturing and intelligent industrial systems. Its integration into DTs systems makes it possible not only to fill historical data gaps, but also to explore future or unobservable scenarios, strengthening the predictive and adaptive capacity of these models.

The comparative analysis of the techniques revealed that, although classical approaches continue to be useful in simple contexts, it is in advanced Deep Learning techniques – particularly GANs and their variants – that the greatest potential for evolution lies, both in terms of statistical fidelity and versatility of application.

It is expected that, with the evolution of hybrid models and the increasing integration between synthetic data and real-time IoT systems, the role of these techniques will become even more central in the next generation of Digital Twins-based solutions. Future research could explore empirical validation in real industrial environments, as well as the impact of synthetic data quality on the performance of predictive models. Future research should more explicitly consider the differences between production and non-production data, exploring to what extent artificial data generation methods can (or cannot) be generalized across distinct contexts.

References

Aghazadeh Ardebili, A., Ficarella, A., Longo, A., Khalil, A., & Khalil, S. (2023). Hybrid Turbo-Shaft Engine Digital Twinning for Autonomous Aircraft via AI and Synthetic Data Generation. *Aerospace*, *10*(8), 1–17. DOI: 10.3390/aerospace10080683

Alanazi, Y., Sato, N., Ambrozewicz, P., Hiller-Blin, A., Melnitchouk, W., Battaglieri, M., Liu, T., & Li, Y. (2021). A Survey of Machine Learning-Based Physics Event Generation. *IJCAI International Joint Conference on Artificial Intelligence*, (Mc), 4286–4293. DOI: 10.24963/ijcai.2021/588

Almada-Lobo, F. (2015). The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES). *Journal of Innovation Management*, *3*(4), 16–21. DOI: 10.24840/2183-0606_003.004_0003

Aranjuelo, N., García, S., Loyo, E., Unzueta, L., & Otaegui, O. (2021). Key strategies for synthetic data generation for training intelligent systems based on people detection from omnidirectional cameras. *Computers and Electrical Engineering*, *92*(July 2020), 107105. DOI: 10.1016/j.compeleceng.2021.107105

Assefa, S.A., Dervovic, D., Mahfouz, M., Tillman, R.E., Reddy, P., & Veloso, M. (2020). Generating synthetic data in finance: Opportunities, challenges and pitfalls. *ICAIF 2020 – 1st ACM International Conference on AI in Finance*. DOI: 10.1145/3383455.3422554

Baan, J. (2021). A Comprehensive Introduction to Bayesian Deep Learning|Towards Data Science. Retrieved June 15, 2025, from DOI: https://towardsdatascience.com/a-comprehensive-introduction-to-bayesian-deep-learning-1221d9a051de/

Barth, R., IJsselmuiden, J.M.M., Hemming, J., & van Henten, E.J. (2017). Optimizing Realism of Synthetic Agricultural Images using Cycle Generative Adversarial Networks. *Proceedings of the IEEE IROS Workshop on Agricultural Robotics*, 18–22.

Batuwita, R., & Palade, V. (2010). Efficient resampling methods for training support vector machines with imbalanced datasets. *Proceedings of the International Joint Conference on Neural Networks*, 1–8. DOI: 10.1109/IJCNN.2010.5596787

Beregi, R., Pedone, G., Háy, B., & Váncza, J. (2021). Manufacturing execution system integration through the standardization of a common service model for cyber-physical production systems. *Applied Sciences (Switzerland)*, *11*(16). DOI: 10.3390/app11167581

Blagus, R., & Lusa, L. (2012). Evaluation of SMOTE for high-dimensional class-imbalanced microarray data. *Proceedings – 2012 11th International Conference on Machine Learning and Applications, ICMLA 2012*, *2*, 89–94. DOI: 10.1109/ICMLA.2012.183

Camarinha-Matos, L.M., Fornasiero, R., Ramezani, J., & Ferrada, F. (2019). Collaborative networks: A pillar of digital transformation. *Applied Sciences (Switzerland)*, *9*(24). DOI: 10.3390/app9245431

Chawla, N.V, Bowyer, K.W., Hall, L.O., & Kegelmeyer, W.P. (2002). SMOTE : Synthetic Minority Over-sampling Technique, *16*, 321–357.

