# POLITECNICO DI TORINO

**Master's Degree in Civil Engineering**


**Master's Degree Thesis**

# Generative Adversarial Networks for Data Augmentation in Structural Health Monitoring

**Supervisors**

**Prof. Giulio VENTURA**

**Prof. Mauro CORRADO**

**PhD Sasan FARHADI**

**Candidate**

**Ten. Mariateresa IAVARONE**

**November 2024**

# Abstract

The present study contributes to the development of an automated system that leverages Deep Convolutional Neural Network-based approaches for the monitoring of prestressing wire breakage in concrete structures subjected to severe aging factors, such as bridges. Advanced methodologies for data acquisition and signal processing within the framework of Structural Health Monitoring (SHM) are explored, focusing on data augmentation techniques to address the critical issue of limited data availability. The work involved conducting controlled destructive tests on two prestressed concrete bridges in L'Aquila, Italy, prior to their planned demolition. A combination of accelerometers and acoustic emission sensors was utilized to capture vibration data during the controlled breakage of prestressing wires. This approach provided essential real-world data, which is crucial for comprehensive analysis. The acquired elastic wave signals were transformed into time-frequency representations via the Short-Time Fourier Transform (STFT), employing various window sizes to find an optimal balance between time and frequency resolution. The resulting spectrograms, normalized for consistency, served as the primary input for training Generative Adversarial Networks (GANs), which were utilized to address the data scarcity. This study compares different GAN architectures, including the Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and Least Squares GAN (LSGAN). Among these, the LSGAN showed superior performance, producing stable and high-quality augmented STFT images. The generation of synthetic datasets plays a central role in enhancing deep learning algorithms to identify structural anomalies, thereby improving predictive maintenance and critical degradation detection capabilities. Through these methodologies, the thesis contributes new perspectives into the application of deep learning for SHM, emphasizing the importance of data augmentation to support more effective and reliable infrastructure monitoring. The findings indicate the robustness of the GANs for augementation of STFT images to enhance SHM, which can lead to an increment in the resilience, safety and longevity of critical civil infrastructure, especially bridges.

# Acknowledgements

*Al Prof. Giulio Ventura,*
*per essere un esempio di vita tenace e risoluto*
*e per aver creduto in me, incoraggiandomi a non arrendermi mai.*

*Al Prof. Mauro Corrado,*
*per essere stato di ispirazione, per la sua passione incrollabile nella vita professionale e interpersonale, che mi hanno spronata a migliorare.*

*Al dottorando Sasan Farhadi,*
*per la tua disponibilità, il tuo aiuto pratico e la tua pazienza infinita.*

*Alla mia mamma, Lucia,*
*il mio pilastro e la mia fonte inesauribile di forza.*

*Alla mia nonna, Rosa,*
*per il suo affetto instancabile e il suo esempio di resilienza e saggezza.*

*A mio fratello, Giuseppe,*
*per essere sempre al mio fianco, con la tua presenza silenziosa ma fondamentale.*

*Al mio fidanzato, Claudio,*
*per il tuo amore incondizionato, il tuo incoraggiamento e la tua presenza, ogni singolo giorno del nostro (spero) lungo viaggio insieme.*

*Al mio incredibile "dream team", Giadi, Mauri e Ale,*
*per essere stati parte imprescindibile per il compimento di questa difficile e lunga conquista, durata due anni,*
*e per aver lottato insieme nella "giungla" della vita.*

*A me stessa,*
*perché ogni giorno è una nuova sfida, abbracciala.*

# Table of Contents

# List of Tables

# List of Figures

# Glossary

**AcousticNet** A hybrid deep learning model developed for analyzing acoustic emissions related to prestressing tendon breakage in concrete bridges

**B-Scan** A type of ultrasonic imaging used in Non-Destructive Evaluation (NDE) that displays a cross-sectional view of the internal structure of an object

**CycleGAN** A type of Generative Adversarial Network designed for unpaired image-to-image translation, enabling style transfer and domain adaptation without paired training examples

**XAI** Explainable Artificial Intelligence; a set of methods and techniques aimed at making the predictions and decisions of AI systems interpretable and understandable to humans, enhancing trust and transparency

**FID** Frechet Inception Distance; a metric used to evaluate the quality of images generated by generative models, such as Generative Adversarial Networks, by comparing the distribution of generated images to the distribution of real images in the feature space of a pre-trained Inception model; lower FID scores indicate better image quality and closer resemblance to real images

**IEPE interface** Integrated Electronics Piezo-Electric interface; a type of signal conditioning that combines the piezoelectric sensor's low output signal with an integrated amplifier, providing a more robust signal for data acquisition systems; it is commonly used in accelerometers and microphones for applications like structural health monitoring (SHM)

**KDE** Kernel Density Estimate; a non-parametric way to estimate the probability density function of a random variable

**Leaf-Bootstrapping** A method used in Generative Adversarial Networks (GANs) to improve training performance by clustering data into smaller subsets before training

**MMD** Maximum Mean Discrepancy; a statistical measure used to compare the distributions of real and generated data, often utilized to evaluate the performance of generative models like GANs

**MEWP** Mobile Elevating Work Platform; a machine used to provide temporary access to inaccessible areas, particularly for construction or maintenance work at height, commonly used in structural monitoring and testing

**MixedAE** A variant of the autoencoder model that incorporates homologous recognition to improve representation learning in self-supervised learning tasks

**NDE** Non-Destructive Evaluation; a set of analysis techniques used to evaluate the properties of a material, component, or system without causing damage, crucial for inspecting structural integrity in SHM applications

**RCGAN** Recurrent Conditional Generative Adversarial Network; a GAN model that incorporates recurrent neural networks to handle sequential data, allowing conditional generation based on input sequences

**RWCP-SSD** Real World Computing Partnership Sound Scene Database; a dataset used for evaluating sound event classification systems

**SIS** Self-Inception Score; a metric used to evaluate the quality of images generated by GANs by comparing them to a reference set of images

**StackGAN** Stacked Generative Adversarial Network; a GAN model that generates high-resolution images in a multi-stage process by progressively refining outputs

**TimeGAN** A generative adversarial network designed for time-series data, combining unsupervised learning with supervised training to preserve temporal dynamics and correlations

**TSGAN** Time Series Generative Adversarial Network; a GAN model specialized in generating synthetic time-series data, capturing temporal patterns effectively

**UAV** Unmanned Aerial Vehicle; an aircraft operated without a human pilot onboard, commonly used for remote monitoring and data collection in various applications, including SHM

# Acronyms

**AE** Acoustic Emission

**AI** Artificial Intelligence

**CGAN** Conditional Adversarial Network

**CIM** Corrupted Image Modeling

**CNN** Convolutional Neural Network

**DA** Data Augmentation

**DCGAN** Deep Convolutional Generative Adversarial Network

**DL** Deep Learning

**GAN** Generative Adversarial Network

**LSGAN** Least Squares Generative Adversarial Network

**MAE** Masked Autoencoder

**ML** Machine Learning

**SHM** Structural Health Monitoring

**SNR** Signal-to-Noise Ratio

**STFT** Short-Time Fourier Transform

**VAE** Variational Autoencoder

**WGAN** Wasserstein Generative Adversarial Network

# Chapter 1

# Introduction

Bridges are a vital part of infrastructure, promoting connectivity and enabling efficient transportation. Ensuring their safety is essential for protecting human lives, underscoring the need for careful inspection and continuous monitoring. Such monitoring is essential for detecting potential structural issues and preventing failures. The scarcity of monitoring data makes it difficult to detect and predict structural anomalies in time. Without a sufficient variety of representative data, machine learning (ML) models may fail to accurately identify emerging issues in infrastructure. This problem is compounded by the rarity of significant structural events, such as wire breaks, which occur infrequently and do not provide enough samples for effective model training. Therefore, there is a critical need for innovative solutions to enrich datasets and enhance the reliability of anomaly detection models. A bridge's structural behavior can be influenced by various factors, including the materials used, its geometric design, traffic load, and environmental conditions. Bridge inspection involves evaluating the structural health of bridges to ensure they can continue to be efficient as intended throughout their lifespan. This process helps in detecting current issues and predicting future changes in their structural state. Automatic monitoring systems, which utilize sensors and artificial intelligence (AI), have become increasingly important for assessing bridge health. Recent research has shown that detecting breakages in prestressing wires in concrete bridges is critical for maintaining safety and preventing catastrophic failures [1, 2]. The use of acoustic emission techniques, combined with ML models, has proven effective in identifying critical structural events, as highlighted by Farhadi et al. (2024) [1].
To address these challenges more effectively, advancements in deep learning (DL) have introduced innovative approaches. Generative deep learning (GDL) algorithms have demonstrated their ability to learn the normal behavior of monitoring systems and assess the structural health of bridges in terms of their mode shapes. These models can be extended to produce synthetic mode shapes under ideal monitoring conditions. Any discrepancies between synthetic and actual mode shapes can

