WIND FARM FEASIBILITY ANALYSIS WITH VIRTUAL FARMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

EREN NARİN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

AUGUST 2022

Approval of the thesis:

WIND FARM FEASIBILITY ANALYSIS WITH VIRTUAL FARMS

Submitted by **EREN NARİN** in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems Department, Middle East Technical University** by,

Prof. Dr. Deniz ZEYREK BOZŞAHİN
Director of **Informatics Institute** _____

Prof. Dr. Sevgi ÖZKAN YILDIRIM
Head of Department, **Information Systems** _____

Assoc. Prof. Dr. Altan KOÇYİĞİT
Supervisor, **Information Systems** _____

Examining Committee Members:

Prof. Dr. Banu GÜNEL KILIÇ
Information Systems, METU _____

Assoc. Prof. Dr. Altan KOÇYİĞİT
Information Systems, METU _____

Assist. Prof. Dr. Serhat PEKER
Management Information Systems, İzmir Bakırçay University _____

Date: 24.08.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Eren NARİN

Signature :

ABSTRACT

WIND FARM FEASIBILITY ANALYSIS WITH VIRTUAL FARMS

NARİN, Eren
M.S., Department of Information Systems
Supervisor: Assoc. Prof. Dr. Altan KOÇYİĞİT

August 2022, 67 pages

The negative impacts of global warming are felt increasingly every day, and how humanity manages energy is one of the most potent causes of global warming. In the era of increasing utilization and importance of renewable energy resources, faster and more accurate decision-making becomes crucial against the chaotic nature of weather. At the top of the decision list, finding feasible and profitable locations to install renewable energy farms presents another challenge. However, typical renewable energy farm feasibility analysis approaches include time-consuming and costly processes. Due to the nature of wind, wind farm feasibility analysis is the most challenging problem among other renewable energy resources, and the lack of data makes this process harder. To overcome this and facilitate the wind farm feasibility analysis process, we have proposed a Generative Adversarial Network (GAN) based novel approach. We have used a limited amount of real historical data and augmented this data for new locations by using geospatial and time-series information. After that, long-term productions with a neural network-based prediction model are predicted by using this synthetic data. With this approach, we not only provide a faster solution for feasibility analysis but also propose a novel geospatial time-series data generation model. Results show that synthetic data that preserves geospatial and time-series

characteristics of the original data can be used for long-term feasibility analysis for a selected location and a specified time interval.

Keywords: renewable energy, wind farm, feasibility analysis, GAN, production prediction

ÖZ

SANAL SANTRALLER İLE RÜZGAR SANTRALİ FİZİBİLİTE ANALİZİ

NARİN, Eren
Yüksek Lisans, Bilişim Sistemleri Bölümü
Tez Yöneticisi: Doç. Dr. Altan KOÇYİĞİT

Ağustos 2022 , 67 sayfa

Küresel ısınmanın olumsuz etkileri her geçen gün kendini daha fazla hissettirmektedir. İnsanlığın enerjiyi yönetme şekli ise küresel ısımanın en güçlü sebepleri arasında yer almaktadır. Yenilenebilir enerji kaynaklarının öneminin ve kullanımının arttığı şu günlerde, hızlı ve doğru karar alabilmek büyük önem taşımaktadır. Birçok farklı, kısa veya uzun vadeli kararın yer aldığı yenilenebilir enerji yönetimi sürecinde, fizibilite analizi ve santral kurulacak yerin belirlenmesi, diğer bütün kararları etkileyen temel bir adım olma özelliği taşımaktadır. Ancak günümüzde uygulanan fizibilite analizi yöntemleri uzun süreler alan maliyetli süreçlerdir. Tahmin edilmesi zor yapısından dolayı rüzgar santrallerinin fizibilite analizi, diğer yanilenebilir enerji kaynaklarına kıyasla daha zorlu bir süreçtir. Birçok bilginin işlendiği bu süreçleri, veri eksikliği gibi problemler daha da zor hale getirmektedir. Bu problemlere çözüm üretmek ve rüzgar santrallerinin fizibilite analizi süreçlerini hızlandırmak adına bu çalışmada, Çekişmeli Üretici Ağ (GAN) temelli bir makine öğrenmesi yaklaşımı önerdik. Bu yaklaşımda, elimizde bulunan sınırlı miktarda veriyi, coğrafi komşulukları ve zaman serisi karakteristiklerini göz önünde bulundurarak çoğalttık. Ardından, oluşturduğumuz sentetik veriyi kullanarak, yapay sinir ağı mimarisinde bir tahmin modeliyle uzun süreli üre-

timleri tahmin ettik. Bu yaklaşım sayesinde hem rüzgar santrallerinin fizibilite analizi süreçlerini hızlandıracak bir alternatif ortaya koymuş olduk, hem de özgün bir sentetik coğrafi zaman serisi verisi yaratma modeli sunmuş olduk. Çalışmanın çıktıları, gerçek verinin coğrafi ve zaman serisi karakteristiklerini taşıyan bir sentetik veri setinin, santral kurulması planlanan bir konumun fizibilte analizinde kullanılabileceğini göstermektedir.

Anahtar Kelimeler: yenilenebilir enerji, rüzgar santrali, fizibilite analizi, GAN, üretim tahmini

To my big family...

ACKNOWLEDGMENTS

First, I would like to express my sincere gratitude to my supervisor Dr. Altan Koçyiğit for his guidance, patience, and encouragement. His guidance led this work to not only a dissertation but also a start-up that aims to develop innovative solutions to the renewable energy domain. After that, I would like to thank all my family and friends for their support, especially my lovely wife, Görkem, who is also a grad student during this work. And finally, I would like to thank the Turkish State Meteorological Service (MGM) and Bilgin Energy for the historical data that they provided. Their contributions carried ideas to real-world examples.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

2D	2 Dimensional
ACF	Autocorrelation Function
API	Application Programming Interface
ARIMA	Auto-Regressive Integrated and Moving Average Model
AUC	Area Under Curve
BCE	Binary Cross Entropy
BN	Batch Normalization
cGAN	Conditional Generative Adversarial Network
CNN	Convolutional Neural Network
CTGAN	Conditional Tabular Generative Adversarial Network
DCGAN	Deep Convolution Generative Adversarial Network
DeepClimGAN	Deep Climate Generative Adversarial Network
DRS	Discriminator Rejection Sampling
EPİAŞ / EXIST	Enerji Piyasaları İşletme Anonim Şirketi / Energy Exchange Istanbul
ESM	Earth System Model
GAN	Generative Adversarial Network
GMM	Gaussian Mixture Model
GP	Gradient Penalty
JS	Jensen-Shannon
KL	Kullback-Leibler
KNN	K-Nearest Neighbors Algorithm
LSTM	Long-Short-Term Memory
LSGAN	Least Squares Generative Adversarial Network

MAE	Mean Absolute Error
MCMC	Markov Chain Monte Carlo
MGM	Turkish State Meteorological Service
MHGAN	Metropolis-Hastings Generative Adversarial Network
MIE	Moran's I Metric
MLP	Multi-Layer Perceptron
MSE	Mean Squared Error
NN	Nearest Neighbor
NWP	Numerical Weather Prediction
PACF	Partial Autocorrelation Function
PCA	Principal Component Analysis
ReLU	Rectified Linear Units
RGAN	Recurrent Generative Adversarial Network
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
SpaceGAN	Space Generative Adversarial Network
t-SNE	T-Distributed Stochastic Neighbor Embedding
TGAN	Tabular Generative Adversarial Network
TGAN-skip-Improved-WGAN-GP	Tabular Generative Adversarial Network with skip connections and improved Wasserstein Generative Adversarial Network with gradient penalty
TimeGAN	Time-series Generative Adversarial Network
TRTR	Train on real, test on real
TRTS	Train on real, test on synthetic
TSTR	Train on synthetic, test on real
TSTS	Train on synthetic, test on synthetic
UTM	Universal Transverse Mercator
WassersteinGAN / WGAN	Wasserstein Generative Adversarial Network

WGS	World Geodetic System

CHAPTER 1

INTRODUCTION

Renewable energy investments are increasing day by day; thus, making fast and accurate decisions is becoming more critical. These decisions may be short-term decisions like market clearing, medium-term decisions like production planning, or long-term decisions like feasibility analysis. Feasibility analysis of potential renewable energy farms is a crucial step for efficient and profitable renewable energy investments. This operation is widely conducted with physical methods nowadays. However, the downscaling of numerical weather prediction (NWP) data —which necessitates a description of the area, including roughness and obstacles— is frequently used in these physical approaches, which are time-consuming and expensive operations [1]. Among renewable energy resources, the wind is the hardest to predict. Thus, the feasibility analysis of wind farms is more costly than other renewable energy resources. Due to the stochastic nature of wind, historical data is of great importance. Still, the lack of data is another main reason for the high cost of the feasibility study.

The main purposes of a wind farm feasibility study are listed below:

- Determining the most appropriate location for the farm.
- Determining the most appropriate scale and turbine type for the farm.
- Examining wind potential of the location.
- Determining physical and planning constraints.
- Determining project costs, payment, and return on investment.
- Risk assessment.

Among these purposes, determining the most appropriate location/scale and examining the wind potential of the location are data-driven tasks. These tasks require a vast amount of data that usually is not available. Surface shape, altitude changes, and wind map are a few data categories that analysts should process. Common approaches for these tasks generally consist of collecting data on-site and using interpolation-based wind maps. However, collecting data is time-consuming, and there is significant uncertainty in wind speeds on wind maps. Moreover, even if all the data required is available and accurate, processing this amount of input is a complex task. Machine learning offers promising solutions to these two problems; lack of data and solving complex problems. In this work, we applied machine learning solutions to both issues.

Renewable energy is one of the areas where the lack of data is a common problem. There are two main reasons for this: First, renewable energy data is categorized as strategically important. Therefore it could be hard to obtain this data for 3rd party analysts or application developers. The second reason is that renewable energy production is strongly related to weather conditions. Historical weather data and weather forecasts are required for working on problems such as renewable energy efficiency or future forecasts. However, weather data can be expensive to obtain or may not even be available for required locations. Collecting this data will not be a good option if years of weather data are needed to solve a problem. At this point, augmenting weather data with a generative neural network may be a good solution.

The synthetic data generation approach is of widespread interest for domains such as image generation. However, synthetic weather data generation is a different problem than image generation. Despite weather data having spatial correlations like images, weather data also has temporal characteristics that should be preserved in the synthetic data. This problem makes generating realistic weather data more difficult. On the other hand, different studies in the literature handle geospatial data generation and time-series data generation problems separately. In this work, we examined related studies and combined the most feasible approaches for geospatial and time-series data generation problems with novel additions. Finally, we applied our proposed method to a significant business case that suffers from a lack of data/inefficient and costly processes.

Wind farms are usually located at hard-to-reach locations where no weather station is available. Thus, our research aims to propose a solution that is not only to recover partial data but also to missing data that correlates with available historical data from different locations. Our approach can be applied independently of historical data availability. Of course, more historical data will lead to better results. Our proposed method makes the feasibility analysis process faster and cheaper. In the days when renewable energy investments are increasing dramatically, investors or governments can quickly analyze wind power capacities of different locations using this method. Also, various wind power capacities for a site can be tested in terms of feasibility.

There are three main research questions in this dissertation:

- **RQ1:** If we have historical geospatial weather data from different locations, can realistic synthetic weather data be generated for a site where no past data is available?
- **RQ2:** Can the generated synthetic data simulate the time-series characteristics of weather data?
- **RQ3:** Can the generated synthetic data be used for long-term feasibility analysis of potential wind farms?

The remainder of this dissertation is organized as follows: Chapter 2 presents an overview of related studies in the literature, Chapter 3 presents the proposed method, Chapter 4 presents experiments and evaluation results with discussions, and Chapter 5 presents the conclusions and directions for future work.

CHAPTER 2

RELATED WORK

Throughout human history, humans have tried to understand and predict the weather. Many different prediction techniques were applied, and humans have made important decisions according to these predictions. Nowadays, statistical approaches are the most popular techniques for weather prediction. Machine learning and deep learning models are becoming strong alternatives to physical weather models, and the usage of neural networks in this domain is increasing day by day. However, lacking data is a crucial problem, as these techniques require much data. Hence, academic research about synthetic data generation in the energy and weather domains has recently increased. Most of these researches also inspired this dissertation.

In this work, we combined some of the effective techniques in literature and proposed improvements according to our business case. First, to generate synthetic geospatial time-series data, we used neighborhood information like SpaceGAN [2], which is comprehensively explained in the following sections. However, while SpaceGAN uses only location information for neighborhood analysis, we added time criteria to neighbor selection to preserve the time-series characteristics of the original data. This novel approach came with significant problems. We applied both empirical and statistical normalization techniques to use location in meters and time in hours as different dimensions of the same space. Once dimensions are normalized, we can select neighbors according to the original data point in terms of location and time. We used K-Nearest Neighbor (KNN) algorithm to analyze neighborhood structure and expanded the input data set with neighborhood information. After these preprocessing operations, we built a cGAN architecture to use this neighborhood information as conditions of generated synthetic data. In GAN architecture, we utilized network layers with continuous loss function Mean Square Error (MSE) and classification loss function Binary Cross-Entropy (BCE). This way, generated data via the generator of the GAN is penalized symmetrically. We generated synthetic wind speed data for the target location with this GAN model. After that, we used an RNN model for wind energy production prediction. Thus, the time-dependent correlation between wind speed and wind power generation can be preserved while the temporal dynamics in wind power generation are also preserved. We evaluated synthetic data at the end of these operations in statistical and exploratory ways. The proposed wind farm feasibility analysis is completed after the abovementioned steps.

2.1 GAN Approaches

Since Goodfellow $et\ al.$ first introduced the Generative Adversarial Network (GAN) [3], different researchers have strengthened the GAN methodology in different ways and introduced alternative applications of GAN. Conditional Generative Adversarial Networks (cGAN), which Mirza and Osindero proposed [4], is one of the most popular of these applications. Mirza and Asindero presented this novel way for synthetic data generation to resolve ambiguous data label problem. They conditioned both the generator and discriminator on some additional information y. At every learning iteration, the inputs and outputs of the model are checked on this condition, and the generator and discriminator are optimized. This way, the generator's possible sample space is narrowed down, and the generator becomes more direct to the target space. This work has been referenced by many other studies and is still widely used.

In another work [5], GAN's unstable training process is tried to be improved. The proposed method, Deep Convolution GAN (DCGAN), is built with convolutional and convolutional-transpose layers with Adam optimizer and Rectified Linear Units (ReLU) and Leaky ReLU activation functions. To resolve the gradient problems, Batch Normalization (BN) algorithm was used in both the generator and the discriminator. The BN algorithm limits the sample size that the generator collects. This makes the generator more likely to collapse [6] despite DCGAN outperforming the GAN application on different test cases.

Mode collapse is a common problem for GAN applications. This problem occurs when the generator is stuck to a single or small set of outputs. If the generator finds an easy way to trap the discriminator, it will adhere to this way and will fool the discriminator over and over again. Another common problem of the GAN architecture is the vanishing gradient problem. This problem occurs when the discriminator is locally saturated and rejects all generator outputs. Hence generator can not learn the distribution of input. To overcome these issues, Arjovski $et\ al.$ proposed WassersteinGAN (WGAN) [7]. WGAN uses Wasserstein distance as a loss function, while original GAN uses Jensen-Shannon (JS) divergence [3]. The main difference between Wasserstein distance and JS distance is that Wasserstein distance is continuous and almost differentiable everywhere. In this way, the GAN training process becomes more stable. This method offers a solution to both mode collapse and vanishing gradient problems.

Another alternative solution for the vanishing gradients problem was proposed by Mao $et\ al.$ [8], which is Least Squares GAN (LSGAN). Since regular GAN discriminator models use sigmoid cross entropy as a loss function, samples far away from the decision boundary on the correct side are marked as no loss. This may cause misguidance in the discriminator. Mao $et\ al.$ proposed an alternative loss function for discriminator: least squares. This symmetric loss function penalties outliers equally regardless of which side of the decision boundary. In this way, the generator outputs approach the decision boundary, and samples become more similar to real data. Moreover, this method resolves the vanishing gradient problem while stabilizing the training process. Results show that LSGAN can generate higher quality images than regular GAN.

Preserving the distribution of input data is an important task of GANs. In their work [9], Azadi $et\ al.$ used the rejection sampling method to improve the generator's sample quality. For this, they used the discriminator's learned information for training the generator. Rejection sampling is based on generating samples from a proposal distribution which is easier to generate samples from, and accepting or rejecting these samples according to their acceptance probability. In mentioned work, the authors applied this method to the generator according to the discriminator's learned information and named this method Discriminator Rejection Sampling (DRS). Thus, generator samples eventually become closest to input distribution. Under strict assumptions, the proposed method can recover the input data distribution. However, due to the nature of the rejection sampling algorithm, if input data has sharp peaks or downs, most of the outputs will be rejected, making the training process more expensive.

In the rejection sampling algorithm, samples are independent of each other. Instead of this, in the Markov chain Monte Carlo (MCMC) method [10], all samples are relevant to the previous sample, and it is decided what the next sample will be according to the previous sample's acceptance. Thus, this method is more suitable for high-dimensional data. In their work [11], Turner $et\ al.$ developed the Azadi $et\ al.$'s idea by using MCMC instead of the rejection sampling algorithm. In this way, generated samples are related to the previous sample's acceptance probability. The first samples will be burn-in samples, but eventually, the generator will learn the exact distribution of input data. This method makes generating samples more expensive, but sample quality will increase. The results of this work also prove this proposition.

Learning the distributions becomes harder when input data have different data types, such as numerical, categorical, text, date, etc. So many different business domains, such as medical, education, or energy, contain multi-type tabular data. And in most cases, these columns are related to each other. Xu and Veeramachaneni suggested a widely used model for synthesizing interrelated multi-type tabular data, Tabular GAN (TGAN) [12]. In this model, they firstly normalized the variables with reversible data transformation techniques. They used Gaussian Mixture Model (GMM) for normalizing numerical data and used one-hot-encoding for categorical values to make them differentiable. They built the generator with the Long-Short-Term Memory (LSTM) network due to its ability to capture temporal correlations of recurrent networks and built the discriminator with Multi-Layer Perceptron (MLP). Also, Kullback–Leibler (KL) divergence was used as the loss function to make the model more stable. They tested both machine learning efficacy and preserving correlation capability of TGAN and compared the model with several different data synthesizing techniques in terms of these tests' scores. Results show that TGAN overperforms other synthesizing methods in most cases. In another work, Park $et\ al.$ suggested TableGAN with similar purposes [13]. The main difference between TableGAN and TGAN is that TableGAN uses convolutional layers instead of recurrent layers in the generator. Moreover, TableGAN uses cross-entropy loss while TGAN uses KL divergence. TableGAN, too, shows successful performance in generating synthetic tabular data generation. On the other hand, TGAN is a bit more popular than TableGAN in the development community [14][15].

The generative models should also preserve temporal correlations across time to generate time-series data. To meet this requirement, Yoon $et\ al.$ proposed an alternative model for synthesizing time-series data [16]. They focused on combining the con-

trollability of supervised learning with the flexibility of unsupervised learning. Four different network structures that play distinct roles in modeling the data were used: Autoencoders; embedding and recovery networks and adversarial networks; sequence generator, and sequence discriminator. Autoencoders' main task is providing a mapping between feature and latent space while decreasing dimensionality. This structure allows adversarial components of the model to learn temporal dynamics. Adversarial networks also work in embedding space. The proposed model is compared with different network structures in different scenarios. T-Distributed Stochastic Neighbor Embedding (t-SNE) and Principal Component Analysis (PCA) were used for visual comparison. Also, models are compared in terms of their discriminative and predictive scores. According to the results, TimeGAN overperforms other state-of-the-art benchmarks in generating realistic time-series data.

2.2 GAN Applications

Since the introduction of GAN, GANs have been applied to different scenarios in different domains. Synthetic image generation is the most popular scenario where GANs have been used. For example, the Facebook AI Research Team used GANs for domain-to-domain knowledge transformation [17]. They trained their model at the input domain with labeled image data and generated sample images for a new domain. A filter function f is employed for transforming training data to the expected data format before the generator is trained. They tested their model in different scenarios, transforming real people images into people emojis or number photos on boards or papers to hand-written numbers. Outputs of this work show GAN methodology may be used for knowledge transfer between cross domains. Like this work, Dosovitskiy and Brox employed GANs for handling feature distances instead of Euclidean distances between images [18]. Changing loss parameters induced better image visuals, but the variable selection and optimization remained as future work.