Chen, J., & Little, J.J. (2019). Sports camera calibration via synthetic data. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, *2019–June*, 2497–2504. DOI: 10.1109/CVPRW.2019.00305

Chryssolouris, G. (2013). *Manufacturing systems: theory and practice*. Springer Science & Business Media.

Cochran, D.S., Kinard, D., & Bi, Z. (2016). Manufacturing System Design Meets Big Data Analytics for Continuous Improvement. *Procedia CIRP*, *50*, 647–652. DOI: 10.1016/j.procir.2016.05.004

Douzas, G., Bacao, F., & Last, F. (2018). Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences*, *465*, 1–20. DOI: 10.1016/j.ins.2018.06.056

Drummond, C., & Holte, R.C. (2003). Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats Over-Sampling. *Physical Review Letters*, *91*(3).

El Emam, K., Mosquera, L., & Hoptroff, R. (2020). *Practical synthetic data generation: balancing privacy and the broad availability of data*. O'Reilly Media.

Figueira, A., & Vaz, B. (2022). Survey on Synthetic Data Generation, Evaluation Methods and GANs. *Mathematics*, *10*(15), 1–41. DOI: 10.3390/math10152733

Foster, D. (2022). *Generative deep learning*. "O'Reilly Media, Inc.

Friederich, J., Francis, D.P., Lazarova-Molnar, S., & Mohamed, N. (2022). A framework for data-driven digital twins for smart manufacturing. *Computers in Industry*, *136*, 103586. DOI: 10.1016/j.compind.2021.103586

Gaussian mixture models. (2025). Retrieved June 15, 2025, from DOI: https://scikit-learn.org/stable/modules/mixture.html

Goodfellow, I., Bengio, Y., & Aaron, C. (2017). Deep learning. *MIT Press*, *521*(7553), 785. DOI: 10.1016/B978-0-12-391420-0.09987-X

Goodfellow, I.J., Pouget-abadie, J., Mirza, M., Xu, B., & Warde-farley, D. (2014). Generative Adversarial Nets, 1–9.

Han, H., Wang, W.Y., & Mao, B.H. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. *Lecture Notes in Computer Science*, *3644*(PART I), 878–887. DOI: 10.1007/11538059_91

He, H., Bai, Y., Garcia, E.A., & Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *Proceedings of the International Joint Conference on Neural Networks*, (3), 1322–1328. DOI: 10.1109/IJCNN.2008.4633969

Jaskó, S., Skrop, A., Holczinger, T., Chován, T., & Abonyi, J. (2020). Development of manufacturing execution systems in accordance with Industry 4.0 requirements: A review of standard- and ontology-based methodologies and tools. *Computers in Industry*, *123*. DOI: 10.1016/j.compind.2020.103300

Jo, T., & Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, *6*(1), 40–49. DOI: 10.1145/1007730.1007737

Kingma, D., & Welling, M. (2013). Auto-Encoding Variational Bayes, (Ml), 1–14.

Lan, L., You, L., Zhang, Z., Fan, Z., Zhao, W., Zeng, N., Chen, Y., & Zhou, X. (2020). Generative Adversarial Networks and Its Applications in Biomedical Informatics. *Frontiers in Public Health*, *8*(May), 1–14. DOI: 10.3389/fpubh.2020.00164

Leng, J., Wang, D., Shen, W., Li, X., Liu, Q., & Chen, X. (2021). Digital twins-based smart manufacturing system design in Industry 4.0: A review. *Journal of Manufacturing Systems*, *60*(March), 119–137. DOI: 10.1016/j.jmsy.2021.05.011

Lopes, P.V., Silveira, L., Guimaraes Aquino, R.D., Ribeiro, C.H., Skoogh, A., & Verri, F.A.N. (2024). Synthetic data generation for digital twins: enabling production systems analysis in the absence of data. *International Journal of Computer Integrated Manufacturing*, *37*(10–11), 1252–1269. DOI: 10.1080/0951192X.2024.2322981

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics* (Vol. 5, pp. 281–298). University of California press.

Mishra, M., Leturiondo-Zubizarreta, U., Salgado-Picón, Ó., & Galar-Pascual, D. (2015). Hybrid modelling for failure diagnosis and prognosis in the transport sector. Acquired data and synthetic data: [Modelización híbrida para el diagnóstico y pronóstico de fallos en el sector del transporte. Acquired data and synthetic data]. *Dyna*, *90*(2), 139–145. DOI: 10.6036/7252

Mourtzis, D. (2021). Design and operation of production networks for mass personalization in the era of cloud technology. Elsevier.