indicate changes in structural health, helping to identify potential damage. Regular monitoring through structural health monitoring (SHM) systems is widely recognized as an essential practice for ensuring the safety and reliability of large infrastructure [3, 4]. Moreover, Farhadi et al. (2024) highlighted the potential of data augmentation (DA) and innovative signal processing techniques to enhance traditional monitoring approaches, making real-time detection of structural anomalies more effective [2].
However, despite these technological advancements, one of the main challenges in SHM is the limited availability of data for training AI-based models. Significant structural events, such as wire breaks or major cracks, occur infrequently, resulting in sparse and imbalanced datasets. This scarcity of data can prevent the development of robust models capable of accurately detecting and predicting anomalies. DA offers a promising solution to this issue by generating synthetic data that mimics real-world events, increasing the volume and diversity of the dataset. By expanding the dataset, AI-based models can better learn the patterns indicative of structural damage, leading to improved reliability and performance in real-world SHM applications. GDL models, particularly Generative Adversarial Networks (GANs), have emerged as effective methods for DA, providing synthetic data that maintains the statistical properties of real data, while addressing the challenges posed by limited data availability.
Recognizing the importance of bridges is crucial, as they have historically contributed to societal development by supporting the expansion of infrastructure and promoting economic growth from ancient times to the present. However, the impacts of bridge failures can be severe, posing serious risks to human lives and overall well-being. Bridges facilitate the movement of people, goods, and information, making their proper maintenance and monitoring essential for ensuring robustness and functionality. Factors such as inadequate design, heavy traffic loads, and adverse weather conditions can compromise the structural integrity of bridges, making them susceptible to damage and failure. It is crucial to note that a bridge's initial design phase defines its theoretical static behavior through mathematical models, but this behavior evolves once the bridge is constructed, as real-world factors like materials, geometry, and environmental conditions come into play.

Building on this context, this thesis explores the potential of GANs and its variants for DA in SHM. By employing GANs to generate synthetic data, it is possible to address the challenges posed by data scarcity and enhance the detection and prediction capabilities of ML models. Specifically, using STFT-based GANs allows for the generation of time-frequency representations of structural signals, expanding the range of events available for predictive analysis. By doing so, this research intends to contribute to safer and more reliable infrastructure monitoring systems, reducing the risk of undetected failures.

To emphasize the critical importance of SHM, it is essential to examine some prominent examples of bridge failures. These cases highlight the severe consequences that can arise from structural deficiencies, including tragic losses of life, injuries, and substantial financial costs for repairs and reconstructions.

## 1.1 Infrastructures Incidents

The importance of this topic becomes more clear through real-world examples. Over the years, numerous bridge collapses have occurred around the world, resulting in tragic loss of life, injuries, and significant financial costs for reconstruction and repairs. Such structural failures not only create immediate risks to the community but also decrease public trust in critical infrastructure. These tragic events emphasize the importance of consistent inspection and timely maintenance to identify and address vulnerabilities before they lead to catastrophic outcomes. The following are notable examples of bridge collapses, both in Italy and internationally, which underscore the urgent need for improved monitoring and maintenance practices.

One notable example is the 1970 collapse of the L'Aquila Bridge in Italy during severe flooding. The extreme conditions led to water stagnation and overflow, causing the embankments and piers to give way, which resulted in the northern span collapsing. Fortunately, there were no casualties, as the bridge was empty at the time.
In 1978, the Great Belt Bridge in Denmark experienced the loss of a vessel. A violent storm forced the sailboat to cross the bridge despite the warning lights. The sailboat was struck by a storm gust, and several containers fell on the bridge's span. The resulting collapse injured several individuals and caused significant property damage.
Another tragic incident occurred in 1978 with the Pont de la Machine in Geneva, where a damaged tunnel caused flooding around the supports, resulting in the collapse that killed 24 people and injured 22 more. Similarly, the Congress Street Bridge in Detroit collapsed in 1982 due to the impact of a cargo ship, though fortunately, there were no injuries. In 1994, the Catuja Bridge's 178 m span in Brazil collapsed during maintenance work. The movable section was opened for passing vessels when metal cables broke with noise. A truck on the bridge fell into the water, but everyone escaped. In 1995, the Philadelphia Cottman Avenue Bridge met a similar fate as the wires broke, then fell on a subway line, killing two and injuring 20.
The 2018 collapse of the Morandi Bridge in Genoa, Italy, is perhaps one of the most devastating examples. Built between 1963 and 1967 using then-innovative

post-tensioned concrete technology, a large section failed, killing 43 people and injuring 600 others (see Figure 1.1). A state commission found that design flaws and heavy vehicle use contributed to its collapse. The bridge had previously experienced maintenance-related problems and passed inspection just 48 hours before the disaster.

**Figure 1.1:** The collapse of the Morandi Bridge in Genoa, Italy, in 2018. This catastrophic event highlights the importance of SHM for critical infrastructure.

In 2017, the La Reale viaduct in the Piedmont region of Italy collapsed suddenly. Investigations revealed that the primary cause of failure was the incorrect injection of wires, which led to the absence of grout inside the sheaths for a significant portion of the deck. This void allowed aggressive environmental conditions to accelerate corrosion, resulting in a fragile failure without any evident warning signs except for subtle white efflorescence. The findings highlighted that conventional inspections may not detect critical internal issues. This event underscores the importance of advanced monitoring and preventive measures to identify underlying structural risks [5] (see Figure 1.2).
Similarly, in 2014, the Petrulla viaduct in the province of Agrigento, Sicily, suffered a partial collapse. The failure was attributed to the breaking of prestressed concrete beams that supported the structure. The incident injured four people, including a pregnant woman. The collapse was an absolute reminder of how deterioration due to environmental exposure and material fatigue can compromise structural integrity if not properly monitored. These cases illustrate the need for continuous, real-time monitoring and robust maintenance strategies (see Figure 1.3).

**Figure 1.2:** The collapse of the La Reale viaduct in Fossano, Italy, highlighting the consequences of wire corrosion and structural deficiencies.

**Figure 1.3:** The partial collapse of the Petrulla viaduct in Agrigento, Italy, due to the failure of prestressed concrete beams.

These examples, including the collapses of the La Reale and Petrulla viaducts, emphasize the potential dangers of insufficient structural monitoring and the importance of adopting sophisticated monitoring technologies to ensure infrastructure safety. Learning from these failures, it becomes clear that proactive and continuous monitoring is crucial to prevent similar incidents in the future.

## 1.2 The Structural Health Monitoring and its Challenges

Since the construction of the first bridges, the safety and stability of bridge structures have been extremely important. Today, the increasing number of vehicles and weights crossing bridges, along with climate change, affect their integrity and stability more than ever. As bridges age and deteriorate over time, the risk of instability or vulnerability increases, making regular monitoring essential. This need is further amplified by the growing number of bridges and stricter regulations, necessitating a systematic approach to bridge inspection [6].

Traditional bridge inspections evaluate factors like safety, water resistance, stability, and operational attributes to determine if repairs are necessary. However, this visual approach has limitations: it's subjective due to inspector interpretation, some parts of the structure may be difficult to access, it consumes a lot of time, and often lacks detailed reporting. Additionally, bridges are continuously exposed to environmental conditions like wind, temperature fluctuations, ice, and traffic loads, which increase the risk of deterioration. Insufficient monitoring may ultimately lead to bridge collapse.

To improve efficiency, automated systems have been developed. These systems collect data using cameras and sensors, performing post-processing for damage detection. For example, video inspection uses sequences of images to monitor predefined points of interest. However, these methods still require a qualified inspector to visually assess the bridge structure almost immediately after data acquisition to verify detected defects. As a result, SHM has become an essential practice to ensure the safety and longevity of critical infrastructure like bridges. SHM involves continuously collecting and analyzing data from various sensors installed on the structure, such as as accelerometers, strain gauges, and acoustic emission detectors. The data collected play a key role in identifying potential structural issues before they escalate into serious problems. However, these systems still face challenges. Common type of damage, such as cracks, steel wire cuts, corrosion and material fatigue, can weaken a bridge's structural integrity over time if not detected early. The limitations of traditional visual inspections have led to increased interest in automated systems that can continuously monitor bridges and provide real-time feedback on their condition.

Automated SHM systems offer many advantages over traditional manual inspections. They can continuously collect data from various sensors, allowing real-time monitoring of a bridge's condition. They measure factors like vibrations and temperature

changes, offering a detailed view of the bridge's structural health. Despite these benefits, automated SHM systems still face several challenges, especially in terms of collecting reliable and comprehensive data. Environmental factors, equipment limitations, and the infrequent occurrence of significant structural damage can make it difficult to collect enough data for in-depth analysis. DL methods have become essential for processing the large amount of data collected by sensors in SHM. In particular, advanced sound event classification techniques, which utilize DL models, enhance the detection and interpretation of acoustic emissions identify signal potential structural changes or failures. These methods are crucial for accurately identifying anomalies in real-world SHM applications, where background noise could often interfere with signal analysis. Especially, acoustic emission (AE) signals are essential for detecting structural failures. Zhu et al. [7] conducted a detailed review of DL-based techniques for processing AE signals, demonstrating their effectiveness in analyzing data in real-time. Additionally, Abdeljaber et al. [8] applied one-dimensional convolutional neural networks (1D CNNs) to detect damage based on vibration data, further proving the potential of deep learning for real-time SHM applications. These methods provide immediate insights, allowing for early detection and prediction of structural issues before they become critical.