The lack of data is a crucial problem in the energy domain. A comprehensive energy prediction or analysis solution requires data from different sources such as energy production & consumption data, breakdown and maintenance data, weather data, etc. These data usually are hard to obtain or maybe even not available. With this limited data, the accuracy of the developed prediction models, especially in the renewable energy domain, is insufficient. Hence, training the prediction models with a combination of real and synthetic data seems promising. Based on this, Fekri $et\ al.$ conducted a work [19]. In their work, they generated synthetic energy consumption data for gathering efficient smart grid planning and operation insights. Due to Recurrent Neural Networks (RNN) ability to capture temporal dependencies, Convolutional Neural Networks (CNN) in GAN structure are replaced with RNN, and this method was named Recurrent GAN (RGAN). To improve training stability and increase the quality of generated data, WGAN and MHGAN approaches were also applied. Before the training process, all features are normalized with ARIMA and Fourier Transform between 0-1. To protect the temporal characteristics of the data, input data was given with the sliding window method to the GAN generator, not random input data. Although GAN-generated images can be visually evaluated, evaluation of GAN-generated time series data is still challenging. In this work, the authors

used train on real, test on synthetic (TRTS), train on synthetic, test on real (TSTR), train on real, test on real (TRTR), and train on synthetic, test on synthetic (TSTS) techniques for evaluation. These techniques are based on a simple prediction model which is trained with different input data and is tested on its prediction accuracy. TRTR case evaluates the quality of the forecasting model itself, and other cases' outputs are compared with TRTR case's outputs. Results show that the model trained with synthetic data achieves similar accuracy as the one trained with real data. With the addition of features generated by ARIMA and Fourier transform, the proposed model can generate higher quality data than regular GAN, and the accuracy of the prediction model trained with generated data increases.

Another similar work was conducted by Hazra $et\ al.$ [20]. In this work, to reduce the imbalance cost of the energy market, energy demand forecast accuracy is tried to improve using synthetic data. The proposed model (TGAN-skip-Improved-WGAN-GP) is based on Tabular GAN (TGAN) with skip connections (TGAN-skip) and improved Wasserstein GAN (WGAN) with gradient penalty (Improved-WGAN-GP). Combining these methods solves the mode collapse problem while reducing convergence time. The proposed model is trained with preprocessed tabular energy consumption data and synthetic consumption data generated for different periods. The results of different applications show that a prediction model trained with real and synthetic data has higher prediction accuracy than a prediction model trained with only real data. Therefore, this work proposes a promising solution to the lack of input time series data in the energy domain.

Moon $et\ al$, handled similar problem with different methods [21]. Like previous examples, they separated external feature data generation and energy load data generation processes due to GAN's sometimes failing to learn the correlation between input (external features such as time and weather conditions) and output data (energy load). Conditional Tabular GAN has been used for generating synthetic external feature data. After that, a deep learning-based regression model trained with real data is used for energy load data generation. In this way, the correlation between features and energy load is preserved. Finally, a forecast model is trained with a combination of real and fake data. Evaluation scores have shown that synthetic data usage has increased the energy load forecast accuracy.

GANs have also been used for geospatial data generation recently. Since energy data is strongly related to weather data, focusing on preserving the geospatial characteristics of GAN-generated data would be beneficial. The Alan Turing Institute researchers Klemmer $et\ al.$ suggested a novel approach to geospatial data generation problem [2]. In this work, the authors introduced SpaceGAN, which can learn spatial correlations with a conditional GAN. Authors used neighbor features as conditions of GAN. At first, they vectorized the neighborhood dimensions. Afterward, neighbor points were selected via the K-Nearest Neighbors (KNN) algorithm. The method was evaluated in two different cases. The first case is regularly distributed color points on 2D space, and the second case is house prices in California. Moran's I Metric (MIE) and Root Mean Squared Error (RMSE) were used for discriminator evaluation criteria in both cases. Either way, the suggested method successfully generated synthetic data and increased the prediction model performance by augmenting training data with generated data. Since MIE aims to calculate spatial correlations, usage of MIE gave better results than the usage of RMSE. This work is an excellent example

of geospatial conditioned GAN usage in 2D space.

Geospatial data are widely used for Earth System Models (ESMs) too. These models are for simulating environmental parameters through time. Pucko $et\,al.$, suggested an alternative solution for ESM analysis [22]. In this work, cGAN was used as a replacement for ESMs which are complex and expensive to run. Two different conditional parameters were defined: monthly average temperature and precipitation values (c_1) and recent K days averages of all climate variables (c_2). c_1 variable gives information about the type of scenario, such as high or low warming, while the c_2 variable gives information about recent trends and continuity. Results show that GAN-generated data have similar characteristics to real climate data and may be used for increasing ESM simulations performance while dramatically lowering the working time of these simulations. In another work, based on DeepClimGAN, Ayala $et\,al.$ separated models according to season [23]. Loosely conditioned models for summer-spring and winter-fall times were used for data generation, and evaluation scores show that model performance and accuracy increased compared to DeepClimGAN.

As another example of geospatial data generation with GAN, Wu and Bilijecki used conditional GAN for creating building footprints and city maps from land-use data and road networks [24]. This way, they can translate geographical data with less information and higher performance. They trained the model with sliced map images, and at the end of the process, they combined output slices into a single detailed map. Output images are capable of revealing urban form. This faster map data transformation method can be used for different use cases such as population estimation, urban morphology, energy, climate simulations, etc.

2.3 Wind Energy Prediction Approaches

As mentioned before, wind speed or wind power forecasting is not a new interest for the academy. Like Shahram Hanifi $et\,al.$ systematically examined, there are many different wind power forecast studies and approaches [1]. Shahram Hanifi $et\,al.$ classified these studies according to prediction horizons and prediction methodologies. Prediction horizons for wind power forecasting are grouped under four classes; very short-term, short-term, medium-term, and long-term predictions, which are distributed from 30 minutes to months. Also, prediction methodologies are grouped under three classes; persistence methods, physical methods, and statistical methods. Analysis results show that physical methods are more suitable for longer-term predictions but require more computation time. On the other hand, statistical approaches are faster, cheaper, and more successful for shorter-term predictions. This review also includes some improvement suggestions for more accurate predictions.

CHAPTER 3

METHODOLOGY

Feasibility analysis means evaluating a project regarding applicability, practicality, profitability, and compliance with regulations. In this work, we focus on the practicality and profitability analysis of a wind farm that has not yet been built. Applicability and compliance with regulations are out of this work's scope. This work aims to determine a candidate wind farm's possible energy production for the past few years if this farm was built in a specific location. With this information, investors can decide on a wind farm's most suitable sites and energy generation capacity. In this way, they can avoid insufficient or unnecessary investments.

There are two main parts of the practicality and profitability analysis of wind farms: Determining the wind potential of a selected location and determining possible productions of a candidate wind farm with a specific set of turbines. These tasks are consecutive processes and are conducted with different approaches. Generally, wind maps are used to determine a location's wind potential. Wind maps or wind atlases usually consist of 10 minutes to 1 hour of average wind measurements from 10 meters to 100 meters for 10 to 30 years. These data may be collected by weather stations, satellites, or other weather instruments. Since obtaining such data for a multitude of potential wind farm locations is almost impossible, interpolation techniques are widely used on these maps. However, due to the chaotic nature of wind, interpolating wind speed data causes large uncertainty and errors on wind maps.

Edward Lorenz's definition of chaos is: "When the present determines the future, but the approximate present does not approximately determine the future." [25]. Lorenz also gives weather and climate as natural examples of chaotic systems. Lorenz's statement also claims the idea that chaotic behaviors are predictable in the short term but unpredictable in the longer term. Also, Tian validates this statement for wind in his work [26]. Formulating these chaotic behaviors is challenging, and interpolation techniques may be insufficient to solve this problem. However, machine learning and deep learning approaches propose an alternative way to simulate these behaviors. Due to the GAN's ability to preserve spatial correlations, we used this neural network approach instead of interpolation techniques.

To select features of the GAN model, we followed the "First Law of Geography", made famous by W. R. Tobler, who states that "everything is related to everything else, but near things are more related than distant things" [27]. Since wind data is geospatial, neighborhood information is crucial in determining the current wind situation and future behaviors. We used the closest known data points to predict an unknown location's wind speed. First, we selected the closest points to the target point.

Second, we added the closest points' information to the target point as features. After that, we generated synthetic data with GAN using these features as conditions of generated data. We used this synthetic data as historical wind speed data of the target location. In other words, we used this data as a wind map.

We used the KNN algorithm in this work for neighbor data points selection. KNN algorithm selects neighbors according to a distance metric such as Euclidian distance calculated in terms of the distances in N-dimensional feature space. For a regular geospatial neighborhood selection, dimensions are latitude and longitude, or for a Cartesian coordinate system, X and Y values. However, since we aim to preserve time-series characteristics of the data besides geospatial correlations, we used four dimensions for selecting neighbors: Latitude, longitude, altitude, and time (Figure 3.1). The meter unit can define latitude, longitude, and altitude. On the other hand, time can't be defined in terms of meters. Moreover, these dimensions have different scales and variances. These make applying normalization to all dimensions mandatory. Hence, we used specific normalization techniques to combine latitude, longitude, altitude, and time dimensions on the same scale. We selected the normalization coefficients of time dimensions with Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) methods. By considering the domain characteristics, we applied a different technique to determine the normalization coefficients of geospatial distances: latitude, longitude, and altitude. We analyzed the distributions of these parameters in our dataset, and we used standard deviation values of parameters as their normalization coefficient. Thus, we have been capable of manipulating time, altitude, and coordinate dimensions in the same space for neighbor point selection.

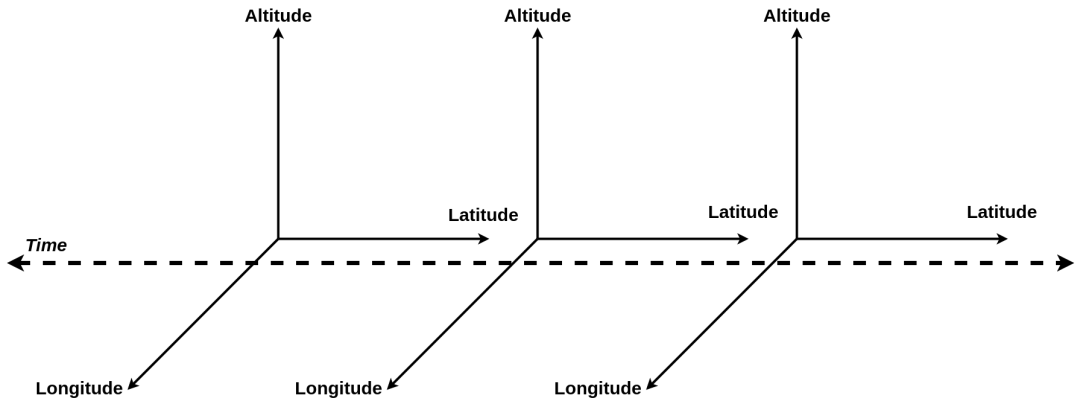
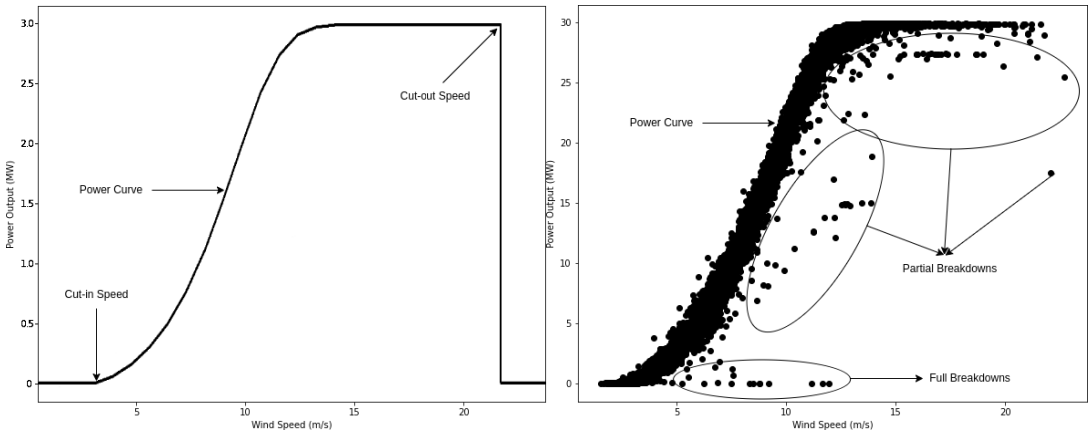


Figure 3.1: Dimensions that are used for neighbor point selection with KNN

Another major part of the feasibility analysis of wind farms is determining the possible production of a candidate wind farm with a specific set of turbines. This task is mostly applied by using power curves of different turbine types. However, there are some issues with this method [28][29]. First, the power curves and the behaviors, such as cut-in and cut-out speeds of turbines, may change according to their manufacturers. Cut-in speed is the minimum wind speed limit that a wind turbine is able to generate power. On the other hand, cut-off speed is the limit where the wind turbine shuts down itself to prevent any damage. The second issue is power curves are derived under ideal conditions. The outputs of a turbine may differ by location, altitude, and time. Another issue about power curve modeling is power curves give

the possible output of a single turbine. Power curves may give misleading results when a group of turbines' productions is wanted to predict. The effects of these issues on wind speed - production curves can be observed in the Figure 3.2. Also, partial or complete breakdowns and maintenance operations can also be observed in this figure. All of these issues make the production analysis of a wind farm a complex task. Thus, instead of a non-linear power curve, we used a neural network-based prediction model to predict possible past productions of candidate wind farms. Due to the generative models' poor ability to preserve complex correlations, we worked on generating synthetic wind speed data and predicted wind power data with a different neural network-based time-series prediction model using generated wind speed data. This approach also adds extra flexibility to the proposed approach. With this method, different scales and turbine types can be tested rapidly. We trained our prediction model with a known wind farm's past production and wind speed data. This way, the model can learn different scenarios like breakdowns or geographic conditions. We selected this reference wind farm according to its similarity with candidate wind farms regarding production capacity, turbine types and count, and location. This pre-trained prediction model is our virtual farm.



(a) An example power curve of a typical wind turbine

(b) The power curve of a real wind farm with hourly averages

Figure 3.2: Power curves of a single turbine and a wind farm

We propose that using synthetic wind data and virtual wind farms for these analyses will make the feasibility analysis of wind farms more applicable and cheaper. In this way, investors and governments may make quick decisions; thus, wind energy investments may speed up. However, generating synthetic wind data is not a trivial task. Generated data should contain local information like maritime or landform and should preserve time-series characteristics of wind data. We investigated different researches in the Related Work chapter (Chapter 2) that handle these two concerns separately. Our aim is to propose a novel integrated solution to generate synthetic geospatial time-series data and use this data for the feasibility analysis of a wind farm with a virtual wind farm.

The flow of the proposed wind farm feasibility analysis methodology is delineated in the Figure 3.3. All steps are labeled, and details of related steps are referenced in the rest of the Methodology chapter with these labels. The process inputs are historical

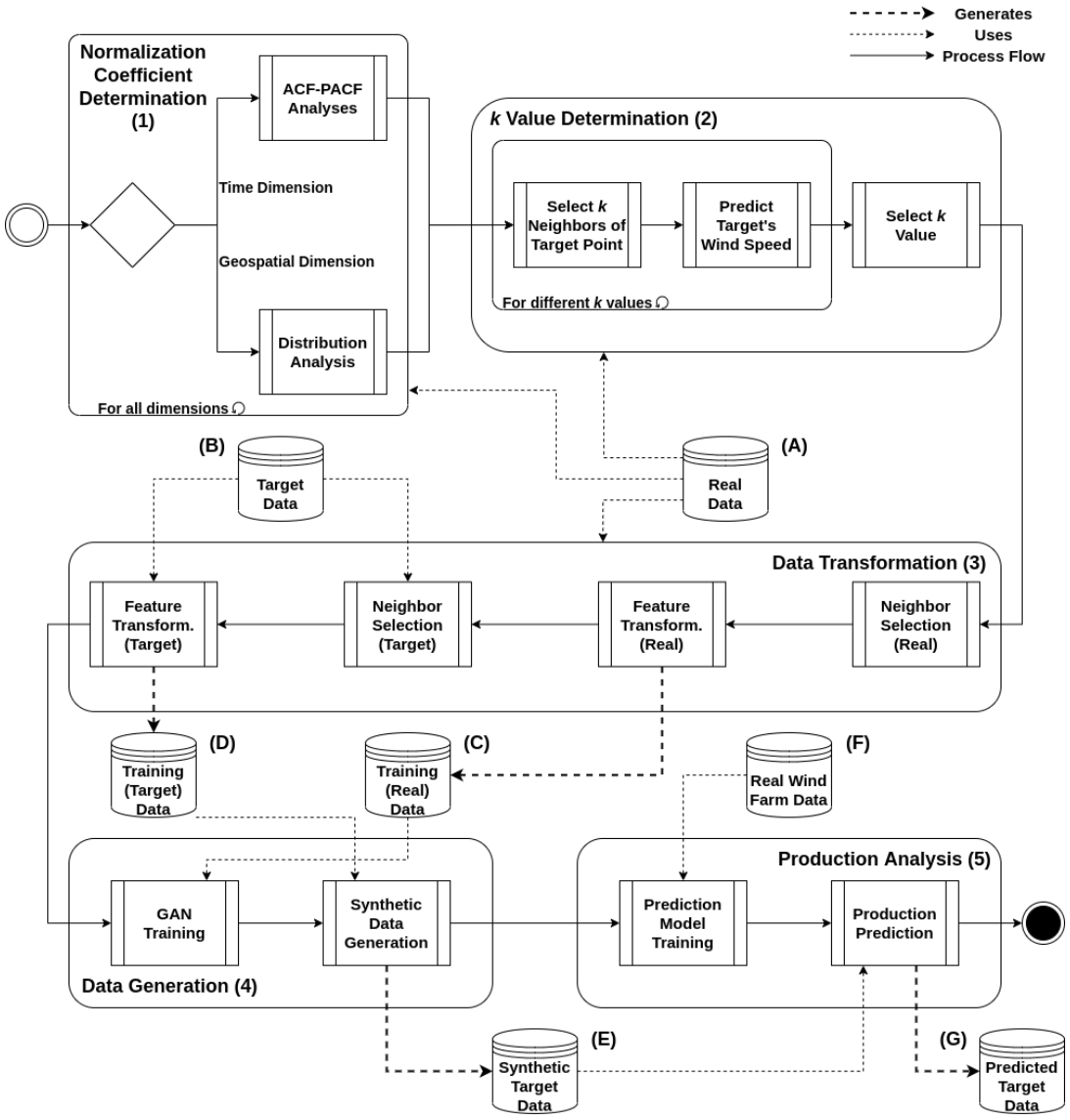


Figure 3.3: The flow of feasibility analysis process of a candidate wind farm with a virtual wind farm

weather data from different weather stations, the candidate wind farm's target location, and the feasibility study's time interval. Target location and time interval are combined under "Target Data" (Figure 3.3 - B), while "Real Data" (Figure 3.3 - A) composes historical weather data. The process output is the predicted production of the candidate wind farm for the target location and time interval (Figure 3.3 - G).

3.1 Neighborhood Analysis

To analyze neighborhood relations, a neighborhood function should be selected first. However, since our neighborhood dimensions are in different scales and units, we normalized all dimensions before neighborhood selection (Figure 3.3 - 1). Other-

wise, dimensions with lower scales will dominate the neighborhood selection, and dimensions with higher scales will have nearly no effect on the neighborhood selection process. We determined independent normalization coefficients for all dimensions to get all dimensions on the same scale. According to the normalization coefficients that were determined, we updated the distance function of the neighborhood analysis method and determined neighborhood-related parameters of the distance function (Figure 3.3 - 2). After these steps, we can select the neighbor points from the most correlated points to the target.

3.1.1 Distance Function

KNN is a non-parametric supervised learning method used for classification and regression [30]. Our motivation for using KNN is to preserve local information in terms of both geography and time. For example, for weather data on an hourly basis, the next hour's and the previous hour's data are the closest neighbors of the current hour, and the consecutive hours' weather parameters are strongly correlated to each other. Also, the nearest locations' weather conditions give potent insights into the current location's weather conditions. Based on this, we combined geospatial and time distances and selected the neighbor points according to this combined distance (Figure 3.1).