Nikolenko, S.I. (2021). Synthetic data for deep learning. *Springer Optimization and Its Applications* (Vol. 174). DOI: 10.1007/978-3-030-75178-4_1

Patki, N., Wedge, R., & Veeramachaneni, K. (2016). The synthetic data vault. *Proceedings – 3rd IEEE International Conference on Data Science and Advanced Analytics, DSAA 2016*, 399–410. DOI: 10.1109/DSAA.2016.49

Ping, H., Stoyanovich, J., & Howe, B. (2017). DataSynthesizer: Privacy-preserving synthetic datasets. *ACM International Conference Proceeding Series*, *Part F1286*. DOI: 10.1145/3085504.3091117

Pujana, A., Esteras, M., Perea, E., Maqueda, E., & Calvez, P. (2023). Hybrid-Model-Based Digital Twin of the Drivetrain of a Wind Data Generation. *Energies*, *16*.

Qian, T., Srivastava, J., Peng, Z., & Sheu, P.C.Y. (2009). Simultaneously finding fundamental articles and new topics using a community tracking method. Lecture

Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Vol. 5476 LNAI). DOI: 10.1007/978-3-642-01307-2_82

Rajabi, A., & Garibay, O.O. (2022). TabFairGAN: Fair Tabular Data Generation with Generative Adversarial Networks. *Machine Learning and Knowledge Extraction*, *4*(2), 488–501. DOI: 10.3390/make4020022

Rios, A.J., Plevris, V., & Nogal, M. (2023). Synthetic Data Generation for the Creation of Bridge Digital Twins What-If Scenarios. *COMPDYN Proceedings*, 4801–4809. DOI: 10.7712/120123.10760.21262

Rubin, D.B. (1993). Statistical Disclosure Limitation (SDL). *Encyclopedia of Database Systems*. DOI: 10.1007/978-0-387-39940-9_3686

Russell, S.J., & Norvig, P. (2016). *Artificial intelligence: a modern approach*. pearson.

Schleich, B., Anwer, N., Mathieu, L., & Wartzack, S. (2017). Shaping the digital twin for design and production engineering. *CIRP Annals – Manufacturing Technology*, *66*(1), 141–144. DOI: 10.1016/j.cirp.2017.04.040

Segovia, M., & Garcia-Alfaro, J. (2022). Design, Modeling and Implementation of Digital Twins. *Sensors*, *22*(14). DOI: 10.3390/s22145396

Shao, G. (2021). Use Case Scenarios for Digital Twin Implementation Based on ISO 23247.

Shao, G., Jain, S., Laroque, C., Lee, L.H., Lendermann, P., & Rose, O. (2019). Digital Twin for Smart Manufacturing: The Simulation Aspect. *Proceedings – Winter Simulation Conference*, *2019-Decem* (Bolton 2016), 2085–2098. DOI: 10.1109/WSC40007.2019.9004659

Tekinerdogan, B., & Verdouw, C. (2020). Systems architecture design pattern catalog for developing digital twins. *Sensors (Switzerland)*, *20*(18), 1–20. DOI: 10.3390/s20185103

van Dinter, R., Tekinerdogan, B., & Catal, C. (2022). Predictive maintenance using digital twins: A systematic literature review. *Information and Software Technology*, *151*(February), 107008. DOI: 10.1016/j.infsof.2022.107008

Wright, L., & Davidson, S. (2020). How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, *7*(1). DOI: 10.1186/s40323-020-00147-4

Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019). Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems*, *32*(NeurIPS).

Xu, L., & Veeramachaneni, K. (2018). Synthesizing Tabular Data using Generative Adversarial Networks. Retrieved from DOI: http://arxiv.org/abs/1811.11264

Zhan, G., Qingbo, Z., & Tingxin, S. (2014). Analysis and research on dynamic models of complex manufacturing network cascading failures. *Proceedings – 2014 6th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2014*, *1*(1), 388–391. DOI: 10.1109/IHMSC.2014.101

Zhang, J., Fukuda, T., & Yabuki, N. (2022). Automatic generation of synthetic datasets from a city digital twin for use in the instance segmentation of building facades. *Journal of Computational Design and Engineering*, *9*(5), 1737–1755. DOI: 10.1093/jcde/qwac086