In the context of SHM, damage identification plays a critical role in ensuring the safety and integrity of large structures such as cable-stayed bridges. Ni et al. [9] demonstrated the importance of operational modal analysis for real-time damage detection in bridges, emphasizing the need for continuous monitoring to prevent catastrophic failures. Similarly, Sohn et al. [4] reviewed a wide range of SHM methodologies, identifying both the challenges and advancements in the field over the years. ML has introduced significant advancements in the field of SHM, providing automated methods for detecting and predicting damage. However, challenges remain, particularly in terms of data scarcity and model interpretability. Worden et al. [10] explored these opportunities and challenges, offering insights into the potential of ML in SHM while also acknowledging the hurdles that must be overcome. Wang et al. [11] proposed methods for large-scale environmental sound classification using dimensionality reduction techniques. These methods can be adapted to SHM to reduce computational burden while preserving critical information for accurate structural event classification.

Therefore, one of the main challenges in SHM remains the limited amount of data available for analysis. This lack of data makes it difficult to train ML and DL models, which typically rely on large datasets for accurate predictions. In this context, DA techniques offer a promising solution to increase both the volume and diversity of available data.

## 1.3 Generative Deep Learning and its application in Structural Health Monitoring

Generative deep learning (GDL) has emerged as a powerful solution to address the challenge of limited data in SHM. Models such as GANs, introduced by Goodfellow et al. [12], consist of two neural networks: a generator that creates data and a discriminator that evaluates its authenticity. These models work in an adversarial manner, where the generator tries to fool the discriminator while the discriminator aims to detect the fake data. This process allows GANs to generate increasingly realistic data over time. In SHM, GANs can produce synthetic data that mimics real structural event data, helping ML models detect anomalies more effectively, even when limited data is available.

Prior to the application of GDL, automated bridge inspection systems had begun incorporating digital technologies, primarily in image processing and sensor data acquisition. These systems evolved gradually, with each advancement improving specific aspects of inspection but often still requiring substantial manual intervention. Early bridge inspection methods primarily relied on image processing, allowing structural engineers to collect visual data of bridge components. However, these methods necessitated extensive manual post-processing to identify structural issues, imposing a significant workload on engineers.
With the integration of traditional DL methods, semi-automatic inspection systems emerged. These systems used neural networks trained on annotated images to assist in detecting bridge defects, reducing the amount of manual work required. By automating part of the classification process, engineers could more efficiently identify potential structural issues based on image data, although manual validation was often still necessary.
Recent advances in DL have enabled the development of fully automatic bridge inspection systems. These systems incorporate advanced algorithms that enhance image quality by filtering out noise and highlighting relevant features, such as the depth and severity of defects. Such advancements allow for faster and more accurate decision-making compared to previous methods, minimizing the need for human intervention and supporting more reliable bridge monitoring processes.

While these early systems produced satisfactory results, they heavily relied on basic image processing techniques that often required extensive manual intervention. The introduction of GDL allowed automated systems to analyze visual data more comprehensively. For instance, GDL models can enhance image resolution, reduce noise, and even generate synthetic data that mimics real-world bridge conditions,

making the model be able "learn" from more varied examples. This allows automated systems to extract and analyze key parameters, such as the severity and type of structural defects, with a higher degree of accuracy and consistency. By training on both real and synthetic data, GDL-based systems can better identify critical issues in bridge structures. As a result, these advanced systems can now process large datasets, such as visual images captured during inspections, more efficiently and accurately, reducing the time required for post-inspection analysis.

Additionally, other techniques like masked autoencoders (MAE), introduced by He et al. (2021) [13], have demonstrated their effectiveness in learning representations from incomplete data. This approach allows models to perform well in various downstream tasks by reconstructing missing parts of input data, which is particularly useful in SHM where sensor data can be noisy or incomplete. Self-supervised learning techniques, such as BEiT, introduced by Bao et al. (2021) [14], have also contributed significantly to image modeling by predicting visual tokens from large, unlabeled datasets. This allows models to generalize better and perform more effectively in real-world SHM scenarios, where labeled data can be scarce or unavailable.

Other generative models, such as Variational Autoencoders (VAEs), introduced by Kingma and Welling [15], and methods like Corrupted Image Modeling (CIM) [16], have also shown potential in SHM. These models generate synthetic data by learning the underlying distribution of the data, which makes them useful for augmenting datasets. For example, CycleGAN, developed by Zhu et al. [17], is an unpaired image-to-image translation model that is especially useful when paired examples are scarce, a common issue in SHM.

## 1.4 This research: Generative Adversarial Networks for Data Augmentation in Structural Health Monitoring

Building upon the discussed applications of GANs in SHM, this thesis focuses on applying Short-Time Fourier Transform (STFT)-based GANs to augment structural event data collected from bridge monitoring systems. By transforming time-series sensor data into a frequency-time representation using STFT, which effectively captures both temporal and spectral features, GANs are utilized to generate synthetic data. This synthetic data enhances the robustness of SHM datasets and supports more effective statistical analysis. STFT-based GANs are designed to capture the essential characteristics of structural event data, which are crucial for detecting and predicting anomalies in bridge structures.

In this research, Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN) and Least Squares GAN (LSGAN) are specifically employed to generate synthetic STFT representations of structural events. The choice of these GAN architectures allows for the exploration of different training dynamics and loss functions, providing a comprehensive understanding of their effectiveness in SHM applications. DCGANs utilize deep convolutional layers to generate high-quality images, making them suitable for capturing the intricate patterns in STFT spectrograms. WGANs address issues like mode collapse and training instability by minimizing the Wasserstein distance between real and generated data distributions. LSGANs, on the other hand, use least squares loss functions to stabilize training and improve the quality of generated data. While statistical tests are applied to better understand the original signals, the primary focus of this research is not on performing detailed statistical analysis. Instead, the investigation centers on how STFT-based GANs can enhance the detection and prediction of structural anomalies in bridges. The ultimate goal is to evaluate the potential of using synthetic data to improve SHM processes for monitoring the health and safety of bridges, thereby contributing to the development of more reliable and effective infrastructure monitoring systems.

# Chapter 2

# Background

As infrastructure ages, the continuous monitoring and detection of damage become crucial to ensure safety, prevent catastrophic failures, and reduce maintenance costs. Structural health monitoring (SHM) has become a fundamental approach in this field, utilizing advanced sensors, data acquisition systems, and computational techniques to assess structural integrity. The integration of signal processing and deep learning (DL) techniques further enhances SHM by improving system efficiency and accuracy. This chapter provides a comprehensive overview of SHM as well as signal processing and DL methods, examining their applications in structural monitoring and damage detection systems.

## 2.1 Structural Health Monitoring Background

The main objectives of SHM include early detecting damage at an early stage, predicting its progression, and enabling timely interventions to prevent catastrophic failures. These goals are essential for maintaining structural integrity and ensuring safety over the lifespan of a structure. Traditional SHM systems rely on sensor networks distributed across the structure, collecting data that is subsequently processed and analyzed to assess the structure's health.

### 2.1.1 Role of Artificial Intelligence in Structural Health Monitoring

Building upon traditional methods, the integration of artificial intelligence (AI) into SHM systems has significantly advanced the field, enabling the automation of damage detection and prediction. AI algorithms can process large amounts of data, making sense of complex patterns that are difficult to detect with traditional methods. As Zinno et al. highlighted [18], AI techniques, such as machine learning

(ML) and DL, have greatly improved the ability of SHM systems to provide accurate diagnostics and real-time monitoring for bridges and other infrastructure.
AI-driven solutions enable automatic processing of sensor data, identifying patterns and detecting anomalies indicative of structural damage. For instance, Gao et al. [19] demonstrated how Generative Adversarial Networks (GANs) could be used to generate structural images for DA, significantly improving the training of AI models in SHM. In particular, AI-driven solutions allow data from sensors to be processed automatically, identifying patterns and detecting anomalies indicative of structural damage. For example, Gao et al. [19] demonstrated how GANs could be used to generate structural images for DA, vastly improving the training of AI models used in SHM (see Figure 2.1).

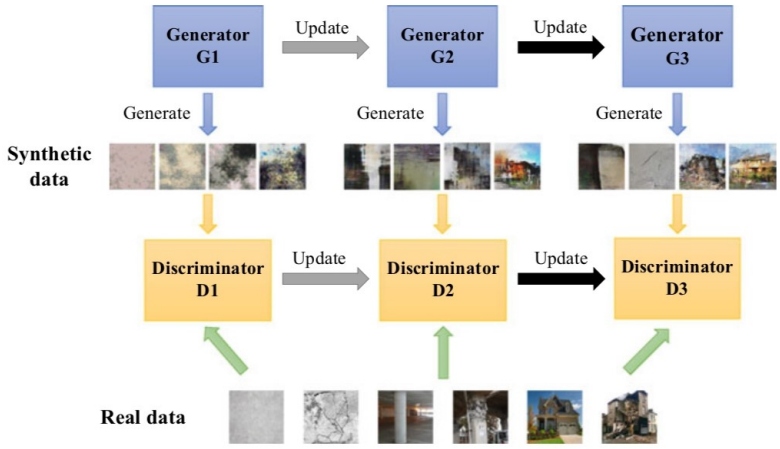



**Figure 2.1:** GAN architecture used by Gao et al. (2019) for generating synthetic structural images. Adopted from [19].