We used a simple regression-based prediction model to determine the k values of different cases (Figure 3.3 - 2). We trained the prediction model for different locations with different k neighbor points' wind speed values and predicted the target location's wind speed value. According to the accuracies of models that were trained with k neighbors' information, we determined an optimum k value. k value was determined for all experiment cases independently. The details of this operation are presented in the Experimental Work chapter (Chapter 4).

We used Euclidean distance as the distance function of KNN. Euclidean distance is equal to the square root of the sum of squares of distances in all dimensions, as shown in the Equation 3.1. However, in our case, dimensions have different units and scales and are non-convertible to each other. This creates a need for the normalization of all dimensions. Hence, before selecting neighbor points with KNN, we normalized all dimensions with normalization coefficients determined by dimension type-dependent techniques. Thus, we can convert time and geospatial distances to each other and use Euclidean distance for the nearest points selection.

$$\begin{aligned} d(p,q) = d(q,p) &= \sqrt{(p_1-q_1)^2 + (p_2-q_2)^2 + ... + (p_n-q_n)^2} \\ &= \sqrt{\sum_{i=1}^{n}(p_i-q_i)^2} \end{aligned} \tag{3.1}$$

In the Equation 3.1, n represents the dimension count, p represents the reference point's value in a given dimension, and q represents the target point's value in a given dimension.

With normalization, Euclidean distance formulation becomes like in the Equation 3.2. With this formulation, all dimensions are brought together on the same scale, and the distance can be defined in terms of correlation.

$$d(p,q) = d(q,p) = \sqrt{(\frac{p_1 - q_1}{NC_1})^2 + (\frac{p_2 - q_2}{NC_2})^2 + ... + (\frac{p_n - q_n}{NC_n})^2} = \sqrt{\sum_{i=1}^{n} (\frac{p_i - q_i}{NC_i})^2} \tag{3.2}$$

In the Equation 3.2, n represents the dimension count, p represents the reference point's value in a given dimension, q represents the target point's value in a given dimension, and NC represents the normalization coefficient of the related dimension.

3.1.2 Distance Normalization

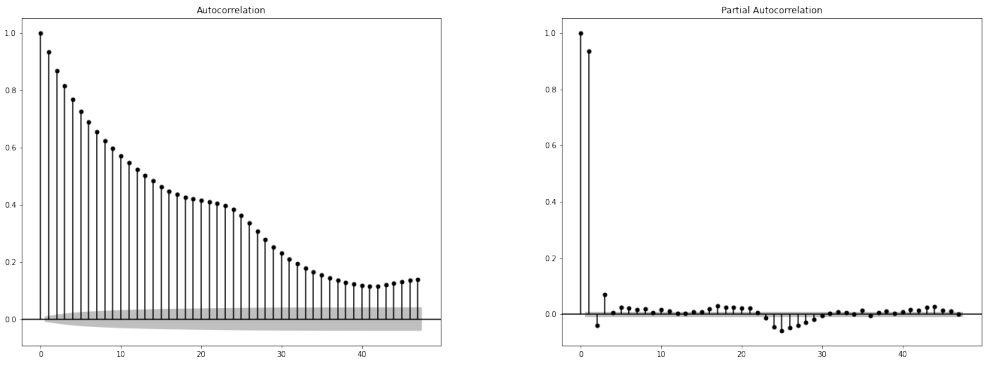
To use time and geospatial distance dimensions in the same space, we normalized these distance features according to their unit effect on wind speed. We aim to determine the unit value in all dimensions, which change the wind speed equally. Hence, we handled the normalization of time and geospatial distance features separately (Figure 3.3 - 1).

3.1.2.1 Time Dimensions Normalization

Wind behaviors are dependent on seasonal effects. These effects mainly occur at yearly intervals. Thus, we used the day of year variable as the first dimension of the time. In this way, we can simulate yearly seasonality in our model. Moreover, our data is on an hourly basis. That means the closest neighbors of a data point are previous and next hours' data points in terms of time. Therefore, we used the hour variable as the second dimension of the time. In this way, we can simulate daily seasonality too in our model.

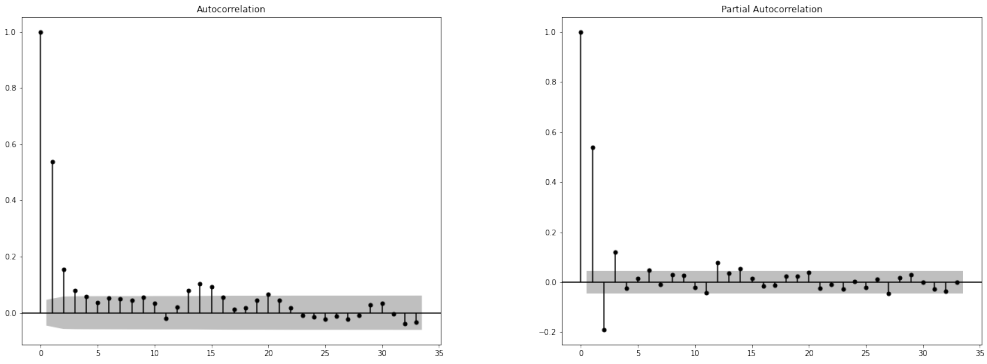
While determining the time dimensions' normalization coefficients, we examined ACF and PACF analyses' outputs. Since the target point's historical data isn't available, we used the closest weather station's data to the target point during this process. ACF and PACF analyses give the correlation of a series with its lagged values. This way, lagged values that determine the current value could be examined. The difference between ACF and PACF is that PACF gives a direct correlation between the current value and lagged value, while ACF gives a combination of direct and indirect correlations between the current value and lagged values. To analyze correlations in a series, both of these methods are crucial. Since we aim to select the points that show the strongest correlations as neighbors, we used ACF-PACF analyses for normalization coefficient determination of time dimensions. This way, points with a strong correlation with the target point are selected as neighbors, while points with weaker correlations with the target point are marked as farther points.

As an example, weather station at Urla/İzmir data was processed (Figures 3.4, 3.5). According to ACF-PACF analyses on an hourly basis (Figure 3.4), it can be observed that wind speed has shown a strong correlation within 3 hours ahead or behind. Hence, the normalization coefficient of the hour dimension can be determined as 3. On the other hand, according to ACF-PACF analyses on a daily basis (Figure 3.5), only 2 days difference in wind speed has an acceptable correlation. Before this analysis, hourly-based data have downsampled to a daily basis with mean values. We assumed that 3 hours difference in wind speed data has a nearly equal effect on wind speed with 2 days difference in wind speed data in this example. Therefore, we become capable of transforming hour and day differences into each other.



(a) ACF analysis (b) PACF analysis

Figure 3.4: Urla/İzmir weather station's wind speed ACF (a) - PACF (b) analyses on an hourly basis



(a) ACF analysis (b) PACF analysis

Figure 3.5: Urla/İzmir weather station's wind speed ACF (a) - PACF (b) analyses on a daily basis

3.1.2.2 Geospatial Distance Dimensions Normalization

As a geospatial location definition notation, World Geodetic System (WGS) 1984 version, also mentioned as the WGS84 version, was used. Since Universal Transverse Mercator (UTM) coordinate system's easting and northing values are easier to process for distance in meters calculation, we used this representation of coordinates instead of the common latitude-longitude representation. In this way, the exact position of the target location can be defined in meters. Also, altitude values in our dataset were defined in meters too.

To determine normalization coefficients of geospatial distance dimensions, we used the standard deviation of these values in our dataset. Standard deviation (σ) is a metric that shows the variation of the data according to its mean value (Equation 3.3). A higher σ value indicates that the values tend to be close to the mean value of the data, and a lower σ value indicates that the data is more spread. Standard deviation is already used in various domains such as academia, finance, medicine, forecasting, etc. The main idea behind the standard deviation usage is that the σ value indicates the value's possible single-step change size in the data. It also indicates the confidence level of estimation according to past data distribution. With the inspiration of these approaches, we used the σ values as the geospatial dimensions' normalization coefficient. We assumed that data points closer than the σ value have the highest correlation with the target point while farther points' correlations than the σ value decrease systematically.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2} \tag{3.3}$$

In the Equation 3.3, σ represents the standard deviation, N represents the data count, x_i represents the $i.$ value in dataset, and $\bar{x}$ represents the mean value of all x values.

As an example, we examined the distributions of altitude values of weather stations (Table 3.1). According to this examination, weather stations at 852.86 meters height mean and the σ value of altitude distribution is 625.71. However, as can be observed in the Table 3.1, the weather station at the lowest point is at 330 meters under sea level. Since submarine weather stations don't collect any wind data, we removed these stations from the dataset before determining geospatial dimensions' normalization coefficients.

Table 3.1: Altitude distribution of weather stations

Count	1929
Mean	852.86
Standard deviation (σ)	625.71
Minimum	-330.00
%25	208.00
%50	891.00
%75	1277.00
Maximum	2937.00

The disadvantage of normalizing geospatial dimensions with this approach is the determined normalization coefficient is strongly dependent on the number of measurement points in the dataset. Since we use the standard deviation value as the normalization coefficient, if the count of measurement points is lesser, the standard deviation will probably be higher, and farther points will be selected as neighbors, too, during the neighbor selection process. This could create a risk if the area worked on has a sharp surface or climate change. Thus, collecting data from different points as much as possible is essential for an uneven work field. In our future work, we aim to develop a more stable and location-free solution for geospatial distance normalization.

3.1.3 Feature Transformation

After the normalization process, we selected neighbor points for the train and target datasets (Figure 3.3 - 3). After neighbors were selected, these neighbors' features should be added to the reference point. Our way of feature transformation is based on adding selected neighbors' related columns to the reference point's row. For example, if our data has 1 target column and n feature columns including time and location information, after k neighbors selection and feature transformation, our new dataset will have $(k+1)*(n+1)-1$ feature columns and still 1 target column. We applied this transformation to both real and target datasets. During target data transformation, historical real data (Figure 3.3 - A) and location and time information of target point (Figure 3.3 - B) were used; while transforming real data, only historical real data (Figure 3.3 - A) was used. In this way, target data neighbors are selected from real data points. Transformed real data (Figure 3.3 - C) was used for generative model training, while transformed target data (Figure 3.3 - D) was used as conditions of generator.

3.2 Generative Model

A generative model was used in this work for more realistic wind map generation in a shorter time (Figure 3.3 - 4). We trained this generative model with transformed real data with neighborhood information (Figure 3.3 - C). During the training process, neighborhood information was used as conditions of the generative model. After the training, synthetic wind speed data was generated using transformed target data

containing target location, time interval, and neighborhood information (Figure 3.3 - D). Like the training process, neighborhood information was used as conditions of the generator during the synthetic wind speed data generation process. As the result of this process, synthetic target data (wind map) (Figure 3.3 - E) was generated. As generative model architecture, GAN was used.

3.2.1 Motivation of GAN Usage

GAN is widespread interest in the academy. Some of the recent studies about GAN are comprehensively examined in the Related Work chapter (Chapter 2). In some of these studies, GAN has been used in the presence of both one-dimensional correlations like time series and multi-dimensional correlations like images. The success of GAN in both of these problems has increased the interest in the GAN method.

GAN is widely used for image augmentation. Most studies show that GAN outperforms other generative and interpolation-based models at spatial data augmentation problems. Moreover, since the GAN's generator is fed with a random space vector, GAN can generate various data while preserving multi-dimensional correlations. This stochasticity and the ability to protect spatial correlations of GAN are our main motivations for using GAN architecture. Since our data is in spatial form, we also used some techniques we inspired from image augmentation studies like neighborhood analysis with KNN. On the other hand, we added time-dependent conditions to GAN to preserve the time-series characteristics of the original data. In this way, we developed a novel spatial time-series data generation approach with GAN, which is the main contribution of this work.

There are two primary motivations for generating wind speed data instead of generating wind power data directly. The first motivation is that the generative models may sometimes fail to preserve complex correlation between input and output data [21]. Since there is a complex correlation between wind speed and the energy production of a wind farm, we decided to generate wind speed data with a generative model and used a prediction model to predict wind power generation. The second motivation is that using a separate model for predicting power output provides flexibility to the proposed method. In this way, different wind farm capacities may be tested rapidly. Thus, the decision-making time will be reduced.

3.2.2 GAN Structure

GAN models consist of two different neural networks; the generator and discriminator networks. This architecture is a two-player game in which the generator and the discriminator compete with each other until real-like synthetic data is generated; hence this model counts as "adversarial" (Figure 3.6). Through the training process, the generator tries to fool the discriminator and improves itself according to the discriminator's outputs. Meanwhile, the discriminator learns the generator's tricks and tries to become invulnerable to fake data. When these two models are met at an optimum point, synthetic data becomes ready to use as near-real data. Since the generator learns the patterns from unlabeled data, the training process is unsupervised. How-

ever, the discriminator uses supervised loss. Hence, GAN architecture is classified as a semi-supervised learning process or an unsupervised learning process that uses supervised loss.

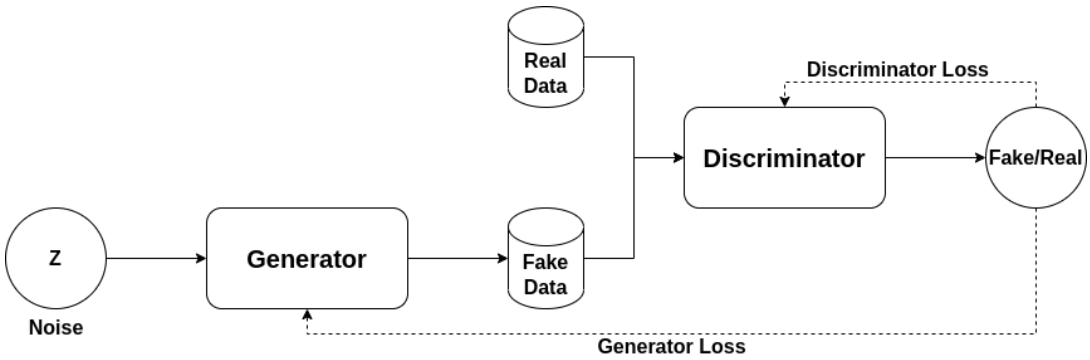


Figure 3.6: GAN Architecture

Conditional GAN (cGAN) architecture was used as the GAN model in this work (Figure 3.7). Target points' location, time, and neighborhood information were used as conditions, and synthetic wind speed data was generated. These conditions were given to the generator with noise vector (latent space) and also were given to the discriminator. Using conditions has made the model more robust to common GAN problems such as vanishing gradients and mode collapse. Also, sample quality has increased.

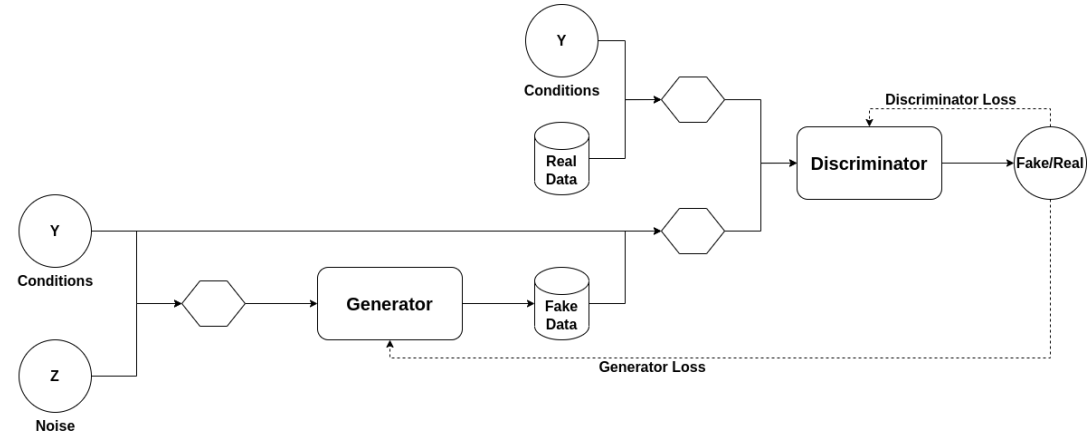


Figure 3.7: Conditional GAN Architecture

3.2.2.1 Generator & Discriminator

The main concerns of GAN structure design are making the model more likely to converge and avoiding mode collapse and vanishing gradient problems. Various techniques are used to prevent these problems of GAN, such as optimizers, activators, normalizations, etc. Due to its generalization ability [31], we used Stochastic Gradient Descent (SGD) Optimizer at both the generator and the discriminator. We used different activation functions at the generator and the discriminator according to the

activation function's layer. Moreover, we used Standard Scaler, a mean subtraction normalization algorithm, at both the generator and the discriminator. Also, using cGAN architecture made the generator model more robust against mode collapse and vanishing gradient problems.

The standard GAN architecture, a.k.a. Vanilla GAN, uses the min-max loss as the GAN loss function [3]. In this way, the generator tries to minimize this function while the discriminator tries to maximize it. The GAN loss function can be further divided into two categories: Discriminator loss and Generator loss. Since we work on continuous data, we used a regression loss function as the generator's loss: Mean Squared Error (MSE). Due to its symmetrical behavior, MSE penalizes all samples equally regardless of their position on the decision boundary, positive or negative side. This way, the samples bigger or lower than the target wind speed data would be penalized equally, and the samples will get closer to real wind speed measurements. The usage of MSE as a loss function makes the model harder to converge, but with sufficient data, this won't be a problem. On the other hand, the discriminator classifies inputs as real or fake. Hence, we used Binary Cross-Entropy (BCE) loss as the discriminator's loss function, which is a binary classification loss function.

The structures of the generator and the discriminator are detailed below:

Generator: The generator model consists of 4 layers: 1 dense input layer, 2 dense hidden layers, and 1 output layer. ReLU is used as an activation function at all layers.

Discriminator: The discriminator model consists of 4 layers: 1 dense input layer, 2 dense hidden layers, and 1 output layer. The Tanh function is used as the activation function at the input and hidden layers. Since the discriminator's output is a scalar value between 0-1, the Sigmoid function is used as the activation function at the output layer.

3.3 Prediction Model

The prediction model was used for past production prediction (Figure 3.3 - 5) of a candidate wind farm using wind maps that were generated in previous steps of the process (Figure 3.3 - E). We trained this model with a known wind farm's past productions and real wind speed measurements (Figure 3.3 - F). This reference wind farm was selected for all experiments differently according to its similarity to the wind farm planned to build in terms of geographic conditions and production capacity. To increase the accuracy of predictions, we selected different wind farms for training in different scenarios instead of training the model with a single reference wind farm and scaling up and down the predictions according to planned production capacity. However, a single trained model for different scenarios will decrease the complexity of the problem. Nevertheless, we stuck to the scenario-specific reference wind farm selection method to reach higher accuracies. At the end of the process, predicted productions of the candidate wind farm for a defined time interval emerged (Figure

3.3 - G). This data is the primary reference for the profitability analysis of a candidate wind farm.

We used wind speed values as input of the prediction model and predicted power outputs. However, wind direction is another parameter affecting a wind farm's power output. Although a single turbine's power output doesn't change with wind direction, shadowing effects may change. There are two shadowing factors that may affect wind energy production; the effects of turbines on each other and the effects of landform on turbines. Since wind farms are designed with minimum shadowing effect concern, wind direction usually has a negligible impact on wind energy production. Nevertheless, since changing wind direction has a temporal effect on wind energy production data, the effects of changing wind direction can also be predicted using a prediction model sensitive to temporal dynamics.

With this model, we aim to simulate all scenarios a wind farm could face, such as breakdowns, maintenance operations, or changing shadowing effects. Thus, we used a neural network-based model for prediction instead of power curves which is a common approach for this process. In this way, prediction results give more realistic insights into the profitability of a candidate wind farm. On the other hand, modeling ideal production curves and breakdown/maintenance scenarios separately may increase accuracies. We aim to work on this method in our future studies.

3.3.1 Method Selection

Predicting the production of a wind farm is a complex task due to various effects on wind power production, such as breakdowns or shadowing effects. Due to this complexity, we decided to use a machine learning approach that shows higher performance on complex problems such as wind power production prediction [1]. However, besides non-linearity and complexity, there is another concern in our case that should also be satisfied: temporality. A breakdown started at previous hours or changing shadowing effect according to wind direction are examples of temporality. We used the LSTM network in the prediction model to preserve these temporal dynamics. LSTM is a special kind of RNN that is capable of learning sequence dependence. Moreover, the LSTM network is resistant to noise and fluctuations. Since we use GAN-generated data as input for the prediction model, resistance to fluctuations is another critical requirement of the prediction model besides preserving temporal dynamics. Studies about LSTM show that the LSTM network can satisfy both of our requirements. Hence, we built our prediction model with LSTM layers.