Additionally, the use of big data analytics has further increased the capabilities of SHM systems by processing vast amounts of sensor data. Big data techniques enable the identification of patterns and trends that may indicate early stages of structural damage. Integrating AI techniques, such as DL, into SHM systems allows for more accurate and automated damage detection. These techniques can automatically classify and predict failures by analyzing historical data, sensor readings, and environmental factors. Farhadi et al. [2] demonstrated the effectiveness of automated event-based SHM systems in early damage detection by tracking prestressing tendon breaks in concrete bridges using acoustic monitoring.
Farhadi et al. [2] proposed an innovative approach to SHM, focusing on the detection and classification of acoustic emissions (AE) related to prestressing tendon breakage in concrete bridges. The study introduced AcousticNet, a hybrid

convolutional neural network (CNN) designed to analyze and classify AE signals. AcousticNet's architecture combines multiple CNN layers to extract features from raw AE data. The model was trained on a dataset of both simulated and real-world AE signals, achieving high accuracy in distinguishing between normal and anomalous events. The development and testing of AcousticNet included several key stages:

- **Data collection and pre-processing**: AE signals were collected from laboratory experiments and in-situ bridge monitoring, then preprocessed to enhance feature extraction for AcousticNet.
- **Hybrid CNN model**: AcousticNet integrated DL techniques with domain-specific knowledge to effectively process and classify AE signals.
- **Validation and testing**: The model was validated using a robust testing framework, demonstrating its ability to accurately identify prestressing tendon breakage.

The study by Farhadi et al. [2] highlights the potential of DL in advancing SHM techniques. However, despite these advancements, several significant challenges remain, particularly concerning data quality, sensor placement and the limited availability of high-quality data for training models.

### 2.1.2 Challenges in Structural Health Monitoring

Despite significant advancements, SHM systems face several technical and logistical challenges that impact their effectiveness. One major obstacle, as highlighted by Yu et al. [20], is the availability of high-quality data, especially in complex environments like tunnels or densely populated urban areas. In such conditions, sensor data can be sparse or distorted by environmental noise, making it difficult to detect early signs of structural degradation.
Another critical challenge in SHM is optimizing sensor placement. The efficacy of SHM systems depends heavily on strategic sensor deployment to gather representative structural data. Poor sensor placement can lead to incomplete datasets, limiting the system's ability to capture a full picture of structural health. Kaleb Smith and Anthony O. Smith's work on TSGAN (Time Series GAN) [21] demonstrates how synthetic data can mitigate some of these issues, even though effective sensor placement remains vital for high-quality data acquisition.
The rarity of certain structural events, such as crack initiation or sudden load-induced failures, presents another challenge for SHM. Studies by Worden et al. [10] and Ko and Ni [3] highlight the need for extensive datasets to train robust models for anomaly detection and prediction. However, the infrequency of these events means that SHM datasets are often sparse, which can limit the robustness

of models in detecting less common structural anomalies.
Additionally, environmental factors complicate SHM data collection. Variations in temperature, humidity, and other environmental conditions can introduce noise into sensor readings, obscuring the signals that indicate structural damage. Abdeljaber et al. [8] showed that one-dimensional convolutional neural networks could filter out noise in vibration data, but environmental interference remains a challenge for achieving consistent SHM accuracy.
Finally, computational demands can restrict real-time SHM applications, particularly in systems utilizing advanced ML algorithms. Although approaches like those discussed by Mosalam and Gao [22] incorporate GAN-based data augmentation (DA) to improve SHM accuracy, the computational requirements can hinder practical implementation, especially in resource-limited settings.

In summary, the highlighted challenges in SHM underscore the need for more resilient and efficient systems capable of providing reliable monitoring across diverse environments.

### 2.1.3 Structural Health Monitoring for Bridges

Bridges are crucial infrastructure components, yet they face constant exposure to stresses such as heavy traffic, temperature fluctuations, and environmental degradation. Traditional SHM methods, heavily depending on periodic manual inspections, often fail to detect early signs of damage, particularly in hard-to-access areas where structural vulnerabilities may go unnoticed. Tragic incidents, including the collapse of Italy's La Reale viaduct, underscore the importance of adopting advanced SHM technologies. Ferro et al. (2022) [5] describe how inadequate grout injection in cable sheaths led to undetected corrosion, resulting in the viaduct's collapse. In the case of the Fossano bridge collapse, discussed by Bazzucchi and Ferro (2019) [23], improper maintenance of post-tensioned tendons and lack of grout led to accelerated corrosion, highlighting the need for enhanced SHM strategies that can detect such invisible vulnerabilities before they lead to catastrophic failures. Both cases reveal critical gaps in inspection methods, as visible signs alone were insufficient to predict the underlying structural degradation. These studies highlight the need for SHM systems that provide real-time, data-driven assessments capable of identifying subtle indicators of structural deterioration before severe damage occurs.

Traditional SHM methods for bridges have relied heavily on periodic manual inspections and maintenance, which can be time-consuming and less effective in detecting early-stage damage. Recent advancements in AI technologies have begun to transform bridge monitoring processes, enabling more comprehensive and continuous assessments. Tragic incidents, such as the collapses of the I-35W bridge in

Minneapolis and the Morandi bridge in Genoa, have underscored the importance of adopting advanced SHM technologies to prevent similar catastrophes [18]. AI-based SHM systems are now capable of identifying early signs of structural damage that may not be detectable through manual inspections, thereby allowing for proactive maintenance interventions. For example, Mosalam and Gao [22] proposed the use of GANs for structural image DA in SHM, which enhances model training accuracy even with limited data availability. Such AI-driven advancements demonstrate the potential for SHM systems to become more reliable and scalable, ultimately improving the safety and resilience of bridge infrastructure.

## 2.2 Signal Processing

Signal processing is essential for analyzing and interpreting data collected from sensors in SHM systems. Data from accelerometers, strain gauges, and acoustic emission sensors often contains noise, requiring specialized techniques to filter and enhance the information. These methods help engineers convert raw signals into valuable insights, identify patterns, and detect anomalies that may indicate structural damage.

### 2.2.1 Traditional Signal Processing Techniques

Traditional signal processing methods, including Fourier transforms, wavelet analysis, and filtering, play a fundamental role in SHM, as noted by Gul and Catbas (2009) in their examination of time-series modeling for structural monitoring [6]. These techniques break down complex signals into their frequency component, making it easier to identify changes that may indicate structural degradation. For instance, Fourier analysis is frequently applied to vibration signals from bridges or buildings to gain insights into their dynamic behavior.

Another powerful method is wavelet analysis, which is especially useful for analyzing non-stationary signals. Wavelet transforms allow the localization of damage by detecting changes in both time and frequency domains. Kaleb Smith and Anthony O. Smith [21] highlighted the importance of these techniques in processing acoustic emissions, where time-frequency domain analysis is crucial for detecting microcracks in concrete structures.

### 2.2.2 Advanced Signal Processing Techniques

Advanced signal processing techniques can be utilized to improve the learning procedure in AI-based models. By using complex data patterns, these techniques enhance the detection and classification of structural anomalies. One of the most

notable developments in this field is the use of DA techniques, especially GANs, which can produce synthetic time-series data that closely mimics real-world signals, allowing more robust ML models to be trained.

Guillermo Iglesias and Edgar Talavera [24] provided a comprehensive review of DA techniques for time-series data, including their applications in SHM. Figure 2.2 provides a taxonomy of these DA methods, ranging from traditional approaches such as slicing, jittering, and scaling to more advanced generative models like Variational Autoencoders (VAEs) and GANs. These techniques, which manipulate existing data or generate new samples, are essential for improving SHM accuracy, particularly when training datasets are limited. A key application of GANs in SHM is generating synthetic vibration signals or acoustic emissions to enhance small datasets. By augmenting these datasets, signal processing algorithms can detect rare events, such as crack initiation or material fatigue, more effectively. The findings of Iglesias and Talavera [24] underscore the need for more advanced DA methods designed specifically for time-series data, particularly in areas where datasets are limited or vulnerable. These findings will inform the development of STFT-based GANs in this research, aimed at strengthening the analysis and augmentation of structural event data.