3.3.2 Prediction Model Structure

The prediction model consists of 2 layers; 1 dense LSTM layer and 1 output layer. The ReLU activation function is used at both the input and output layers.

CHAPTER 4

EXPERIMENTAL WORK

In this chapter, we conducted our experiments with the steps comprehensively explained in the Methodology chapter (Figure 3.3) and presented the results. Three different experiment cases were conducted, and the results were evaluated and compared. Also, obtained insights were explained.

4.1 Dataset

During this work, we used 1929 different meteorology stations' 5 years of archive data (lesser data from newer stations) provided by the Turkish State Meteorological Service (MGM) [32]. This archive data contains wind speed, wind direction, temperature, relative humidity, cloud cover, ceiling height, precipitation, and pressure values on an hourly basis. All of these meteorology stations are located in Turkey. The exact positions of weather stations on the map can be seen in the Figure 4.1. Due to Turkey's geological position, observing different conditions, such as the marine effect or effects of altitude changes, is possible with this data. Since we aim to use neighborhood information in synthetic data generation, we trained models with near stations' data in a limited radius, not the whole data. For different experiments, different subsets of the original dataset were used. In this way, training cost is reduced. We generated only synthetic wind speed data, and other parameters were used as features in generative models. All meteorology stations' latitude, longitude, and altitude data are also available for this work, and we used this information for the neighbor selection process.

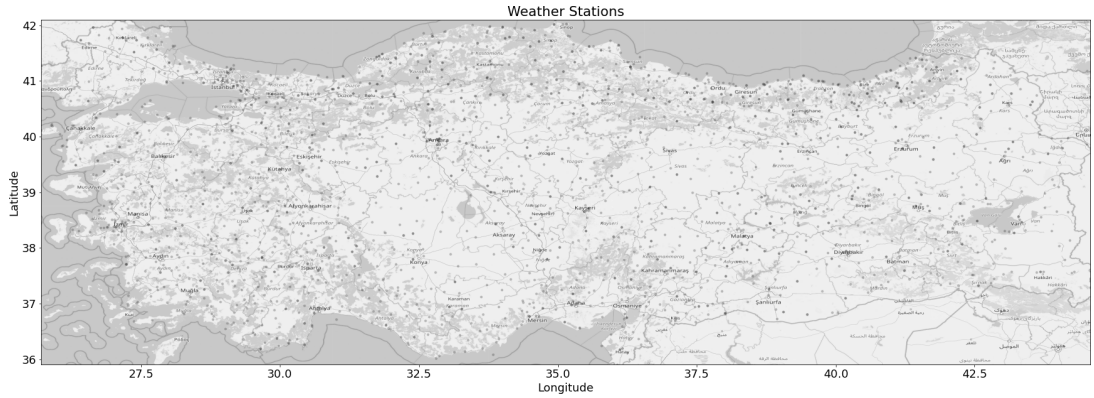


Figure 4.1: Exact locations of weather stations that are data collected from

Due to their high wind potential and installed wind power capacity, we selected İzmir, Balıkesir, and Manisa regions for experiments. According to the Turkish Wind Energy Association's 2021 report [33], these regions have more than %40 of Turkey's total installed wind energy capacity. These regions also have the highest wind energy potentials in Turkey. Thus, predicting these regions' wind speed values is more challenging than in other regions of Turkey. This is another motivation for us to select these regions as experiment cases.

Weather station archive data of selected regions were used for GAN training. For feasibility analysis, three different wind farms were selected from these regions: Soma Wind Farm from Manisa, Bandırma Wind Farm from Balıkesir, and Zeytineli Wind Farm from İzmir (Table 4.1). All of the steps of the proposed method mentioned in the Methodology chapter (Chapter 3) were applied to these three wind farms. We compared outputs of the processes with real past productions of these farms and evaluated the suggested feasibility method. Wind power production data of selected wind farms were obtained from EXIST Transparency Platform [34], which is open data API of the Turkish energy market regulator.

Table 4.1: Details of wind farms that are selected for experiments

Case	Region	Capacity (MW)	Easting	Northing	Altitude (m)
Bandırma	Balıkesir	50	591749.12	4469093.43	286
Soma	Manisa	120	556276.03	4349521.53	668
Zeytineli	İzmir	50	453057.79	4234215.43	127

4.2 Normalization Coefficients Determination

There are five different dimensions in our case: Hour and day of the year as time dimensions and easting, northing, and altitude as geospatial distance dimensions. We determined normalization coefficients of all dimensions separately for all experiments.

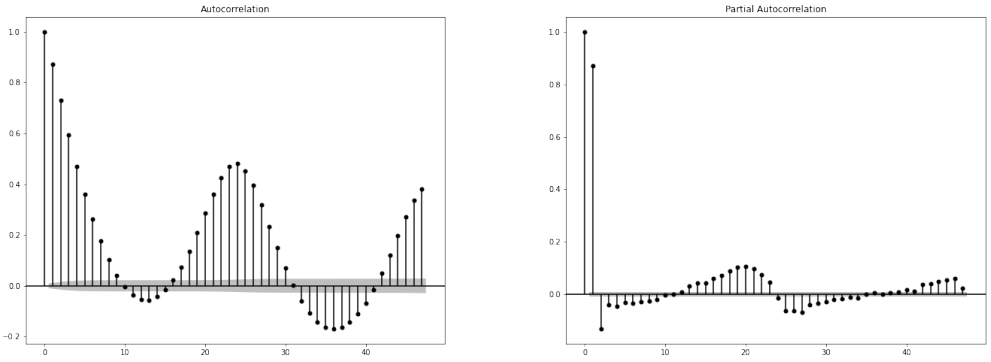
4.2.1 Time Normalization Coefficients Determination

To determine normalization coefficients of time dimensions, we selected the closest weather stations to selected wind farms and examined ACF and PACF analyses of these weather stations' wind speed data on both an hourly and a daily basis. We used our domain expertise while determining the normalization coefficients by ACF-PACF analyses. Dark areas in ACF-PACF plots represent confidence intervals.

4.2.1.1 Hour Dimension Normalization

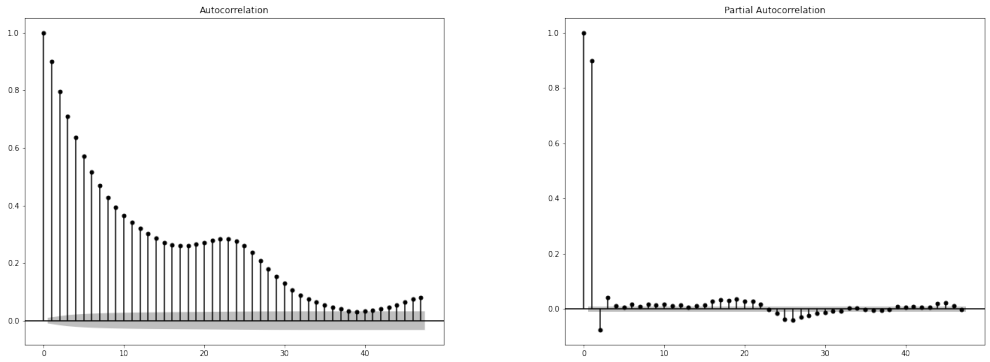
Three different weather stations were selected from Bandırma, Soma, and Zeytineli regions for ACF-PACF analyses. These weather stations' hourly data were examined

to determine the normalization coefficients of the hour dimension. Results of the ACF-PACF analyses are presented in Figures 4.2, 4.3, and 4.4.



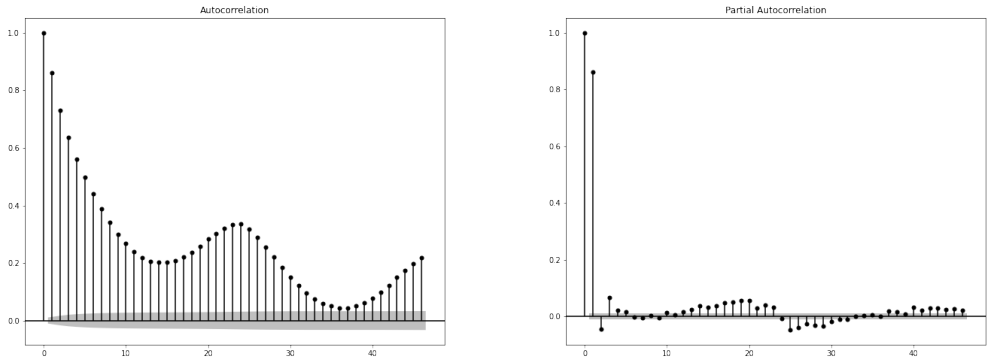
(a) ACF analysis (b) PACF analysis

Figure 4.2: Bandırma/Balıkesir weather station's wind speed ACF (a) - PACF (b) analyses on an hourly basis



(a) ACF analysis (b) PACF analysis

Figure 4.3: Soma/Manisa weather station's wind speed ACF (a) - PACF (b) analyses on an hourly basis



(a) ACF analysis

(b) PACF analysis

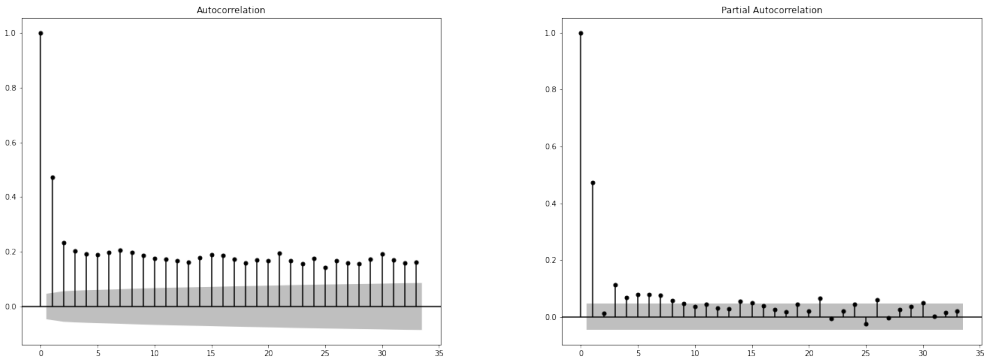
Figure 4.4: Zeytineli/İzmir weather station's wind speed ACF (a) - PACF (b) analyses on an hourly basis

Hourly autocorrelations of all cases show similar behaviors. Autocorrelations are decreasing systematically through lag and due to daily seasonality in wind speeds, slightly increase at 24 hours and multiples. However, since we select neighbors in mixed form, not in a series form, PACF results are more critical for normalization coefficient determination. Autocorrelations show a combination of correlations between consecutive points, while partial-autocorrelations show a direct correlation. Thus, we selected the lag count as the normalization coefficient of the time dimension, which indicates considerably higher correlations in PACF analyses than the confidence interval.

According to hourly ACF-PACF analyses of the weather stations, the normalization coefficients of the hour dimension were selected as 2 for Bandırma (Figure 4.2) and Soma (Figure 4.3) cases and 3 for Zeytineli (Figure 4.4) case.

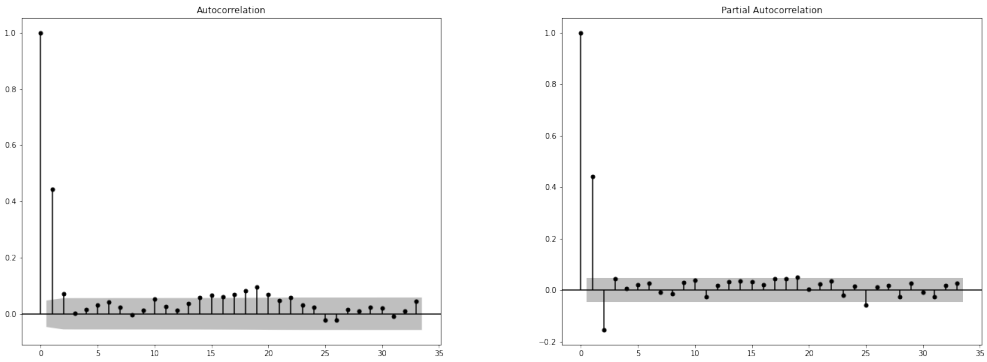
4.2.1.2 Day Of Year Dimension Normalization

Since our dataset in an hourly basis, at first, we downsampled our data to a daily basis with mean values. After that, we examined ACF-PACF analyses of daily wind speed averages. Three weather stations' data from Bandırma, Soma, and Zeytineli regions were analyzed. Results of the ACF-PACF analyses are presented in Figures 4.5, 4.6, and 4.7.



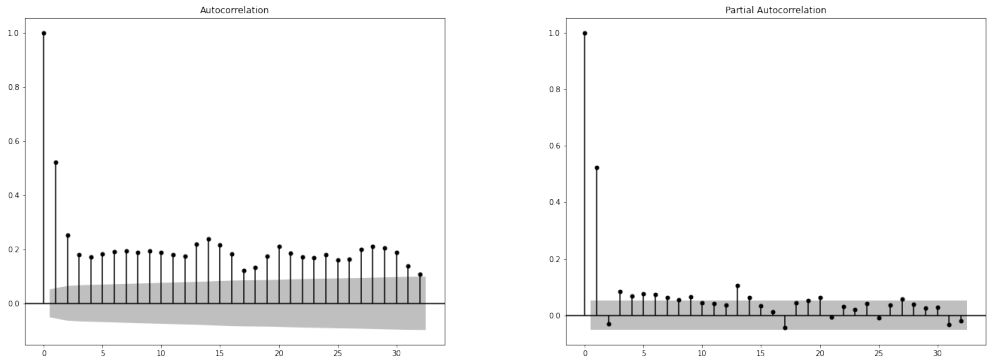
(a) ACF analysis (b) PACF analysis

Figure 4.5: Bandırma/Balıkesir weather station's wind speed ACF (a) - PACF (b) analyses on a daily basis



(a) ACF analysis (b) PACF analysis

Figure 4.6: Soma/Manisa weather station's wind speed ACF (a) - PACF (b) analyses on a daily basis



(a) ACF analysis

(b) PACF analysis

Figure 4.7: Zeytineli/İzmir weather station's wind speed ACF (a) - PACF (b) analyses on a daily basis

We selected the lag count that shows a higher correlation than the confidence interval in ACF and PACF analyses as the normalization coefficient of the day of year dimension. According to daily ACF-PACF analyses of the weather stations, the normalization coefficients of the day of year dimension were selected as 1 for Bandırma (Figure 4.5) and Zeytineli (Figure 4.7) cases and 2 for Soma (Figure 4.6) case.

Determined time dimension normalization coefficients for all cases are listed in the Table 4.2.

Table 4.2: Normalization coefficients of time dimensions for all cases

Experiment Case	**Normalization Coefficient**	
	Hour	**Day of Year**
Bandırma	2	1
Soma	2	2
Zeytineli	3	1

4.2.2 Geospatial Distance Normalization Coefficients Determination

To determine the normalization coefficients of geospatial distance dimensions, we examined the distribution of weather stations on the map. Since our dataset has data from all over Turkey, we only used the weather stations' data in the same city as the target wind farm. Moreover, we removed submarine weather station data from the dataset since these stations don't collect any wind data.

4.2.2.1 Easting Dimension Normalization

We examined easting value distributions of weather stations from Balıkesir, Manisa, and İzmir and selected standard deviation values as normalization coefficients of the easting dimension. Results are presented in the Tables 4.3, 4.4, and 4.5.

Table 4.3: Easting distribution of weather stations in Balıkesir

Count	34
Mean	568854.427951
Standard deviation (σ)	49895.390456
Minimum	469021.005917
%25	544285.079485
%50	572250.996935
%75	597625.810972
Maximum	661690.731254

Table 4.4: Easting distribution of weather stations in Manisa

Count	26
Mean	585699.001737
Standard deviation (σ)	43696.765036
Minimum	521928.023188
%25	551008.519808
%50	575914.156552
%75	621290.937864
Maximum	662256.876026

Table 4.5: Easting distribution of weather stations in İzmir

Count	54
Mean	511507.825837
Standard deviation (σ)	35681.800651
Minimum	436047.494453
%25	490953.886490
%50	509483.909331
%75	522419.987349
Maximum	607277.139207

According to distributions, the normalization coefficients of the easting dimension were selected as 49895.390456 for Bandırma case (Table 4.3), 43696.765036 for Soma case (Table 4.4), and 35681.800651 for Zeytineli case (Table 4.5).

4.2.2.2 Northing Dimension Normalization

We examined northing value distributions of weather stations from Balıkesir, Manisa, and İzmir and selected standard deviation values as normalization coefficients of the northing dimension. Results are presented in the Tables 4.6, 4.7, and 4.8.

Table 4.6: Northing distribution of weather stations in Balıkesir

Count	34
Mean	4403991
Standard deviation (σ)	44432.85
Minimum	4351370
%25	4370465
%50	4391505
%75	4429887
Maximum	4501037

Table 4.7: Northing distribution of weather stations in Manisa

Count	26
Mean	4288103
Standard deviation (σ)	25652.39
Minimum	4234389
%25	4267863
%50	4288812
%75	4305644
Maximum	4337894

Table 4.8: Northing distribution of weather stations in İzmir

Count	54
Mean	4258145
Standard deviation (σ)	35025.68
Minimum	4199477
%25	4238045
%50	4250196
%75	4263268
Maximum	4353063

According to distributions, the normalization coefficients of the northing dimension were selected as 44432.85 for Bandırma case (Table 4.6), 25652.39 for Soma case (Table 4.7), and 35025.68 for Zeytineli case (Table 4.8).

4.2.2.3 Altitude Dimension Normalization

We examined altitude value distributions of weather stations from Balıkesir, Manisa, and İzmir and selected standard deviation values as normalization coefficients of the altitude dimension. Results are presented in the Tables 4.9, 4.10, and 4.11.

According to distributions, the normalization coefficients of the altitude dimension were selected as 464.401606 for Bandırma case (Table 4.9), 315.658257 for Soma case (Table 4.10), and 246.597055 for Zeytineli case (Table 4.11).

Determined all dimension normalization coefficients for all cases are listed in the

Table 4.9: Altitude distribution of weather stations in Balıkesir

Count	34
Mean	345.235294
Standard deviation (σ)	464.401606
Minimum	2
%25	20.25
%50	102
%75	599.25
Maximum	1733

Table 4.10: Altitude distribution of weather stations in Manisa

Count	26
Mean	305.846154
Standard deviation (σ)	315.658257
Minimum	37
%25	92
%50	191.5
%75	436.25
Maximum	1238

Table 4.11: Altitude distribution of weather stations in İzmir

Count	54
Mean	170.925926
Standard deviation (σ)	246.597055
Minimum	2
%25	10.5
%50	66.5
%75	193.5
Maximum	965

Table 4.12.

Table 4.12: Normalization coefficients of all dimensions for all cases

Case	**Normalization Coefficient**				
	Hour	**Day of Year**	**Easting**	**Northing**	**Altitude**
Bandırma	2	1	49895.390456	44432.85	464.401606
Soma	2	2	43696.765036	25652.39	315.658257
Zeytineli	3	1	35681.800651	35025.68	246.597055

4.3 *k* Value Determination

We used a multi-variate quadratic (second-degree polynomial) regression model to determine the k value of the KNN algorithm. The target's real historical wind speed

data was tried to predict with neighbors' wind speed data. After normalization coefficients were determined, the distance function of the KNN algorithm was updated, and neighbors were selected according to this function. Different k neighbors were selected in every iteration, and the target wind speed value was predicted with the neighbors' wind speed values. After prediction, the predicted values were evaluated with the RMSE metric, and the optimum k value was selected.

In Figures 4.8, 4.9, and 4.10 can be observed that, k value change gives similar outputs for all cases. This can be considered proof of the suggested normalization methods' consistency since normalization coefficients were determined independently for all cases. According to k value-prediction RMSE graphs, a neighbor count lower than 40 gives more unstable outputs. However, over 40, RMSE scores are decreasing systematically, and the error curve flattens around k=100. Due to lower variance, we selected k value as 100 for all cases; Bandırma (Figure 4.8), Soma (Figure 4.9), and Zeytineli (Figure 4.10) instead of lower values than 40. On the other hand, a bigger k value selection may cause overfitting. However, since we process a big dataset, the generator model is less likely to overfit. Besides, RMSE change over k value change graphs show that models stabilize around 100 neighbors, and there is no sign of overfitting.