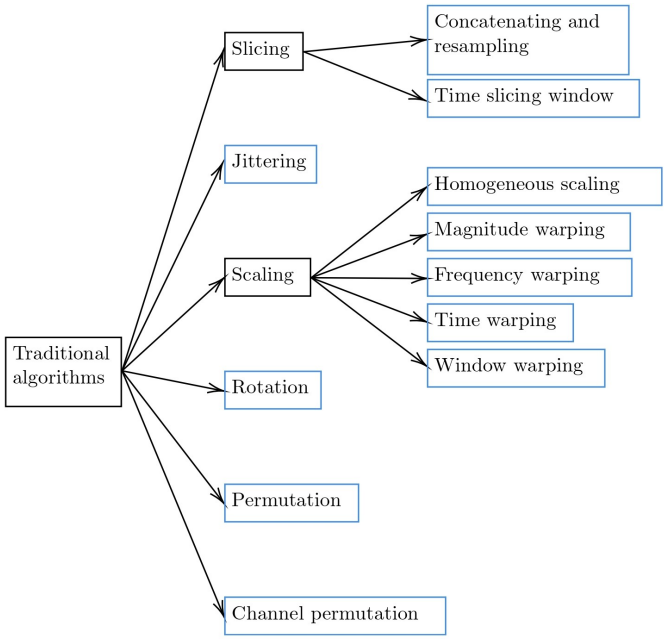



**Figure 2.2:** Taxonomy of Data Augmentation techniques as described by Iglesias and Talavera. Adopted from [24].

Another significant development is the Time Series GAN (TSGAN), introduced

by Smith et al. [21], which uses a conditional GAN architecture to capture complex temporal patterns in time-series data. TSGAN addresses the common SHM challenge of limited training samples by generating high-quality time-series data, making it particularly relevant for modeling rare structural events.
As illustrated in Figure 2.3, TSGAN employs a two-stage GAN architecturein which the first GAN generates synthetic spectrograms from random noise. The second stage involves a conditional GAN that produces realistic time-series data based on these spectrograms. TSGAN incorporates several innovative features. First, it uses Wasserstein GANs (WGANs) to mitigate mode collapse by minimizing the Wasserstein distance between real and generated data distributions. Additionally, it applies conditional GANs (CGANs) to improve the fidelity and diversity of the generated time-series data. Finally, TSGAN supports few-shot learning, making it effective even when training data is limited—a common scenario in SHM applications.

The TSGAN model shows considerable promise for SHM, as it can enhance detection and prediction capabilities by enriching datasets with realistic synthetic data. Future research may explore integrating TSGAN-generated data with other advanced ML techniques to improve the robustness and accuracy of SHM systems [21].

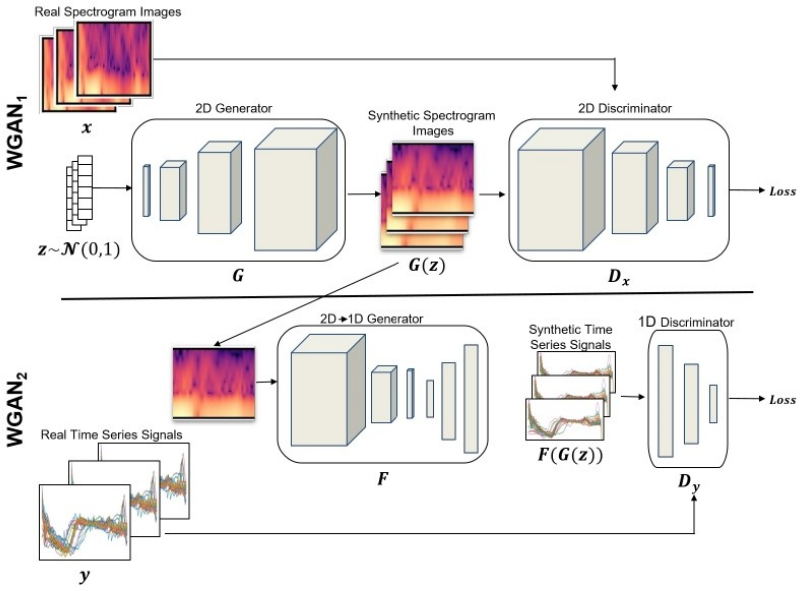



**Figure 2.3:** TSGAN architecture used for generating synthetic time series data. The first WGAN generates synthetic spectrograms, and the second conditional WGAN produces realistic time series data. Adopted from [21].

In addition to TSGAN, Cycle-Consistent GANs (CycleGAN), introduced by Zhu et al. [25], tackle another key challenge in SHM: the lack of paired datasets. CycleGAN enables data transformation between domains without needing paired examples, making it useful for translating data such as acoustic signals into visual spectrograms or other formats suitable for SHM analysis. This cross-modal translation expands DA options, particularly when labeled data is limited.

As illustrated in Figure 2.4, CycleGAN contains two mapping functions ($G : X \to Y$ and $F : Y \to X$) and two adversarial discriminators ($D_Y$ and $D_X$). The cycle consistency loss ensures that translating from one domain to the other and back again yields the original input image. This cycle-consistency feature allows CycleGAN to maintain meaningful transformations, which is particularly relevant in SHM applications where paired data is often unavailable. By simulating structural damage through generated images, CycleGAN contributes to creating more comprehensive datasets for SHM model training.

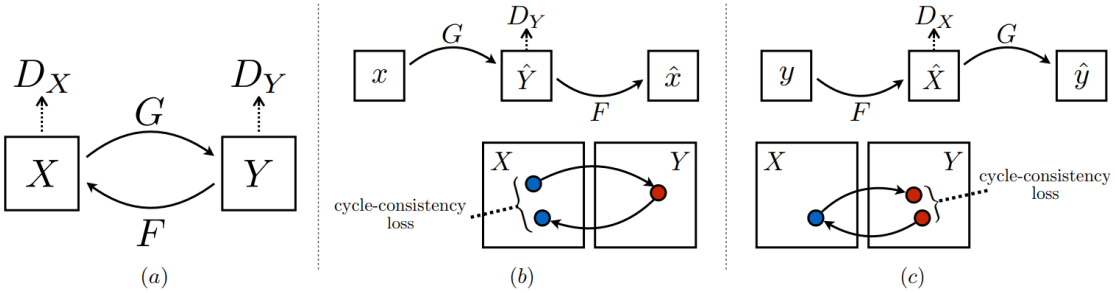



**Figure 2.4:** CycleGAN architecture: The model contains two mapping functions, $G : X \to Y$ and $F : Y \to X$, along with the adversarial discriminators $D_Y$ and $D_X$. Adopted from [25].

Johnson et al. [26] introduced perceptual losses for image generation tasks such as style transfer, which enhance the quality and realism of generated images by focusing on high-level feature representations rather than pixel-by-pixel differences. This approach allows the generated images to retain structural coherence and perceptual fidelity, making them more visually realistic. Such techniques could significantly enhance the visual quality of augmented SHM datasets, especially in applications where the realistic appearance of damage patterns or material textures is crucial for accurate model training and validation.

Furthermore, advanced feature extraction methods, like those proposed by McLoughlin et al. [27], leverage deep neural networks to classify complex environmental sounds, which holds promise for SHM where unstructured acoustic signals often

indicate structural changes. By integrating these methods, SHM systems can better capture subtle variations in acoustic emissions, enabling the detection of anomalies that might signify early damage or structural wear. These techniques provide a more nuanced understanding of sound patterns, enhancing the reliability of SHM systems in real-world monitoring scenarios.

### 2.2.3 Advanced Sound Event Classification

Sound event classification is essential for monitoring and analyzing acoustic emissions within SHM systems, especially in challenging noise environments. Liu et al. [28] introduced a novel classification approach that emphasizes robustness and efficiency, transforming traditional spectrograms into a frequency-energy diagram to highlight energy distribution across frequency bins independently of time. This transformation helps reduce variability due to sound duration, thereby enhancing classification accuracy.

The paper by [28] also presents a two-stage data dimension reduction process:

- **Stage 1**: Importance screening retains only the most significant energy bins, reducing data size while preserving critical information.
- **Stage 2**: Bicubic interpolation further reduces the dimension of feature vectors, maintaining computational efficiency.

This method has proven effective in noisy conditions, as demonstrated by evaluations on datasets like RWCP-SSD, UrbanSound8K, and ESC-50, where it achieves strong performance even under low signal-to-noise ratios (SNRs). Such robustness is promising for real-world applications, where acoustic events are often obscured by background noise.

### 2.2.4 Applications of Signal Processing in Structural Health Monitoring

Signal processing techniques are critical in a variety of SHM applications, particularly in the monitoring of bridges, tunnels, and other critical infrastructures. For example, acoustic emissions, which are generated by the rapid release of energy during crack formation, are commonly monitored using signal processing techniques. Farhadi et al. [2] used acoustic event-based monitoring to track the breakage of prestressing tendons in concrete bridges, demonstrating the effectiveness of automated signal processing in SHM. Moreover, wavelet analysis has been successfully employed to monitor changes in the structural integrity of bridges by analyzing vibration signals. This method allows for the detection of shifts in natural frequencies, which may indicate the presence of damage. By applying advanced filtering

techniques, noise in the data can be reduced, ensuring that relevant information is retained and analyzed effectively.

In conclusion, signal processing continues to be an essential method in SHM, enabling the extraction of critical information from noisy sensor data. The integration of ML techniques, such as GANs and TSGAN, into signal processing workflows further enhances the ability of SHM systems to detect anomalies and predict structural failures with greater accuracy.
According to Iglesias and Talavera [24], traditional DA techniques such as slicing, jittering, and scaling have been widely used for time series data generation. These methods are critical in SHM, where augmenting sensor data can significantly improve model robustness in detecting anomalies like crack initiation or material fatigue.

These advanced signal processing techniques set the foundation for integrating DL methods, which further enhance the capabilities of SHM systems. The next section explores how DL architectures like CNNs, RNNs, and GANs contribute to automating and improving damage detection in SHM.

## 2.3 Deep Learning

Deep learning (DL) has transformed SHM by providing powerful methods for automating damage detection, prediction, and diagnosis. Unlike traditional ML models, which often rely on manually designed features, DL models can automatically learn relevant features directly from raw sensor data. This ability to extract complex patterns from large datasets has made DL a key component in SHM, where multiple sensors continuously monitor the structural health of critical infrastructures.