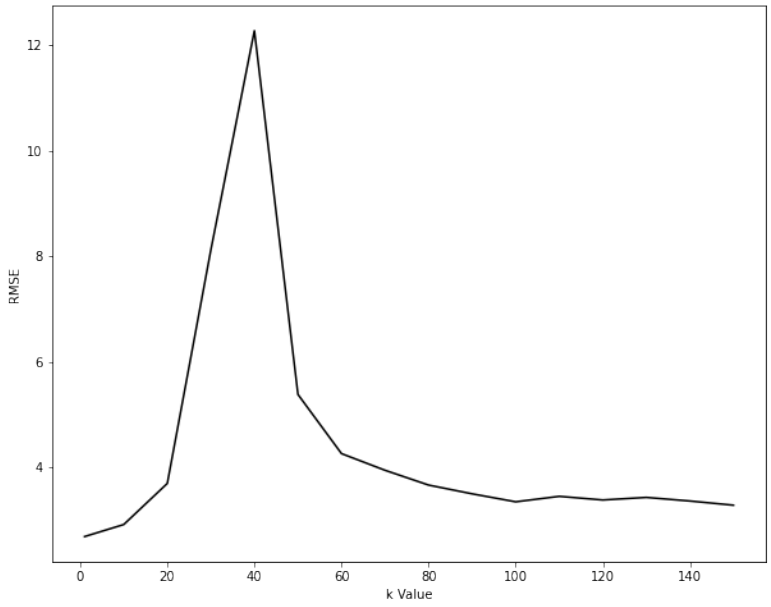


Figure 4.8: RMSE scores of regression models trained with different k neighbors for the Bandırma case

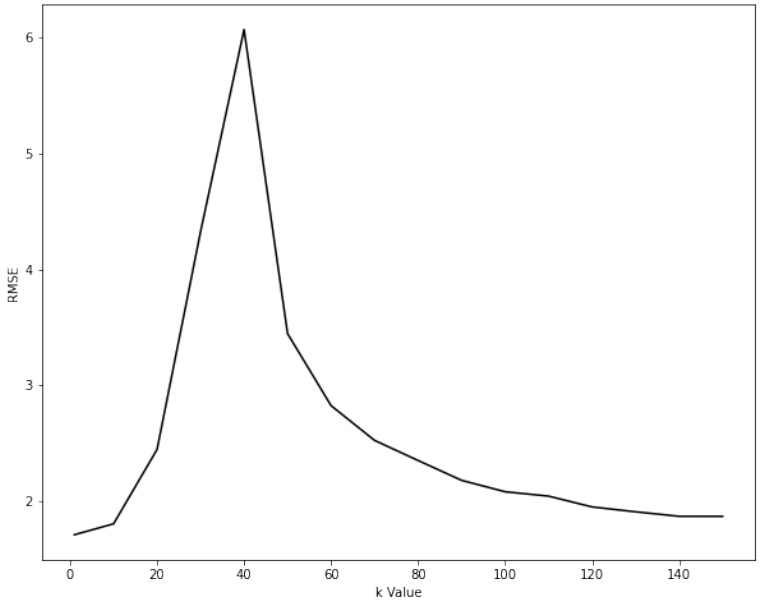


Figure 4.9: RMSE scores of regression models trained with different k neighbors for the Soma case

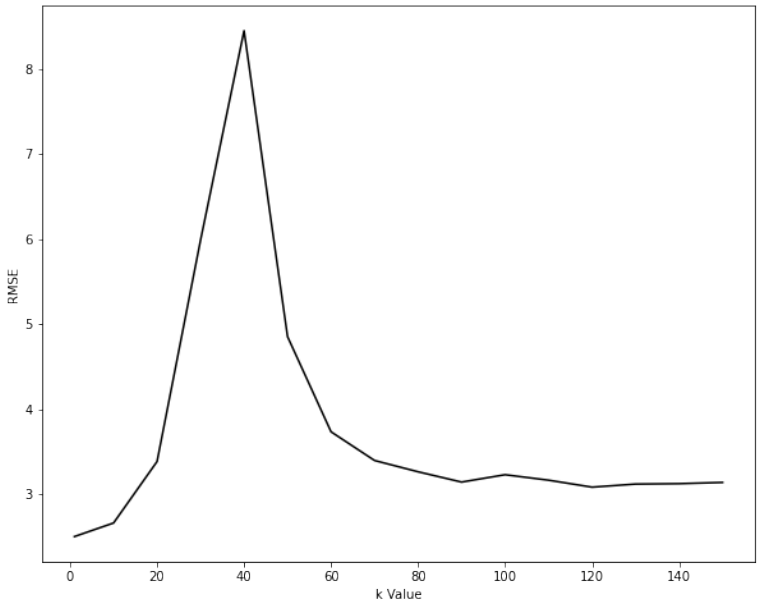


Figure 4.10: RMSE scores of regression models trained with different k neighbors for the Zeytineli case

Determined all dimension normalization coefficients and k values for all cases are listed in the Table 4.13.

Table 4.13: Normalization coefficients and k values of all cases

Case	**Normalization Coefficient**					***k***
	Hour	**Day of Year**	**Easting**	**Northing**	**Altitude**	
Bandırma	2	1	49895.390456	44432.85	464.401606	100
Soma	2	2	43696.765036	25652.39	315.658257	100
Zeytineli	1	1	35681.800651	35025.68	246.597055	100

4.4 Data Transformation

After neighbor selection, we added neighbors' (nn = nearest neighbor) location, time and wind speed to main data (Table 4.14, Figure 3.3 - A). We used these new features as conditions of GAN at later steps. A sample resulting is presented in the Table 4.15 (Figure 3.3 - C). For readability purposes, tables' transposes are presented. The abbreviations of features are also presented in tables, and neighbor points' features are named with these abbreviations. For example, "nn_ws_1" means the first nearest neighbor's wind speed value.

Table 4.14: Sample input data

Row Number	**0**	**1**	**2**	**3**
Wind Speed (ws)	1.2	6.2	0.9	2.2
Hour (h)	16	13	0	13
Day of Year (doy)	209	47	198	310
Easting (e)	535250	572411	533374	597982
Northing (n)	4.274165e+06	4.295967e+06	4.273025e+06	4.260016e+06
Altitude (a)	71	79	212	111

Table 4.15: Sample training data derived from input data with neighborhood information

Row Count	**0**	**1**	**2**	**3**
Wind Speed (ws)	1.2	6.2	0.9	2.2
Hour (h)	16	13	0	13
Day of Year (doy)	209	47	198	310
Easting (e)	535250	572411	533374	597982
Northing (n)	4.27417e+06	4.29597e+06	4.27303e+06	4.26002e+06
Altitude (a)	71	79	212	111
nn_ws_1	6.1	1	1.9	9
nn_ws_2	4.4	0.8	0.7	2
...				
nn_ws_99	1.1	2.2	5.1	1.2
nn_ws_100	1.5	2	0.8	2.4
nn_h_1	14	12	6	15
nn_h_2	20	9	2	8
...				
nn_h_99	18	11	11	6
nn_h_100	6	15	2	17
nn_doy_1	204	49	194	312
nn_doy_2	206	49	192	312
...				
nn_doy_99	287	210	277	172
nn_doy_100	128	212	279	172
nn_e_1	579417	561579	521928	561579
nn_e_2	542083	621590	539071	597982
...				
nn_e_99	597982	648228	554244	621590
nn_e_100	539071	621590	542083	542083
nn_n_1	4.32032e+06	4.26499e+06	4.28437e+06	4.26499e+06
nn_n_2	4.33473e+06	4.28976e+06	4.26736e+06	4.26002e+06
...				
nn_n_99	4.26002e+06	4.23439e+06	4.33789e+06	4.28976e+06
nn_n_100	4.26736e+06	4.28976e+06	4.33473e+06	4.33473e+06
nn_a_1	209	45	320	45
nn_a_2	134	250	1238	111
...				
nn_a_99	111	203	166	250
nn_a_100	1238	250	134	134

4.5 Data Generation

We generated a GAN model with the structure described in the Methodology chapter (Chapter 3) and trained this model with different subsets of the dataset for each case. These trained models generated synthetic wind speed data for the target location and

time period. Generated data were compared with real measurements, and the model was evaluated. The results of each case are presented separately.

4.5.1 GAN Model Training

We determined GAN training parameters with the hyper-parameter optimization technique. The same model has been trained over and over again with different parameters. During this process, we tried learning rate parameters between 0.001-1, batch size parameters between 1-72, and epoch parameters between 1000-500000. Also, we compared Adam and SGD optimizers' outputs. The optimum combination of parameters was determined according to the model's loss, and this combination was used during GAN model training.

The learning rate of the generator and the discriminator networks was determined as 0.01, and the SGD optimizer showed better performance than the Adam optimizer. Also, the batch size of GAN inputs was determined as 12. Since learning rate and batch size values were defined as very low relative to dataset size, to avoid underfitting, the iteration count for GAN training was set to a bigger number; 300000. Once the training environment was prepared, we trained the primary GAN model with three different sub-datasets for each case.

4.5.2 Synthetic Data Generation

With trained GAN models, synthetic wind speed data for selected wind farms' locations were generated: Bandırma, Soma, and Zeytineli. Synthetic data was generated for one year (2020/6 - 2021/6), and a feasibility study for this year was conducted.

Plots of real wind speed measurements and generated wind speed data are presented independently. All plots were generated on an hourly basis. The Figures 4.11 and 4.12 presents the result of Bandırma case, Figures 4.13 and 4.14 presents the result of Zeytineli case, and Figures 4.15 and 4.16 presents the result of Soma case. For a better examination of results visually, plots are presented with both all data and partial data separately.

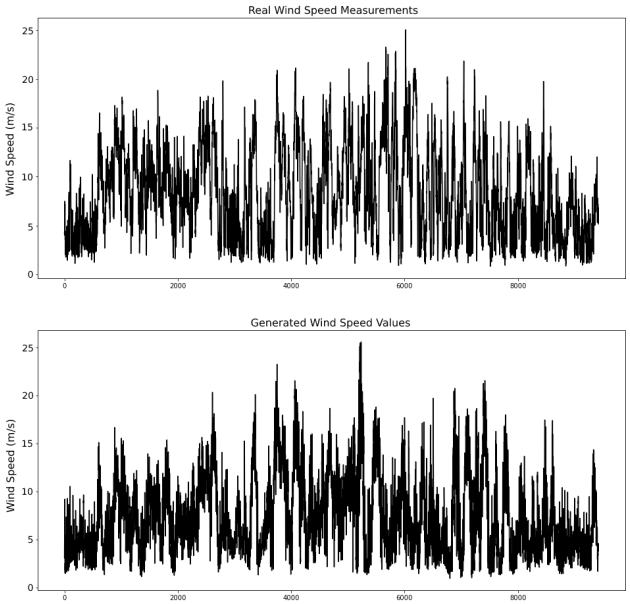


Figure 4.11: Plots of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case

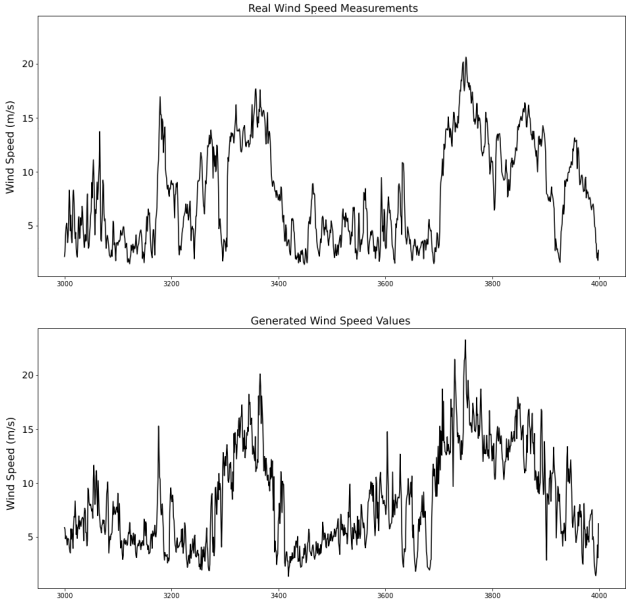


Figure 4.12: Partial plots of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case

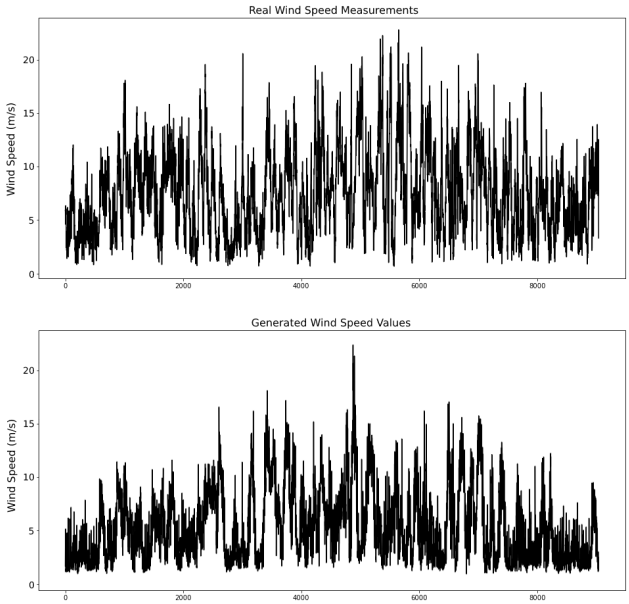


Figure 4.13: Plots of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

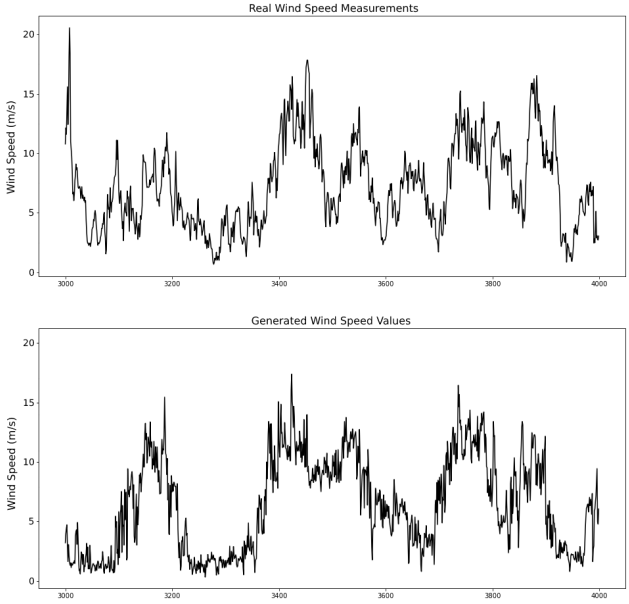


Figure 4.14: Partial plots of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

According to Figures 4.11, 4.12, 4.13, and 4.14, it is observed that generated wind speed data successfully preserved the trend of real wind speed data in the Bandırma and Zeytineli cases. However, the generated data seems to fluctuate more rapidly than real data, and it cannot mimic some sharp speed ups and downs in wind speed. Intense speed ups and downs in wind speed are usually caused by locale pressure or wind direction changes, which usually affect a smaller area. Thus, predicting these wind speed changes is challenging without data collected from the site. However, since we aim to analyze wind in the long term, detecting wind speed trends is more important for our case than detecting temporal wind speed changes. Also, for the same reason, fluctuations in generated wind speed data have a negligible effect on our overall analysis.

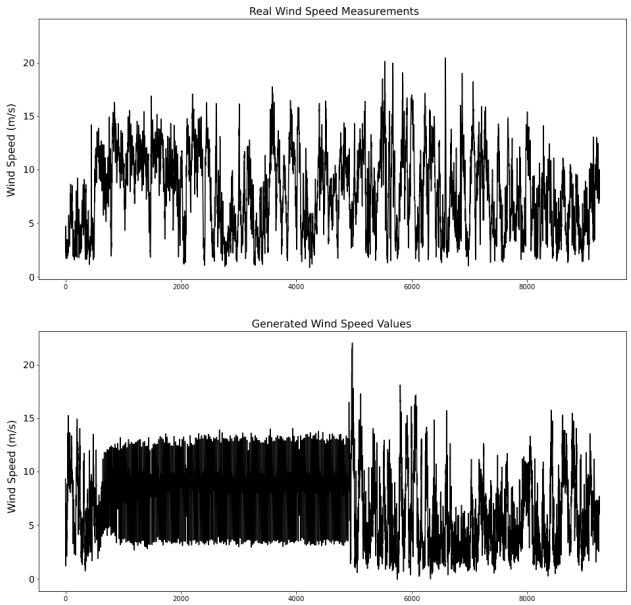


Figure 4.15: Plots of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case

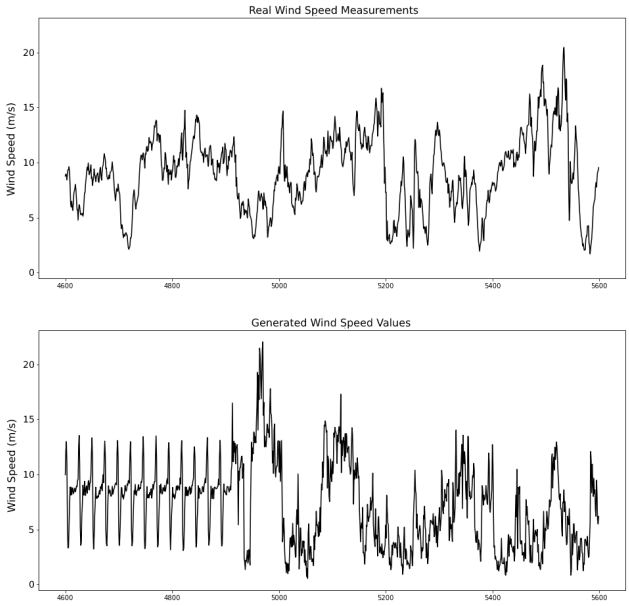


Figure 4.16: Partial plots of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case

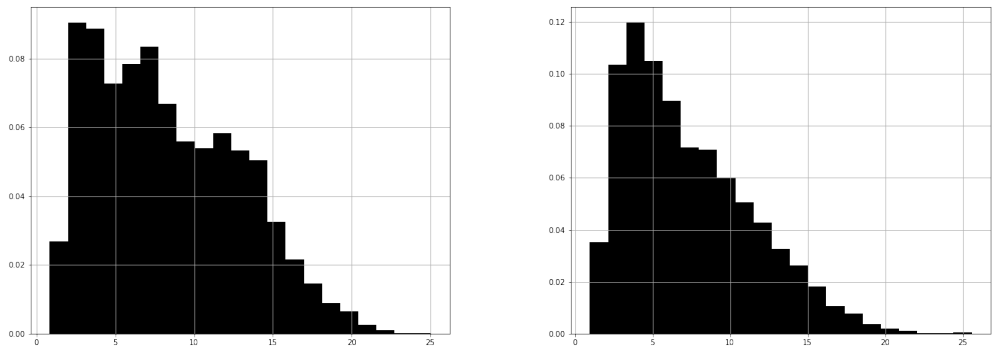
On the other hand, according to Figures 4.15, and 4.16, there is a different issue in the Soma case. It can be observed that the generator followed the same pattern for almost half a year. The KNN algorithm selected similar neighbors for different data points during this period. This is caused by the lack of variety in neighbor points. Soma wind farm is located at 668 meters elevation, which is more than double the mean altitude value of weather stations from the Manisa region (Table 4.10). Also, there are lesser weather stations in Manisa than in Balıkesir and İzmir. Moreover, there are some missing data from Manisa, especially in the period when the same neighbors are selected over and over again. Even though some of the data is missing from all regions, since Soma Wind Farm already has fewer neighbors due to its position, it is affected by missing data or outliers more than other wind farms. Hence, the generator generated similar outputs in the period that the problem occurred. After missing data sources become available again, the number of possible neighbors increases and more various outputs start to be generated. This problem may be overcome with varying normalization coefficients, but since the variety of neighbors is limited, the outputs would still be insufficient. On the other hand, with a more extensive input dataset, this problem may be resolved. Alternative solutions for this issue are presented in the Discussion section. This case also shows us the importance of input data density to generate more realistic wind speed data with the proposed model. We also examined this issue's effects at later steps of the process.

4.5.3 Evaluation of Generated Data

We evaluated generated data with three methods: distribution analysis, discriminator score, and ACF-PACF analyses. Distribution analysis gives us insight into the answer to RQ1. With the discriminator score, we aim to examine whether generated data does look like real data or not. This examination gives us insight into the answers to RQ1 and RQ2. With ACF-PACF analyses, we aim to determine whether the generated data preserves the time-series characteristics of the original dataset or not. This examination also gives us insight into the answer to RQ2.

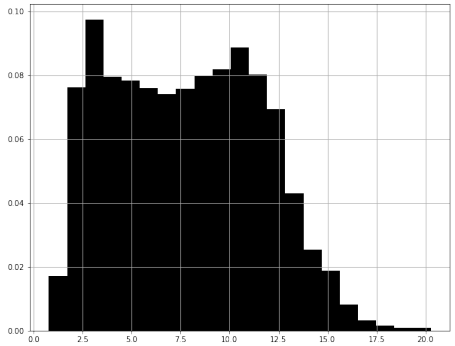
4.5.3.1 Distribution Analysis

To evaluate generated synthetic data, first, we analyzed the distributions of real wind speed measurements and synthetic wind speed values and compared them. Results of the analyses are presented in the Figure 4.17 for Bandırma case, in the Figure 4.18 for Soma case, and in the Figure 4.19 for Zeytineli case.

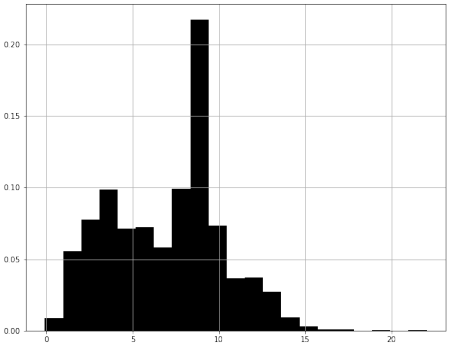


(a) Histogram graph of real wind speed data (b) Histogram graph of generated wind speed data

Figure 4.17: Distribution analyses of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case

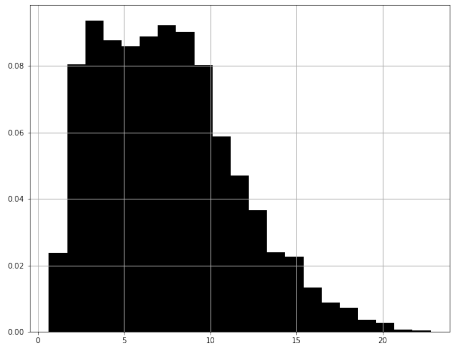


(a) Histogram graph of real wind speed data

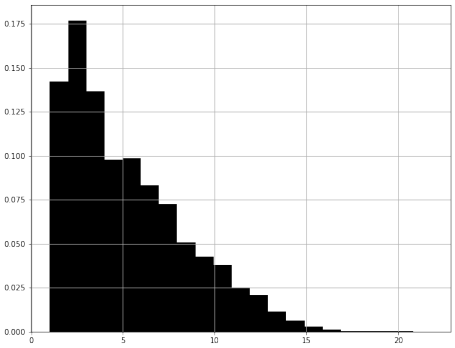


(b) Histogram graph of generated wind speed data

Figure 4.18: Distribution analyses of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case



(a) Histogram graph of real wind speed data

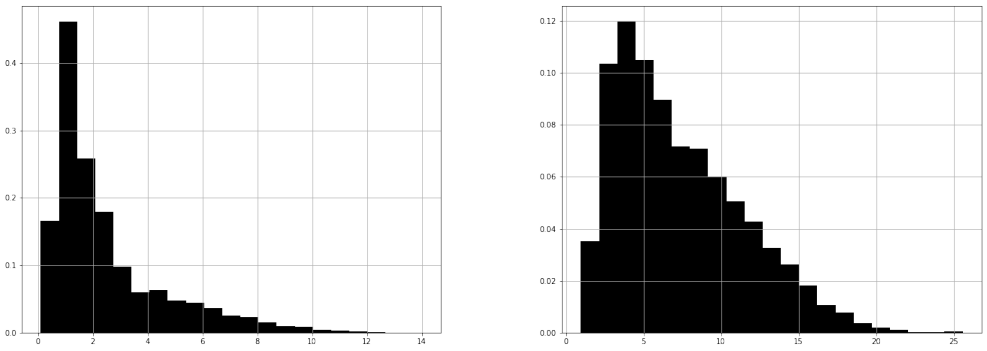


(b) Histogram graph of generated wind speed data

Figure 4.19: Distribution analyses of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

In Figures 4.17, 4.18, and 4.19, it can be observed that wind speed distribution is in right-skewed form at all cases, and the generated data also reflect this form. In the Soma case (Figure 4.18), samples are concentrated in a limited range due to the input data variety problem mentioned before, but still, the distribution is in right-skewed form. However, the main difference between the real data and the generated data distributions is that generated data is concentrated at lower wind speed values compared to real data distribution. The main reason for this behavior is the different regional characteristics of input and target locations. Our input data was gathered from weather stations. Despite there being many weather stations all around Turkey, most of these stations are located near sea-level or urban sites. However, wind farms are usually located at high, non-urban sites. At non-urban sites, fewer obstacles slow down the wind speed. Also, at locations at higher elevations, the wind is stronger and

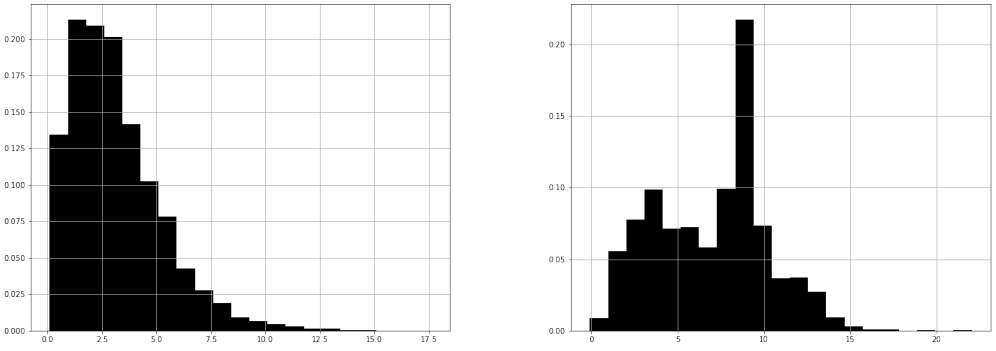
more volatile than at sea level. To evaluate this situation, wind speed distributions of the closest weather stations to each wind farm are presented with the generated data distribution analyses in Figures 4.20, 4.21, and 4.22.



(a) Histogram graph of a weather station from Balıkesir

(b) Histogram graph of generated wind speed data for Bandırma Wind Farm

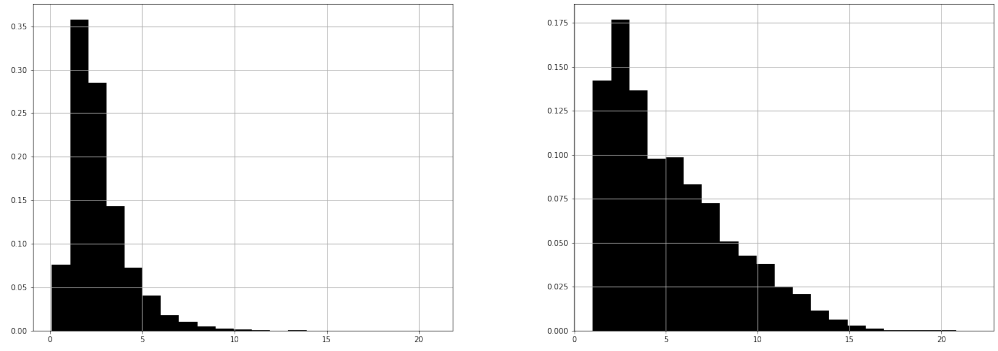
Figure 4.20: Distribution analyses of real wind speed measurements of a weather station from Balıkesir and generated wind speed data for Bandırma Wind Farm



(a) Histogram graph of a weather station from Manisa

(b) Histogram graph of generated wind speed data for Soma Wind Farm

Figure 4.21: Distribution analyses of real wind speed measurements of a weather station from Manisa and generated wind speed data for Soma Wind Farm



(a) Histogram graph of a weather station from İzmir

(b) Histogram graph of generated wind speed data for Bandırma Wind Farm

Figure 4.22: Distribution analyses of real wind speed measurements of a weather station from İzmir and generated wind speed data for Zeytineli Wind Farm

It can be observed in Figures 4.20, 4.21, and 4.22 that, wind speed distributions of weather stations are concentrated at lower wind speed values than wind farms, as we assumed. Also, we can say that the generated wind speed median value is bigger than the weather station's wind speed median and lower than the real wind speed measurements median value. Alternative solutions for this issue are discussed at the end of this chapter. For this process step, preserving the main distribution form of original data is enough to continue the process.

4.5.3.2 Discriminative Score

The discriminator's capability of discriminating between real and synthetic data was calculated as the discriminative score. The same amount of real and synthetic data was given to the discriminator, and the discriminator's loss was examined. The more hesitant the trained discriminator is to distinguish between real and synthetic data, that means the more similar the generated data is to the real data. The examination result is presented in the Table 4.16.

Table 4.16: Discriminative scores of all cases

Case	Data Type	Data Count	Loss
Bandırma	Real	10000	0.69
	Synthetic	10000	0.67
Soma	Real	10000	0.69
	Synthetic	10000	0.69
Zeytineli	Real	10000	0.69
	Synthetic	10000	0.74

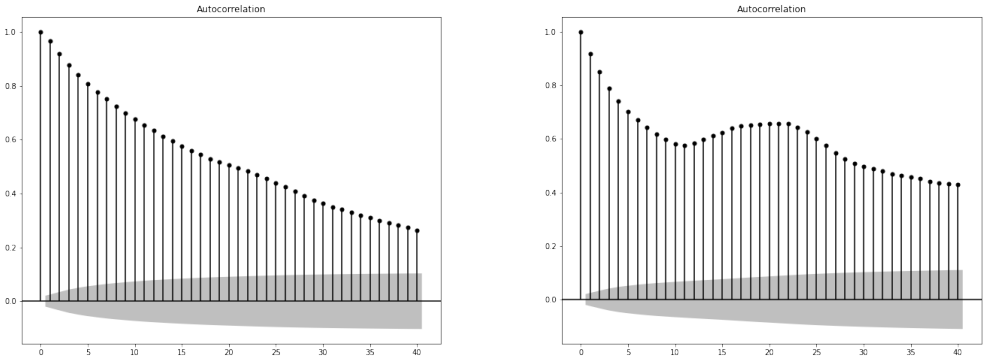
According to the Table 4.16, we can say that the discriminator didn't collapse during training and worked successfully. Since it showed similar performance in discrimi-

nating between real and fake data, the model can be considered stable. On the other hand, the discriminator's loss may not be a good indicator for determining the GAN model's performance. With a stronger discriminator, the GAN model may still generate high-quality data with higher iterations, or with a weaker discriminator, the GAN generator may generate various data that may be useful in specific cases. However, in most studies, even in Goodfellow's first introduction of GAN [3], ideal discriminator loss is mentioned as around 0.69. Our discriminator also satisfies this condition.

4.5.3.3 ACF-PACF Analyses

In this step, we compared the generated data ACF-PACF results with real wind farm data ACF-PACF results to examine similarities in time-series behaviors. In this way, we aim to determine whether generated data preserves the time-series characteristics of original data or not. This examination is the key step for answering RQ2.

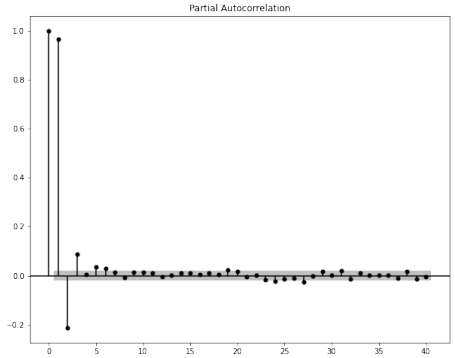
Hourly basis ACF-PACF analyses' results are presented in Figures 4.23, 4.24, 4.25, 4.26, 4.27, and 4.28. Also, daily basis ACF-PACF analyses' results are presented in Figures 4.29, 4.30, 4.31, 4.32, 4.33, and 4.34.



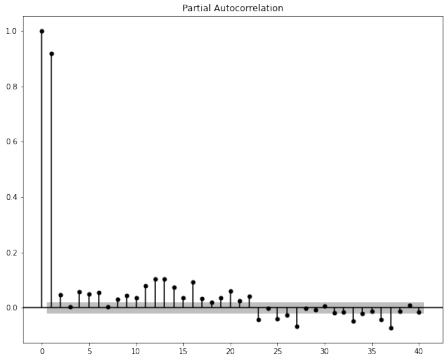
(a) ACF analysis of real wind speed measurements on an hourly basis

(b) ACF analysis of generated wind speed data on an hourly basis

Figure 4.23: Hourly ACF analyses of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case

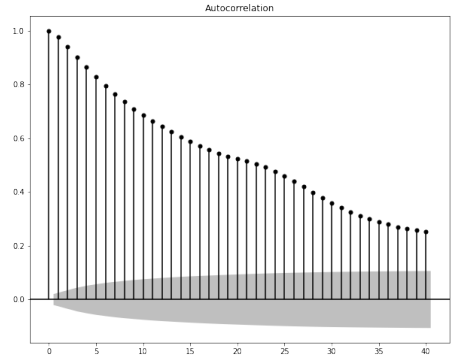


(a) PACF analysis of real wind speed measurements on an hourly basis

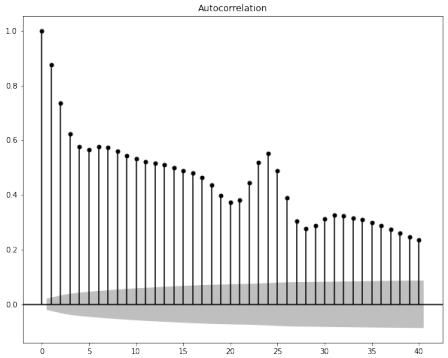


(b) PACF analysis of generated wind speed data on an hourly basis

Figure 4.24: Hourly PACF analyses of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case

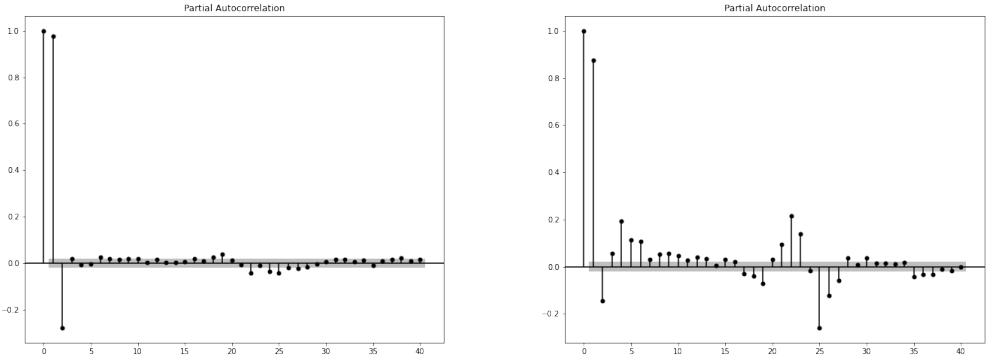


(a) ACF analysis of real wind speed measurements on an hourly basis



(b) ACF analysis of generated wind speed data on an hourly basis

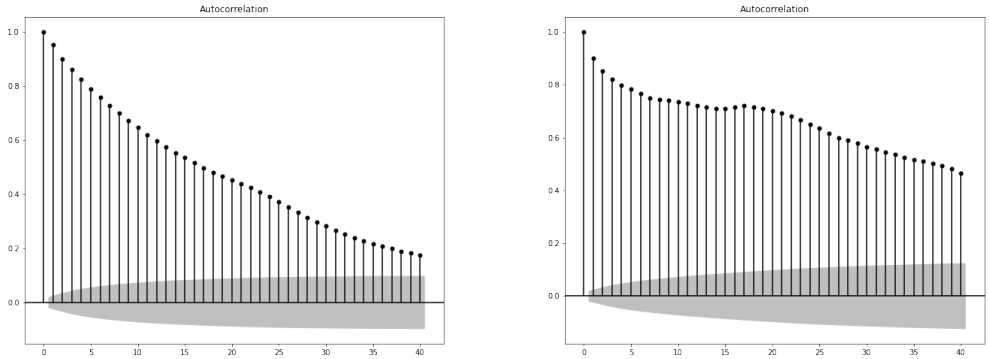
Figure 4.25: Hourly ACF analyses of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case



(a) PACF analysis of real wind speed measurements on an hourly basis

(b) PACF analysis of generated wind speed data on an hourly basis

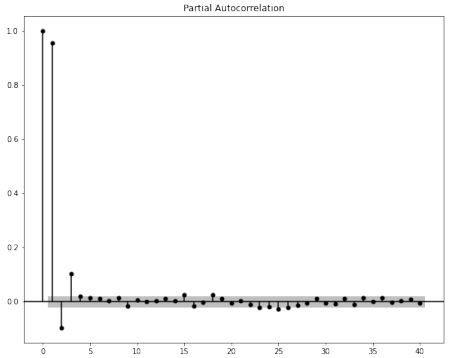
Figure 4.26: Hourly PACF analyses of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case



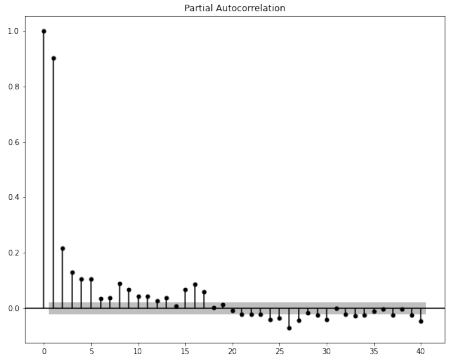
(a) ACF analysis of real wind speed measurements on an hourly basis

(b) ACF analysis of generated wind speed data on an hourly basis

Figure 4.27: Hourly ACF analyses of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

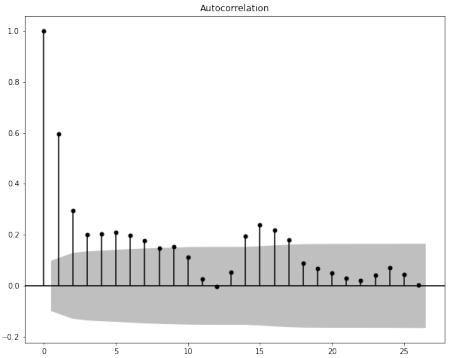


(a) PACF analysis of real wind speed measurements on an hourly basis

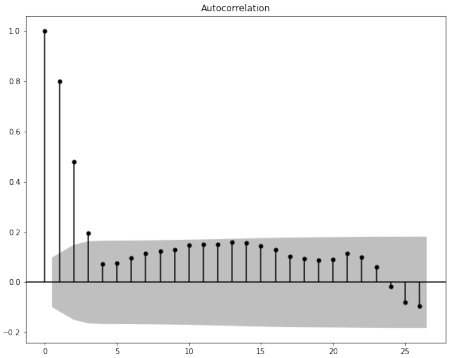


(b) PACF analysis of generated wind speed data on an hourly basis

Figure 4.28: Hourly PACF analyses of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