### 2.3.1 Generative Adversarial Networks

GANs have gained considerable attention in SHM for their ability to generate synthetic data that can be used to train DL models, especially when real-world data collection is challenging. Introduced by Goodfellow et al. [12], in GANs the generator creates synthetic data, while the discriminator evaluates whether the data is real or generated. This adversarial process allows the generator to improve over time and generate more realistic data. In SHM, GANs have been successfully used to generate synthetic structural data, such as vibration signals, acoustic emissions, and structural images, enabling ML models to better detect rare anomalies. For instance, Foster [29] highlights GANs as an effective method for DA, improving SHM models' ability to identify structural anomalies.

Kushal Virupakshappa and Erdal Oruklu [30] demonstrate the versatility of GANs by generating synthetic ultrasonic signals that closely resemble real-world SHM data. Their GAN model, shown in Figures 2.5 and 2.6, comprises a discriminator network that ensures the synthetic signals are realistic and a generator network specifically designed to produce data that aids in augmenting SHM datasets.

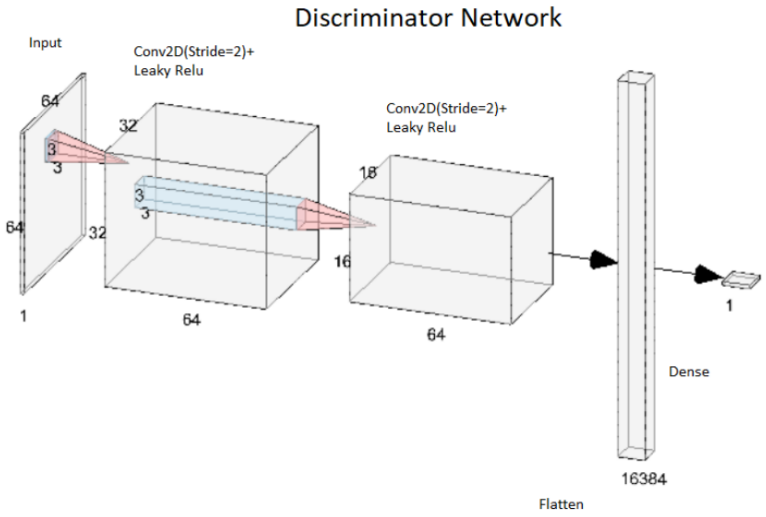



**Figure 2.5:** Architecture of the Discriminator network in Kushal Virupakshappa's GAN model. Adopted from [30].

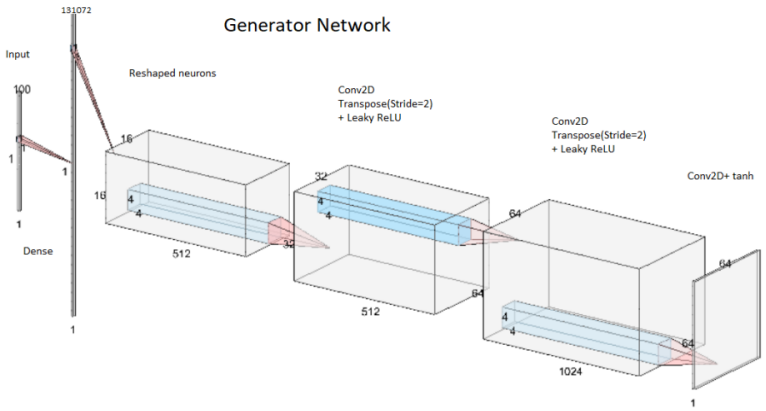



**Figure 2.6:** Architecture of the Generator network in Kushal Virupakshappa's GAN model for generating synthetic structural data. Adopted from [30].

Despite their success, GANs are not without challenges, such as mode collapse,

where the generator produces limited types of data. To address this, Arjovsky et al. (2017) [31] introduced the Wasserstein GAN (WGAN), which improves training stability by minimizing the Wasserstein distance between real and generated data distributions.
Additionally, spectral normalization, introduced by Miyato et al. (2018) [32], controls the Lipschitz constant of the discriminator, helping to prevent training instabilities and improve overall performance. This method has become a standard in GAN architectures, particularly for sensitive SHM applications.
In addition to WGANs and spectral normalization, other improvements in GAN training have been proposed, such as feature matching and minibatch discrimination, as outlined by Salimans et al. (2016) [33]. These techniques mitigate issues like mode collapse and enhance training convergence, which is essential for robust SHM applications.

To ensure the quality of synthetic data, evaluation methods like Maximum Mean Discrepancy (MMD), proposed by Sutherland et al. [34], compare the distributions of real and generated data. In SHM, Maximum Mean Discrepancy helps ensure that synthetic data generated by GANs aligns with the statistical properties of real-world data, thereby improving the reliability of ML models.

Chen et al. [35] introduced a Mixed Autoencoder (MixedAE), a self-supervised learning approach that improves upon traditional Masked Autoencoders (MAEs). MixedAE includes image mixing and homologous recognition tasks during training, which enhances the model's ability to learn object-aware representations (see Figure 2.7).

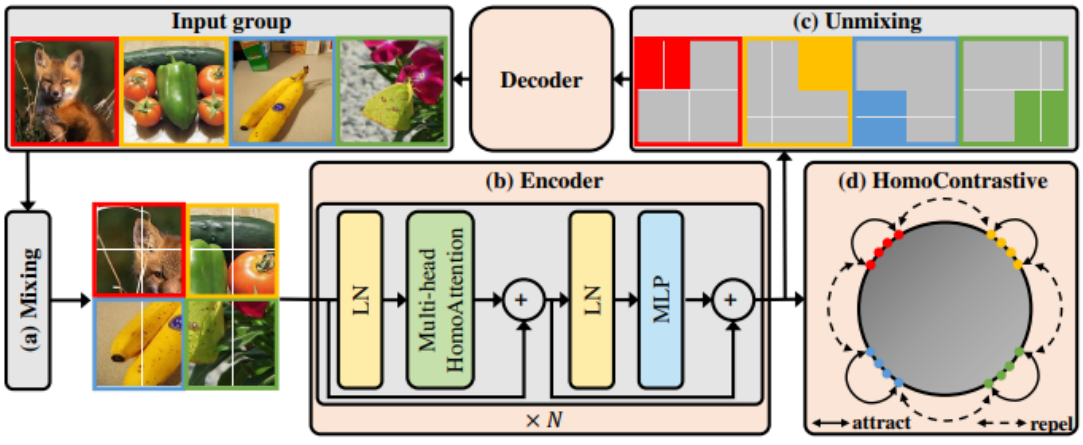



**Figure 2.7:** The architecture of Mixed Autoencoder (MixedAE) designed for self-supervised learning with image mixing. Adopted from [35].

### Variants of Generative Adversarial Networks for Structural Health Monitoring

GANs have evolved significantly, with specialized architectures emerging to address specific challenges in SHM. Eahc variant offers unique advantages for augmenting datasets and enhancing predictive models.

Conditional GANs (CGANs), introduced by Mirza and Osindero [36], generate data conditioned on specific information, like class labels. In SHM, CGANs are valuable for simulating structural responses under different damage conditions, improving predictive accuracy. For instance, Yu et al. [37] demonstrated CGANs' effectiveness in time-series forecasting, an application highly relevant to SHM. By simulating time-series data representing different structural behaviors, CGANs improve predictive accuracy for potential structural failures.
For tasks requiring fine visual detail, discriminative generative networks, as demonstrated by Yu and Porikli (2017) [38], produce ultra-high-resolution images. This is beneficial in SHM for identifying small-scale defects in structures, improving anomaly detection accuracy.

Radford et al. (2015) [39] extended GANs by introducing Deep Convolutional GANs (DCGANs), which utilize deep convolutional layers to generate high-quality images. In the context of SHM, DCGANs are valuable for creating synthetic images of structural elements, enhancing defect detection capabilities. In DCGANs, Convolutional Neural Networks (CNNs) play a central role by enabling the network to learn spatial hierarchies in visual data, which is essential for accurate image generation. CNNs are particularly effective at identifying patterns within images, which makes them suitable for applications like visual inspections in SHM. For example, Gao et al. [19] proposed a method known as Deep Leaf-Bootstrapping GAN, a DCGAN-based architecture for generating synthetic structural images to enhance DA. This architecture uses CNNs to produce diverse and realistic images, facilitating accurate classification of structural damage even when the available data is limited. The Deep Leaf-Bootstrapping GAN introduced by Gao et al. [19] includes several distinctive innovations, summarized as follows:

- **Deep Convolutional GAN (DCGAN)**: The DCGAN architecture generates synthetic structural images for DA.
- **Leaf-Bootstrapping Method**: This method clusters data into smaller subsets, thereby improving the GAN's performance in generating diverse images.
- **Self-Inception Score (SIS)**: A proposed metric for evaluating the diversity and realism of generated images.

In addition to DCGANs, other advanced GAN architectures have been developed to address specific needs in SHM. Multi-stage architectures like StackGAN [40] generate images in progressive stages, enhancing detail in structural images. This feature proves critical for SHM applications, where detecting small-scale defects, such as micro-cracks, requires high-resolution image data. Similarly, ultra-resolution GANs, as demonstrated by Yu and Porikli [38], allow for high-detail anomaly detection, an essential capability for inspecting fine structural elements.