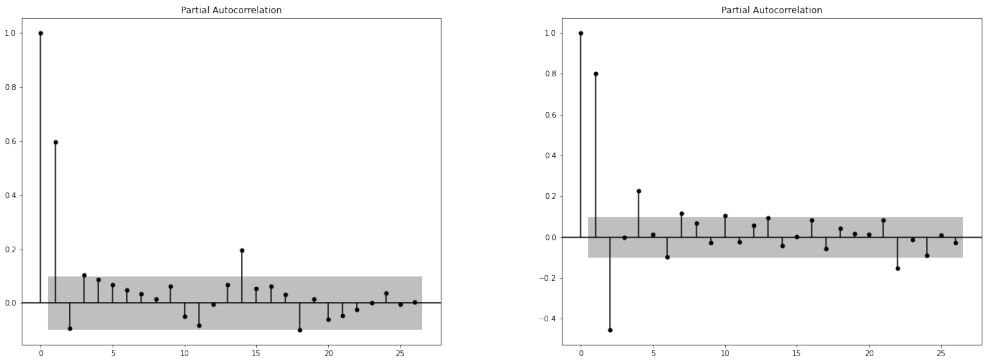


(a) ACF analysis of real wind speed measurements on a daily basis



(b) ACF analysis of generated wind speed data on a daily basis

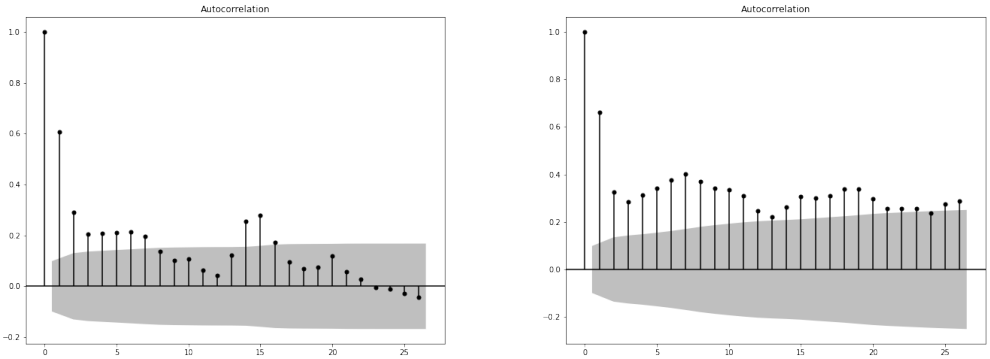
Figure 4.29: Daily ACF analyses of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case



(a) PACF analysis of real wind speed measurements on a daily basis

(b) PACF analysis of generated wind speed data on a daily basis

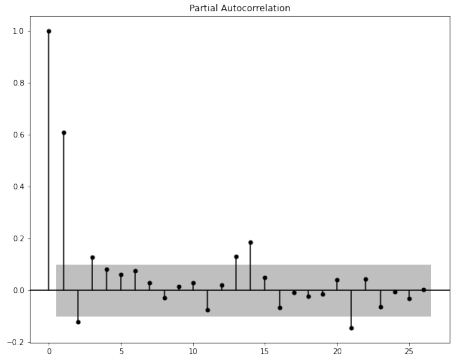
Figure 4.30: Daily PACF analyses of real wind speed measurements of Bandırma Wind Farm and generated wind speed data for this case



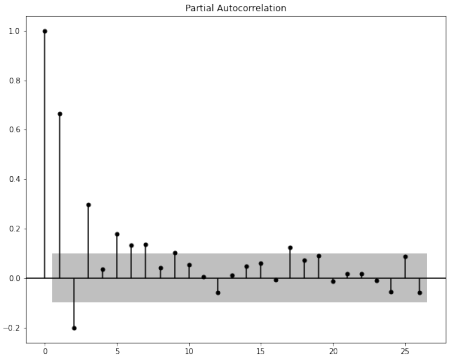
(a) ACF analysis of real wind speed measurements on a daily basis

(b) ACF analysis of generated wind speed data on a daily basis

Figure 4.31: Daily ACF analyses of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case

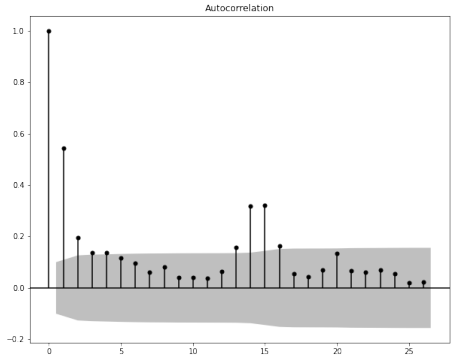


(a) PACF analysis of real wind speed measurements on a daily basis

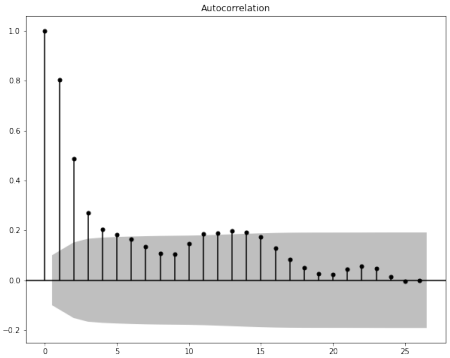


(b) PACF analysis of generated wind speed data on a daily basis

Figure 4.32: Daily PACF analyses of real wind speed measurements of Soma Wind Farm and generated wind speed data for this case

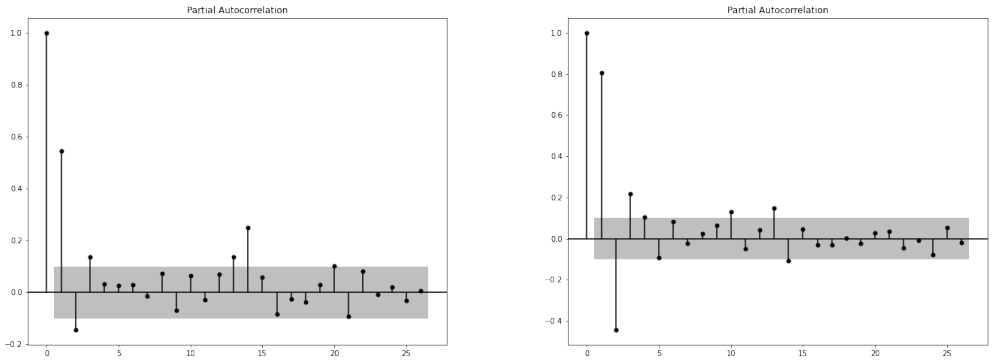


(a) ACF analysis of real wind speed measurements on a daily basis



(b) ACF analysis of generated wind speed data on a daily basis

Figure 4.33: Daily ACF analyses of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case



(a) PACF analysis of real wind speed measurements on a daily basis

(b) PACF analysis of generated wind speed data on a daily basis

Figure 4.34: Daily PACF analyses of real wind speed measurements of Zeytineli Wind Farm and generated wind speed data for this case

According to ACF-PACF results, correlation patterns in original data are also preserved in generated data. At hourly ACF, autocorrelations in generated data decrease systematically through time, and 24 hours lagged data show stronger correlations. Also can be observed at hourly PACF results that after a 1-hour lag, partial-autocorrelations in generated data decrease dramatically like in real data. Also, daily ACF-PACF results of generated data have strong similarities with daily ACF-PACF results of real data. However, the main difference between generated and real data, according to ACF-PACF results, is that generated data has stronger correlations than real data, especially on an hourly basis. Despite the fact that the GAN architecture uses noise to generate stochasticity, the neural networks' output generation process is deterministic. Outputs are generated with the same weights and biases; therefore, the outputs would correlate. GAN's stochasticity was able to suppress these correlations on daily data, but there are still some extra correlations on hourly data. Since these correlation differences are minimal, it has a negligible effect on our outputs. However, in a case where GAN-generated data is processed hourly, these extra correlations should be considered according to the case.

4.6 Production Analysis

We predicted the past productions for the target wind farms with pre-trained models trained with real wind farm data (virtual farm). These predictions are the primary metric of the feasibility study. With these predictions, we aim to analyze the potential productions of the candidate wind farms for the defined period. We used the wind speed data that was generated in previous steps and predicted production on an hourly basis. However, since we aim to analyze long-term productions, we transformed the predictions on a monthly basis with mean values and used monthly predictions to evaluate feasibility analysis. In this way, fluctuations in hourly predictions were smoothened. The structure of the prediction model is described in the Methodology

chapter (Chapter 3).

4.6.1 Prediction Model Training

We determined the LSTM-based prediction model's parameters with the hyper-parameter optimization technique, like in the GAN model parameters determination process. We trained the prediction model with different epochs between 20-300, learning rates between 0.001-1, lag counts between 1-72, and Adam and SGD optimizers and determined the optimum combination of parameters. The hyperparameter optimization process gave similar outputs in all cases, so we used the same training parameters. We used the Adam optimizer with 0.01 learning rate in the prediction model. We trained the model with the last 12 hours values and iterated the training process 120 times. Training losses over epochs graphs are presented in Figures 4.35, 4.36, and 4.37.

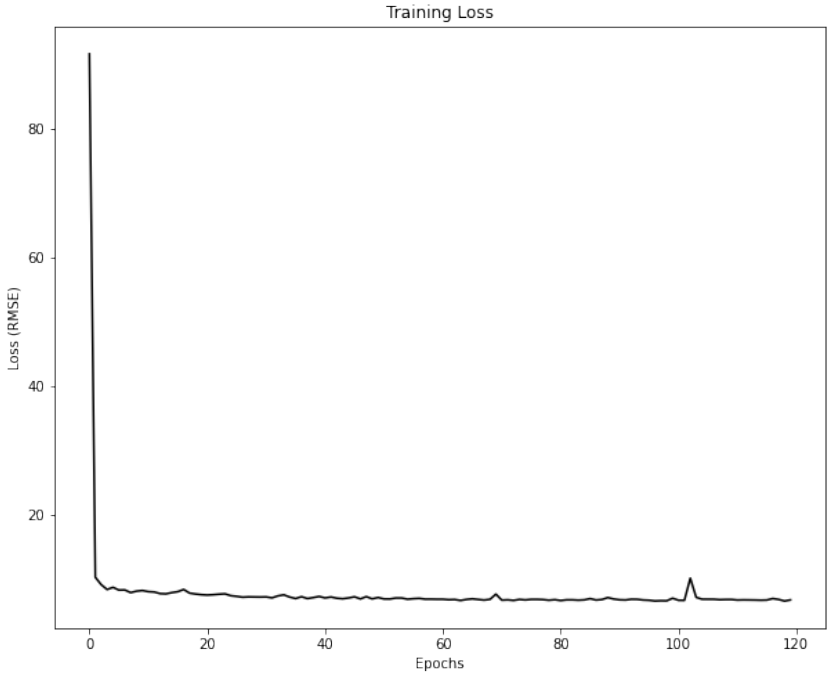


Figure 4.35: Bandırma Wind Farm production prediction model training losses

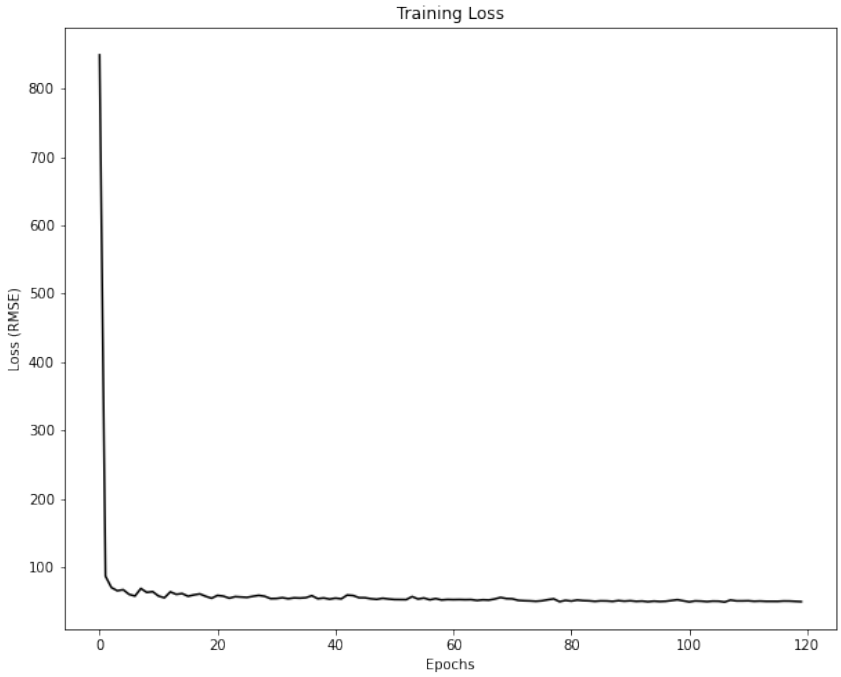


Figure 4.36: Soma Wind Farm production prediction model training losses

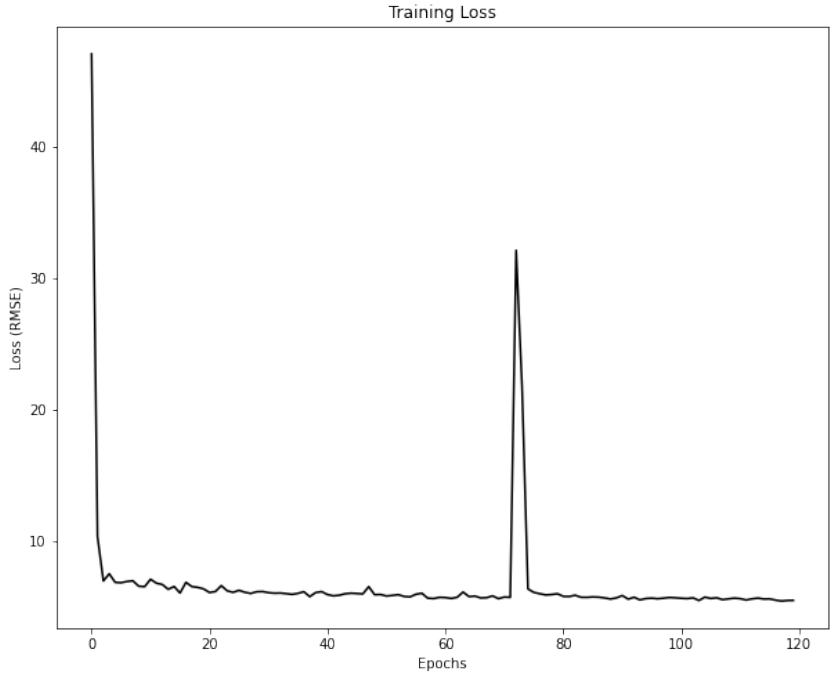


Figure 4.37: Zeytineli Wind Farm production prediction model training losses

4.6.2 Production Prediction

With pre-trained virtual farms, all cases' productions were predicted independently and were compared with real past productions. Plots of real wind power outputs and predicted power outputs are presented in Figures 4.38 and 4.39 for Bandırma case, 4.40 and 4.41 for Soma case, and 4.42 and 4.43 for Zeytineli case. For a better examination of results visually, plots are presented with both all data and partial data separately. All plots were generated on an hourly basis.

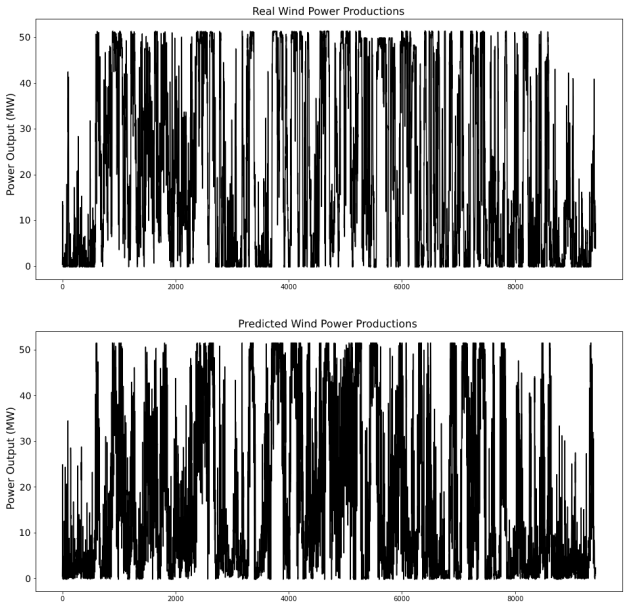


Figure 4.38: Plots of real power outputs of Bandırma Wind Farm and predicted power output for this case

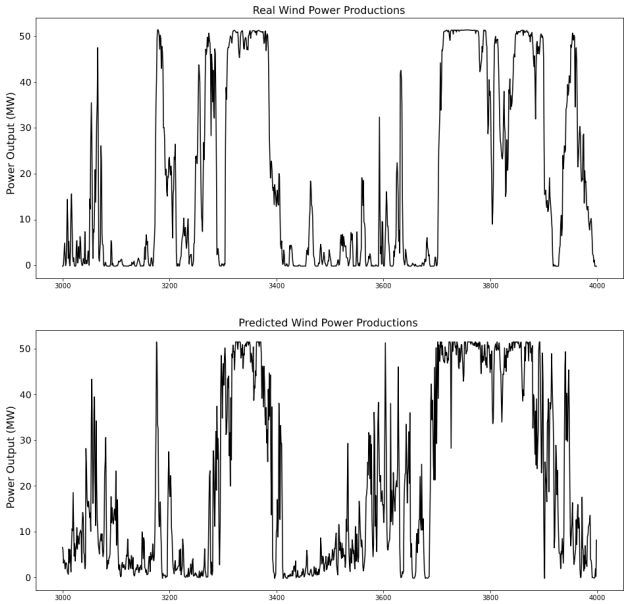


Figure 4.39: Partial plots of real power outputs of Bandırma Wind Farm and predicted power output for this case

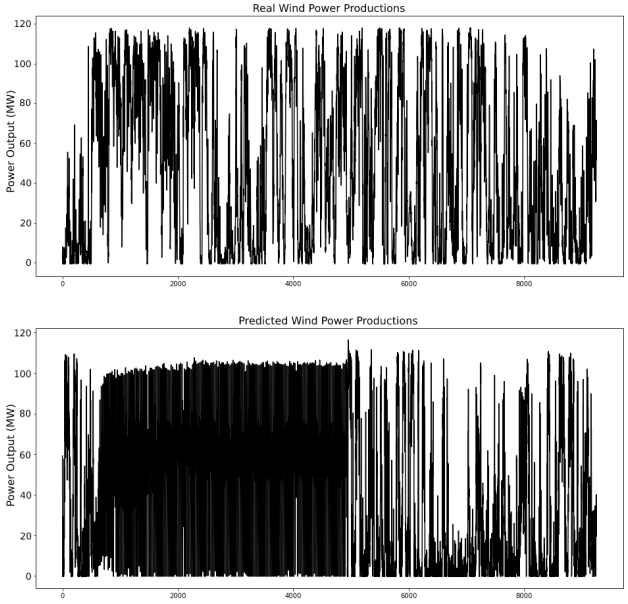


Figure 4.40: Plots of real power outputs of Soma Wind Farm and predicted power output for this case

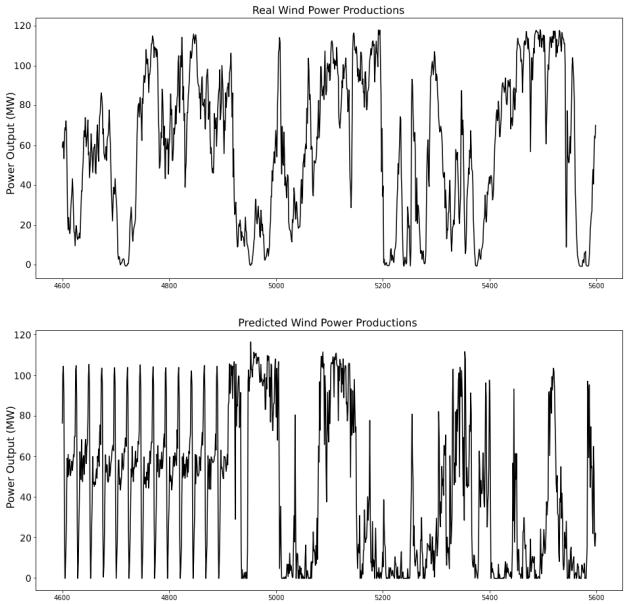


Figure 4.41: Partial plots of real power outputs of Soma Wind Farm and predicted power output for this case

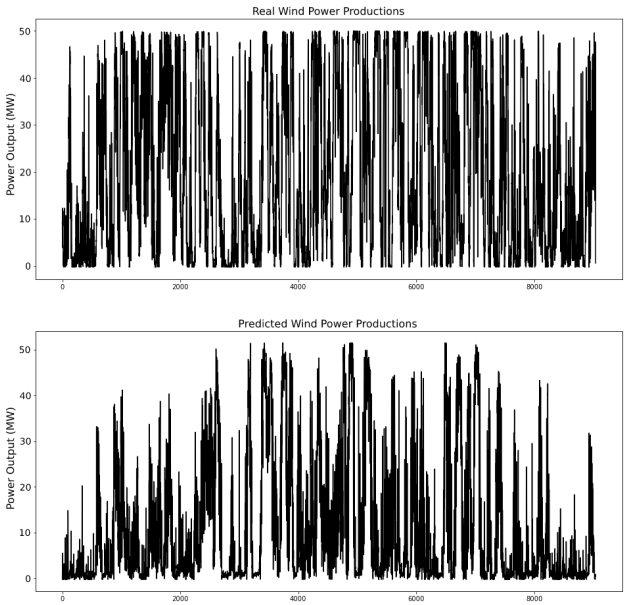


Figure 4.42: Plots of real power outputs of Zeytineli Wind Farm and predicted power output for this case

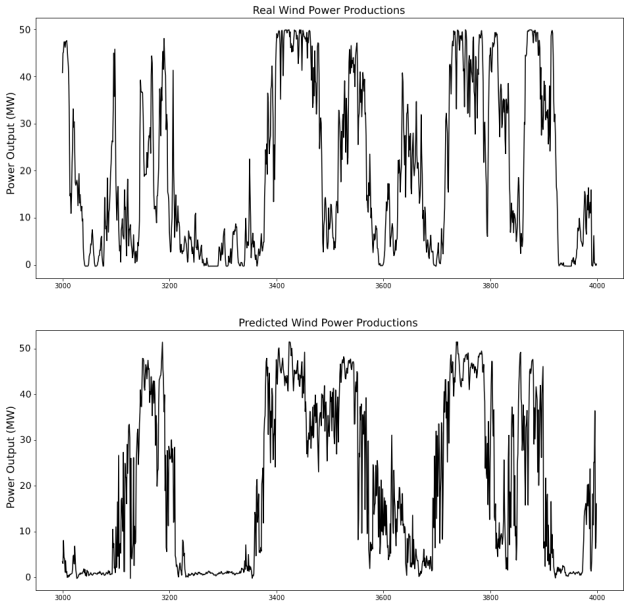


Figure 4.43: Partial plots of real power outputs of Zeytineli Wind Farm and predicted power output for this case

As can be visually observed in the prediction outputs, prediction errors strongly depend on the error in synthetic wind speed data. The power output data trends were successfully predicted using wind maps and virtual farms. Predictions have small fluctuations that real data doesn't have, but since we evaluate predictions with monthly averages, these fluctuations have a negligible effect on the output of the feasibility study.

4.6.3 Evaluation of Predictions

To evaluate production predictions, the RMSE metric was used. All predictions' RMSE scores for different intervals were calculated (Table 4.17), and the method's success was examined. The outputs of this examination give us the insight we need to answer RQ3.

Table 4.17: RMSE scores of predictions for different intervals

Case	Bandırma	Soma	Zeytineli
Capacity (MW)	50	120	50
Average Production (MW)	21.36	49.50	20.13
Hourly RMSE	22.00	50.56	22.25
Daily RMSE	16.96	42.12	19.17
Monthly RMSE	6.39	20.80	10.60
Yearly RMSE	4.40	12.70	9.48
Yearly Percentage Error (%)	20	25	47

According to the Table 4.17, RMSE value is decreasing while evaluation period increase. Mean values of longer periods give better results than shorter periods in all cases. The reason for this is proposed wind map generation method is able to capture trends in input time-series data. Still, it fails to capture the local and temporal dynamics of the target region. Hence, the proposed method may fail if short-term wind speeds or power productions were aimed to analyze. On the other hand, on a yearly period, the proposed model's outputs are reasonable for feasibility analyses. With the advantage of cost and time efficiency, the proposed method seems like a strong alternative for the feasibility analysis of wind farms.

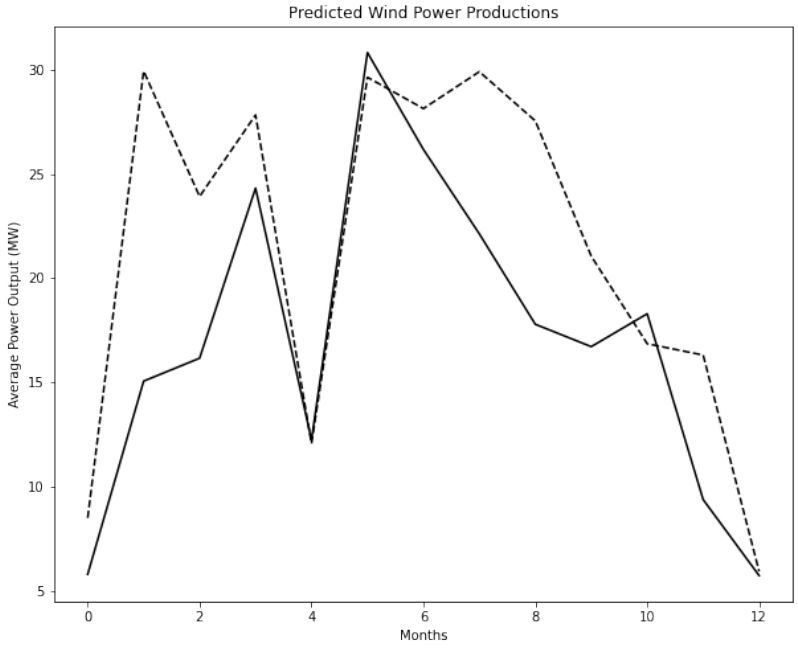


Figure 4.44: Plots of monthly averages of real power outputs of Bandırma Wind Farm and monthly averages of predicted power output for this case

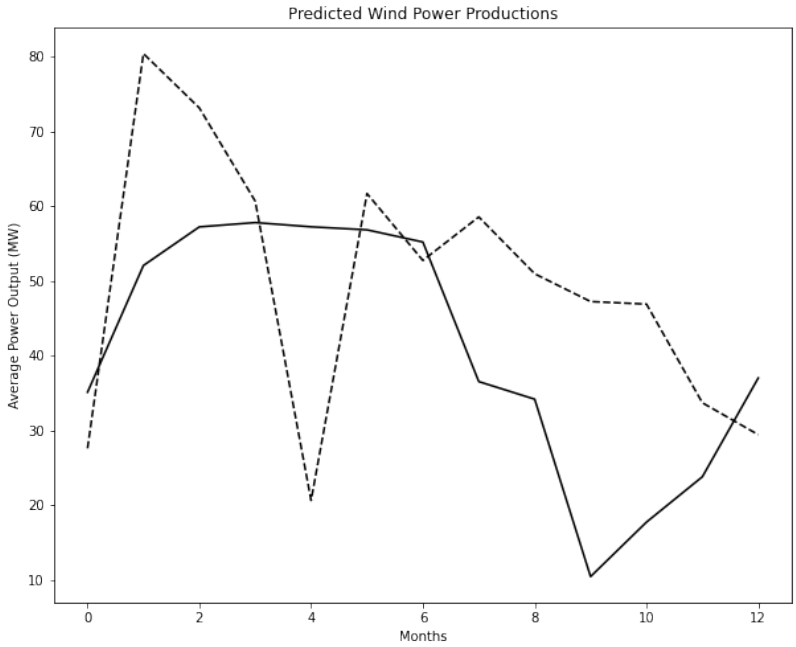


Figure 4.45: Plots of monthly averages of real power outputs of Soma Wind Farm and monthly averages of predicted power output for this case

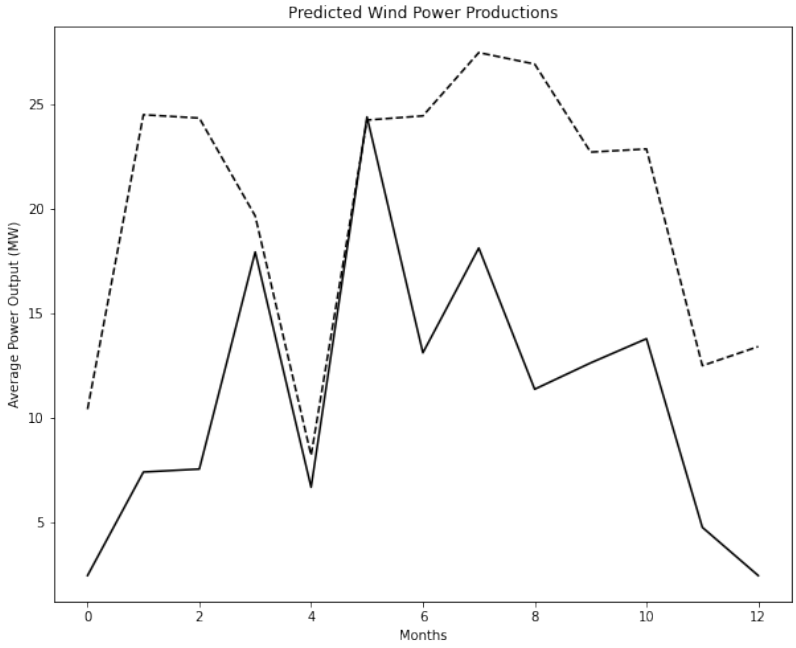


Figure 4.46: Plots of monthly averages of real power outputs of Zeytineli Wind Farm and monthly averages of predicted power output for this case

In Figures 4.44, 4.45, and 4.46, dashed lines show real past productions of wind farm and solid lines show production predictions. According to these figures, the monthly predictions of Bandırma and Zeytineli cases show a similar trend to real data. In contrast, in the Soma case, the predictions' pattern seems irrelevant to the real production pattern. The reason for the irrelevance in the Soma case is that the model failed to find highly correlated neighbors of Soma Wind Farm, located at a high elevation, and stuck to the same data points. With this limited set of neighbors, the prediction model also failed in terms of accuracy. Hence, the Soma case is the most unsuccessful of all experiments. However, this case gives powerful insights into the weaknesses of the neighborhood selection process and how to improve it.

On the other hand, the outputs of the Bandırma and Zeytineli cases are promising. The proposed method successfully predicts real power output series trends in these cases. However, as mentioned in the "Distribution Analysis" section, predicted values are almost constantly smaller than real values. The main reason for this is wind farms are usually located at non-urban sites with high elevations, while most weather stations are close to urban sites and sea levels. Since we trained the GAN model with weather station data, the generator becomes more likely to assume that the target location is an urban site. Hence, it generates smoother wind speed data for the target location. This problem may be easily resolved with more data from wind farms instead weather stations. Also, changing normalization coefficients may increase the accuracy of predictions.

4.7 Discussions

In this chapter, we applied our proposed feasibility analysis method to three different wind farms: Bandırma, Soma, and Zeytineli. The results of experiments are promising, especially when the proposed method's cost and time efficiency advantages are considered. At the end of the process, production predictions of experiment cases for one year have emerged. We predicted Bandırma Wind Farm's (50 MW capacity) monthly productions with 6.39 RMSE score, Soma Wind Farm's (120 MW capacity) monthly productions with 20.80 RMSE score, and Zeytineli Wind Farm's (50 MW capacity) monthly productions with 10.60 RMSE score. Also, we successfully predicted the monthly changes in power outputs in Bandırma and Zeytineli cases.

Our first aim was to generate geospatial wind speed data for a location where no past data was available. For this, we used a GAN model and trained this model with geospatial and time neighborhood information. Evaluation results of generated data show that the generated data preserved the geospatial characteristics of real data. And this gives us the answer to RQ1. ACF-PACF analysis results show that the generated data has similar time-series characteristics to the real data. Hence, we found the answer to RQ2. The evaluation results of predicted productions show the proposed method can be used as an alternative feasibility analysis approach under certain conditions. In two of the three cases, we predicted the past productions of a candidate wind farm with a reasonable error rate. In this way, we found the answer to the RQ3 too. However, the proposed method has shown poorer performance in the Soma case than in other cases. This case has shown us the weaknesses of the proposed method and has given ideas about improvement.

During experiments, we faced some issues that should be resolved in the process. The first issue we faced is the generated wind speed data has lower values than real wind speed values despite trends in synthetic and real wind speed data being matched. We aim to work on two different approaches to resolve this issue. The first approach is collecting more data from wind farms instead weather stations. This way, locations where the data is collected and target locations will have similar regional characteristics; hence generated data would be more suitable for possible wind farm locations. Another approach that we aim to work on is normalizing wind speed values like we do for normalizing neighborhood dimensions. The regional effect on wind speed could be calculated with a regression-based model. This way, wind speed values at wind farm locations that have stronger wind speeds could be downgraded to an urban level where the weather stations are usually located. To conduct this approach, weather domain expertise should also be used.

Another issue we faced during experiments was the lack of variation in the neighbor points in some cases. If the target point is far from input data points in terms of location or time, the proposed model could be stuck with a limited set of neighbors and select these neighbors for all data points. Using a more extensive dataset could also resolve this issue, like in the previously mentioned issue. Also, changing normalization coefficients could resolve this issue too. However, in this way, the model could select data points with lower correlations as neighbors, and prediction accuracy could decrease dramatically.

We conducted the proposed method to analyze the long-term production of candidate wind farms. However, since the generated data fluctuates more than the real data, using this approach for short-term analysis could be insufficient. Time-series smoothing techniques may be used to get more accurate results in shorter periods. Also, if more data from closer points to the target point are available, the proposed method would give more precise results for shorter periods.

CHAPTER 5

CONCLUSIONS

In this work, we propose an alternative way to feasibility analysis of wind farms which is a long and expensive process. We aim to speed up the decision-making process in the pre-installation phase to contribute to increasing wind energy farm installation and efficiency. The proposed method allows multiple locations and different wind farm capacities to be evaluated rapidly and cheaply. The virtual wind farm approach provides flexibility to the method. According to training data used for virtual wind farm training, the virtual wind farm model can represent different wind farm and turbine set characteristics. Also, with the wind map generation with the GAN approach, with a reasonable amount of historical data, realistic wind speed data can be generated for a location where no past data is available. Thus, the necessity of the on-site data collection process can be reduced. Decisions can be made without on-site data or with a smaller amount of on-site data.

Moreover, with this work, we present a novel GAN approach that can generate synthetic geospatial time-series data which is the main contribution of this work. We used this model to augment weather data for locations where no historical data is available. Results show that the proposed approach can generate realistic synthetic data that preserves multidimensional correlations like geospatial and time-series correlations. This model can be used for other different geospatial time-series problems like local market prices or the spread of disease too.

We divided our work into three primary phases: data preprocessing, wind map generation, and long-term prediction of wind energy production. In the first phase, we examined the neighborhood correlations. At first, we normalized all the dimensions defined in different scales and units. After that, we determined a k value as the count of neighbors whose information will be used for prediction. At last, we selected neighbors of all data points and transformed our datasets into more extensive datasets with neighborhood information. In the second phase, we trained our GAN model with these transformed datasets and generated a realistic wind map for the target location. Since our GAN conditions and neighbor points were determined according to time and locational information, generated synthetic data preserved both geospatial and time-series characteristics of original data. With the evaluation of generated data, we validated this argument and found the answers to RQ1 and RQ2; we generated realistic geospatial data for a location with no past data available, and this data showed similar time-series and geospatial characteristics to the original data. In the third phase, we predicted the long-term production of candidate wind farms using synthetic wind speed data and evaluated this data by comparing real past productions. Evaluation results show that under certain conditions, the proposed method achieved

simulating real conditions for a candidate wind farm location and predicted production of the candidate wind farm with reasonable accuracy. That gave us the answer to RQ3; synthetic data can be used for quick feasibility analysis of a wind farm with the proposed method. Physical methods may provide better results for feasibility studies in different conditions, but the proposed method requires fewer data and time. Thus, this method could be a handy tool for governments and renewable energy investors.

In the future, we aim to develop our normalization coefficient determination and neighborhood selection processes to overcome low wind speed generation, lack of variety in neighbor points, and location dependency problems which are mentioned in related sections. And also, by separating maintenance operations and breakdowns forecasts from production forecasts, we aim to increase the accuracy of our model. We believe the proposed method can outperform other feasibility analysis approaches with these additions. Moreover, we aim to test our proposed method on different locations and wind farms. By changing the input dataset, we aim to evaluate our method's outputs with smaller and bigger inputs.

REFERENCES

[1] S. Hanifi, X. Liu, Z. Lin, and S. Lotfian, "A critical review of wind power forecasting methods—past, present and future," *Energies*, vol. 13, no. 15, 2020.

[2] K. Klemmer, A. S. Koshiyama, and S. Flennerhag, "Augmenting correlation structures in spatial data using deep generative models," *ArXiv*, vol. abs/1905.09796, 2019.

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Advances in Neural Information Processing Systems*, vol. 3, 06 2014.

[4] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *ArXiv*, vol. abs/1411.1784, 2014.

[5] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2016.

[6] C. Dewi, R.-C. Chen, Y.-T. Liu, and H. Yu, "Various generative adversarial networks model for synthetic prohibitory sign image generation," *Applied Sciences*, vol. 11, p. 2913, 03 2021.

[7] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 214–223, PMLR, 06–11 Aug 2017.

[8] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2813–2821, 2017.

[9] S. Azadi, C. Olsson, T. Darrell, I. Goodfellow, and A. Odena, "Discriminator rejection sampling," in *International Conference on Learning Representations*, 2019.

[10] L. Tierney, "Markov chains for exploring posterior distributions," *Annals of Statistics*, vol. 22, pp. 1701–1728, 1994.

[11] R. Turner, J. Hung, E. Frank, Y. Saatchi, and J. Yosinski, "Metropolis-Hastings generative adversarial networks," in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 6345–6353, PMLR, 09–15 Jun 2019.

[12] L. Xu and K. Veeramachaneni, "Synthesizing tabular data using generative adversarial networks," *ArXiv*, vol. abs/1811.11264, 2018.

[13] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim, "Data synthesis based on generative adversarial networks," *Proc. VLDB Endow.*, vol. 11, p. 1071–1083, jun 2018.

[14] "TGAN GitHub Repository Page." `https://github.com/sdv-dev/TGAN`. Accessed: 2022-08-18.

[15] "TableGAN GitHub Repository Page." `https://github.com/mahmoodm2/tableGAN`. Accessed: 2022-08-18.

[16] J. Yoon, D. Jarrett, and M. van der Schaar, "Time-series generative adversarial networks," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[17] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[18] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Advances in Neural Information Processing Systems* (D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds.), vol. 29, Curran Associates, Inc., 2016.

[19] M. N. Fekri, A. M. Ghosh, and K. Grolinger, "Generating energy data for machine learning with recurrent generative adversarial networks," *Energies*, vol. 13, no. 1, 2020.

[20] D. Hazra, W. Shafqat, and Y. Byun, "Generating synthetic data to reduce prediction error of energy consumption," *Computers, Materials and Continua*, vol. 70, pp. 3151–3167, 01 2022.

[21] J. Moon, S. Jung, S. Park, and E. Hwang, "Conditional tabular gan-based two-stage data generation scheme for short-term load forecasting," *IEEE Access*, vol. 8, pp. 205327–205339, 01 2020.

[22] A. Puchko, R. Link, B. Hutchinson, B. Kravitz, and A. C. Snyder, "Deepclimgan: A high-resolution climate data generator," *ArXiv*, vol. abs/2011.11705, 2020.

[23] A. Ayala, C. Drazic, B. Hutchinson, B. Kravitz, and C. Tebaldi, "Loosely conditioned emulation of global climate models with generative adversarial networks," *ArXiv*, vol. abs/2105.06386, 2021.

[24] A. N. Wu and F. Biljecki, "Ganmapper: geographical data translation," *International Journal of Geographical Information Science*, vol. 36, pp. 1394 – 1422, 2022.

[25] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of the Atmospheric Sciences*, vol. 20, pp. 130–141, 1963.

[26] Z. Tian, "Chaotic characteristic analysis of short-term wind speed time series with different time scales," *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects*, pp. 1–15, 08 2019.

[27] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, vol. 46, pp. 234–240, 1970.

[28] V. Sohoni, S. Gupta, and R. Nema, "A critical review on wind turbine power curve modelling techniques and their applications in wind based energy systems," *Journal of Energy*, vol. 2016, pp. 1–18, 01 2016.

[29] V. Raj, S. Mathew, and M. Petra, "Artificially intelligent models for the site-specific performance of wind turbines," *International Journal of Energy and Environmental Engineering*, vol. 11, pp. 289–297, 07 2020.

[30] E. Fix and J. L. Hodges, "Discriminatory analysis - nonparametric discrimination: Consistency properties," *International Statistical Review*, vol. 57, p. 238, 1989.

[31] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, p. 1225–1234, JMLR.org, 2016.

[32] "Turkish State Meteorological Service Website." `https://www.mgm.gov.tr/`. Accessed: 2022-08-18.

[33] Turkish Wind Energy Association, "Turkish Wind Energy Statistic Report," tech. rep., Turkish Wind Energy Association, 01 2021.

[34] "EXIST Transparency Platform Website." `https://seffaflik.epias.com.tr/`. Accessed: 2022-08-18.