Furthermore, image-to-image translation models, like pix2pix by Isola et al. (2017) [41], learn mappings between paired images, beneficial when paired SHM data is available. CycleGAN [42], however, supports unpaired translation, addressing the challenges of SHM applications where paired datasets are often unavailable.

For SHM applications that focus on time-series data, essential for monitoring dynamic structures, models like TSGAN [43], which generates synthetic sequences based on real sensor data distributions. By enriching SHM datasets with realistic synthetic sequences, TSGAN enables a more comprehensive assessment of structural health.

Another significant GAN variant, context-aware GANs, introduced by Nie et al. [44]. These models condition data generation on contextual factors such as environmental conditions. In SHM, context-aware GANs can simulate bridge conditions under various environmental stresses, which enhances the accuracy of anomaly detection.

Yoon et al. (2019) [45] introduced TimeGAN, which combines GANs and recurrent neural networks to handle time-series data, a crucial component of SHM for analyzing continuous sensor readings. By generating realistic time-series data, TimeGAN enhances ML models' ability to predict structural failures. Similarly, Esteban et al. (2017) [46] explored Recurrent Conditional GANs (RCGANs) for time-series generation, demonstrating their efficacy in modeling temporal dependencies. In SHM, RCGANs can augment datasets with synthetic sequences, supporting predictive models in anticipating structural behavior changes.

### Applications of Generative Adversarial Networks in Structural Health Monitoring

GANs are applied successfully in SHM to generate ultrasonic signals for Non-Destructive Evaluation (Non-Destructive Evaluation), as demonstrated by Virupakshappa and Oruklu [30]. They showed GANs' effectiveness in generating realistic B-Scan images, critical for detecting hidden structural defects. CycleGAN [17]

facilitates unpaired image-to-image translation, enabling synthetic image generation in SHM even with unpaired data. In addition, Zhang et al. [47] demonstrated how DL techniques, including GANs, can be applied to sound event classification, which is relevant for SHM. In particular, acoustic emissions play a critical role in detecting structural damage in bridges and other infrastructure.

Beyond SHM, GANs benefit applications such as smart grids for energy load forecasting. For example, Srivastava et al. (2016) [48] reviewed various methods for short-term load forecasting, noting the utility of GANs in generating synthetic load data. This enhances forecasting models' ability to handle demand fluctuations, improving grid reliability and efficiency.

### 2.3.2 Text-to-Image Generation with MirrorGAN

GANs have been applied to various SHM tasks, especially for generating synthetic data to augment limited datasets. MirrorGAN, introduced by Qiao et al. [49], is a GAN-based model designed for text-to-image generation, ensuring semantic consistency between generated images and the corresponding textual descriptions. MirrorGAN's ability to generate realistic images based on textual descriptions of structural damage helps expand image datasets, especially when real-world images are limited. Figure 2.8 illustrates the architecture of MirrorGAN.

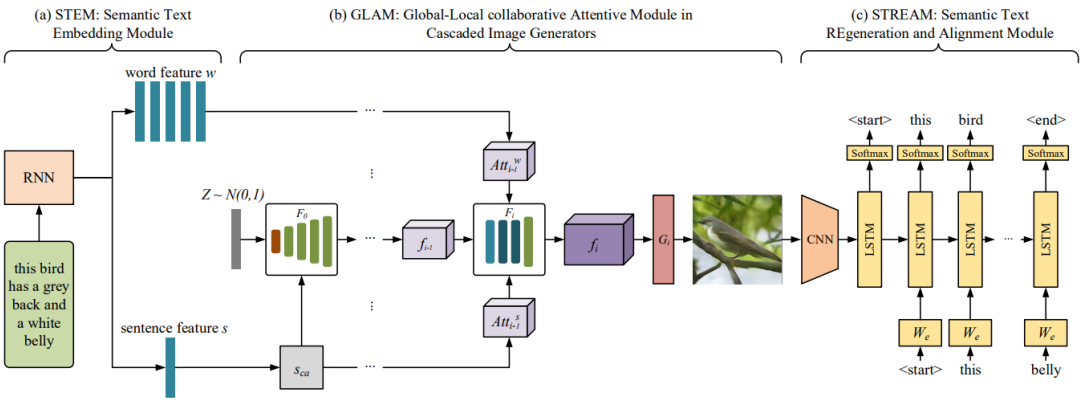



**Figure 2.8:** Architecture of MirrorGAN. Adopted from [49]

### 2.3.3 Applications of Deep Learning in Structural Health Monitoring

DL has been applied to a wide range of SHM tasks, from damage detection to predicting future failures. CNNs are commonly used for visual inspections, detecting

cracks or other forms of structural degradation in bridges and buildings. LSTM networks, on the other hand, are used to analyze time-series data, predicting when and where damage is likely to occur. GANs, by generating synthetic sensor data, allow DL models to be trained even when real-world data is scarce.

Mosalam and Gao [22] explored how GANs can address data scarcity, particularly in complex environments where obtaining labeled data is challenging. Their study also introduced a Deep Convolutional GAN (DCGAN) and a novel Leaf-Bootstrapping method to improve GAN performance. This architecture significantly enhances the quality and diversity of synthetic structural images, even when labeled data is scarce.

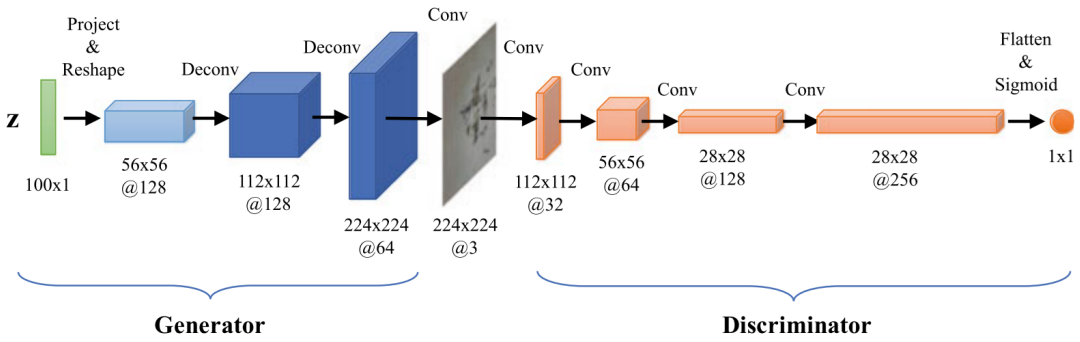



**Figure 2.9:** DCGAN architecture for generating synthetic images. Adopted from [22].

### 2.3.4 Challenges in Deep Learning for Structural Health Monitoring

Applying DL in SHM introduces additional challenges specific to the nature of these advanced models. One of the primary issues is the limited availability of high-quality and labeled datasets for rare events, like structural failures.
Another challenge in DL for SHM is model interpretability. DL models, especially deep neural networks, often function as "black boxes," making it difficult to understand the reasoning behind their predictions. In safety-critical applications like SHM, explainability is essential to help engineers and decision-makers trust the results. Explainable AI (Explainable AI (XAI)) methods are being developed to provide insights into DL models, but implementing XAI in SHM is complex due to the sophisticated structural data involved.
The computational intensity of DL models also poses a challenge, especially for real-time SHM deployment. Models with large computational demands may be unworkable for applications requiring immediate results. Techniques like model

compression, optimization and transfer learning, where pre-trained models on broader datasets are adapted to SHM data, can help mitigate these issues, even though balancing model performance with computational efficiency is still an obstacle.
Finally, DL models in SHM must be robust to noise and environmental variability. SHM data is often affected by external factors such as temperature changes, vibrations, and other environmental noise, which can produce false positives or negatives. Ensuring that DL models can distinguish between actual structural anomalies and environmental variations requires sophisticated pre-processing and tuning.

Overall, the primary challenges in DL for SHM, such as data limitations, comprehensibility, computational demands, and robustness to noise, highlight the need for continued innovation in DL techniques adapted to the specific requirements of SHM.

# Chapter 3

# Methodology

This chapter outlines the key methodologies used in this research, focusing on signal representation techniques and deep generative models applied to Structural Health Monitoring (SHM). These methodologies aim to enhance the detection and prediction of structural anomalies through advanced signal processing and data augmentation (DA) techniques, providing more reliable methods for infrastructure monitoring.

## 3.1 Signal Representation

Signal representation is crucial in processing sensor data for SHM. Choosing the right representation method directly impacts the ability to detect and classify anomalies in the data. Two primary signal representation techniques commonly used in SHM are the Short-Time Fourier Transform (STFT) and wavelet transforms. These methods allow for time-frequency analysis, capturing both temporal and spectral features of the signals.
As highlighted by Farhadi et al. [2], effective signal representation is essential for analyzing acoustic emissions, such as those generated during prestressed tendon wire breakages in bridges. Liu et al. [28] also demonstrated the importance of transforming raw sensor data into frequency-energy features to improve classification accuracy in noisy environments. Both studies emphasize the need for robust time-frequency analysis to enhance SHM system performance.

To gain a comprehensive understanding of the statistical properties of the signal classes, various visualization tools such as kernel density estimate (KDE) plots and violin plots are employed. KDE plots provide a smooth estimate of the probability density function of a random variable, allowing for visual insights into the distribution of features across different classes. Violin plots, on the other hand,

combine the characteristics of a box plot and a KDE plot, effectively illustrating the distribution, spread, and density of the data. These plots are particularly useful for highlighting differences across signal classes and help in identifying distinct features relevant to SHM applications.

### 3.1.1 Short-Time Fourier Transform

STFT is a widely used technique in signal processing that converts a time-domain signal into a time-frequency representation. It is especially useful for analyzing non-stationary signals, such as those encountered in SHM, where structural anomalies like crack formation and wire breakage generate transient events.

The STFT is computed by dividing the signal into short overlapping segments and applying the Fourier Transform to each segment, producing a two-dimensional representation of the signal in both time and frequency. This method provides insights into how frequency components evolve over time, which is essential for diagnosing structural health.

The discrete STFT of a signal $x[n]$ is defined as:

$$\text{STFT}(x[n])(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n} \tag{3.1}$$

where $w[n]$ is the window function, $\omega$ is the angular frequency, and $n$ is the time index. By applying the STFT, SHM systems can analyze acoustic signals in both time and frequency domains, improving the detection of subtle changes that indicate damage.

## 3.2 Deep Generative Models

Generative models play a vital role in SHM, especially when real-world data is limited. These models are capable of generating synthet ic data that can augment datasets and enhance the detection of structural anomalies.
As noted in [29], GANs are particularly effective for generating synthetic data in SHM, where collecting large datasets of structural anomalies is difficult. By training GANs on real sensor data, SHM systems can generate new, plausible examples of damage, such as cracks or wire breakage, enhancing the detection and classification of structural events.

This section discusses three primary types of GANs used in this research: Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and Least Squares

GAN (LSGAN). These models were selected due to their proven effectiveness in generating high-quality synthetic data and addressing common issues in GAN training, such as mode collapse and training instability.

### 3.2.1 Deep Convolutional Generative Adversarial Networks

DCGAN is a variant of the traditional GAN that uses convolutional layers in both the generator and discriminator, improving the generation of high-resolution images. The architecture applies convolutional and transposed convolutional layers, which are better suited for processing and generating image-like data compared to fully connected layers.

**DCGAN Generator** The generator in DCGAN takes as input a noise vector $z$, typically sampled from a standard normal distribution:

$$z \sim \mathcal{N}(0, 1) \tag{3.2}$$

The generator's objective is to map this input to a realistic data sample, such as an image or spectrogram. It consists of a series of transposed convolutional layers that progressively upsample the input. The generator's loss function is designed to maximize the discriminator's output for generated samples:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim\mathcal{N}(0,1)}[\log D(G(z))] \tag{3.3}$$

This loss encourages the generator to create data that the discriminator classifies as real, improving the quality of generated data. The detailed architecture of the generator is outlined in Table 5.2.

**DCGAN Discriminator** The discriminator is a convolutional neural network tasked with classifying input data as real or generated. It down-samples the input through a series of convolutional layers and outputs a scalar probability. The discriminator's loss function is defined as:

$$\mathcal{L}_{DCGAN_D} = -\mathbb{E}_{x\sim\mathbb{P}_r}[\log D(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[\log(1 - D(G(z)))] \tag{3.4}$$

where:

- $D(x)$ is the discriminator's output for real data $x$,
- $G(z)$ is the generator's output for the noise vector $z$,
- $\mathbb{P}_r$ is the distribution of real data,
- $\mathbb{P}_z$ is the distribution of input noise for the generator.

The goal of the discriminator is to correctly classify real and fake data, maximizing the log-probability of classifying real data as real and minimizing the probability of classifying generated data as real.

### 3.2.2 Wasserstein Generative Adversarial Networks

WGAN addresses common issues in traditional GANs, such as mode collapse, by introducing the Wasserstein distance, which provides smoother training dynamics. Mode collapse occurs when the generator in a GAN produces a limited variety of outputs, effectively "collapsing" to a few modes or patterns rather than generating a diverse set of samples. This issue limits the generator's ability to capture the full distribution of the data, leading to poor performance in applications requiring high variability.

**Wasserstein Loss Function** The loss function in WGAN is based on the Wasserstein distance (or Earth Mover's distance), which measures how much "work" is needed to transform the generated data distribution $\mathbb{P}_g$ into the real data distribution $\mathbb{P}_r$. The critic (discriminator in WGAN) is trained to maximize this distance:

$$\mathcal{L}_{WGAN_D} = \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)] - \mathbb{E}_{z\sim\mathbb{P}_z}[D(G(z))] \tag{3.5}$$

where $D(x)$ is the critic's output for real data and $D(G(z))$ for generated data. Unlike in traditional GANs, the critic does not output probabilities but scores representing how "real" the data appears.

**Gradient Penalty** To enforce the 1-Lipschitz constraint necessary for stable training, WGAN introduces a gradient penalty:

$$\mathcal{L}_{GP} = \lambda \cdot \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[\left(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1\right)^2\right] \tag{3.6}$$

where:

- $\lambda$ is the regularization coefficient, often set to 10,
- $\mathbb{P}_{\hat{x}}$ is the distribution of points sampled along straight lines between real and generated data,
- $\|\nabla_{\hat{x}} D(\hat{x})\|_2$ is the L2 norm of the gradient of the critic with respect to $\hat{x}$.

The gradient penalty helps ensure that the critic's output respects the Lipschitz constraint, which stabilizes training and prevents overfitting.

**WGAN Generator Objective** The generator's objective in WGAN is to produce data that maximizes the critic's score:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim\mathbb{P}_z}[D(G(z))] \tag{3.7}$$

This leads to the generator learning to produce data that the critic evaluates as similar to real data, improving the quality of generated samples.

### 3.2.3 Least Squares Generative Adversarial Networks

LSGAN modifies the traditional GAN architecture by using a least-squares loss to stabilize training and reduce issues like vanishing gradients.

**LSGAN Discriminator Loss** The discriminator in LSGAN minimizes the following least-squares loss:

$$\mathcal{L}_{LSGAN_D} = \frac{1}{2}\mathbb{E}_{x\sim\mathbb{P}_r}\left[(D(x)-1)^2\right] + \frac{1}{2}\mathbb{E}_{z\sim\mathbb{P}_z}\left[D(G(z))^2\right] \tag{3.8}$$

where:

- $D(x)$ is the discriminator's output for real data,
- $G(z)$ is the generator's output for input noise $z$,
- $\mathbb{P}_r$ is the real data distribution,
- $\mathbb{P}_z$ is the noise distribution for the generator.

This loss function helps ensure that real data is classified close to 1 and generated data close to 0, resulting in smoother training.

**LSGAN Generator Loss** The generator minimizes the least-squares error to make the generated data as close to real data as possible:

$$\mathcal{L}_{LSGAN_G} = \frac{1}{2}\mathbb{E}_{z\sim\mathbb{P}_z}\left[(D(G(z))-1)^2\right] \tag{3.9}$$

This loss encourages the generator to produce samples that are classified by the discriminator as real (i.e., $D(G(z)) \approx 1$). By implementing these deep generative models, SHM systems can enhance data generation, improve the training of classifiers, and detect structural anomalies more effectively.

## 3.3 Model Evaluation: Fréchet Inception Distance Score

In evaluating the performance of generative models such as GANs, the Fréchet Inception Distance (FID) score is a widely adopted metric used to assess the similarity between the real and generated data distributions. Introduced by Heusel et al. (2017) [50], the FID score measures the quality of generated samples by comparing the statistical properties of their distribution with those of real samples. Specifically, it calculates the Fréchet distance (or Wasserstein-2 distance) between

the two distributions in the feature space of a pre-trained neural network, commonly the Inception network. This feature space represents high-level attributes of the images, enabling the FID score to capture perceptual similarities rather than pixel-level differences.

**Mathematical Definition:** The FID score is calculated based on the mean and covariance of the real and generated data distributions, denoted as $\mu_r, \Sigma_r$ and $\mu_g, \Sigma_g$, respectively. The formula for the FID is given by:

$$\text{FID} = ||\mu_r - \mu_g||^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r \Sigma_g)^{1/2}) \tag{3.10}$$

where:

- $\mu_r$ and $\mu_g$ are the mean feature vectors of the real and generated samples,
- $\Sigma_r$ and $\Sigma_g$ are the covariance matrices of the real and generated samples,
- Tr denotes the trace of a matrix.

The FID score captures both the difference in means and the difference in covariances between the real and generated data distributions. A lower FID score indicates that the generated samples are more similar to the real data, while a higher score reflects greater dissimilarity.

**Feature Space Representation:** The FID score uses the output of a pre-trained Inception network to compute the mean and covariance. Specifically, the activations from an intermediate layer of the network are extracted for both real and generated data. This transforms the data into a feature space where the comparison between distributions can be more meaningful.
The FID score is particularly useful in the context of SHM because it provides an objective measure of how closely the synthetic spectrograms resemble the real sensor data. In this research, the FID score was used to evaluate and compare the performance of the DCGAN, WGAN, and LSGAN models, guiding the selection of the most effective model for DA.

# Chapter 4

# Data Acquisition and Problem Context

As the field of Structural Health Monitoring (SHM) evolves, effective data acquisition strategies become essential for understanding and predicting structural behavior under different conditions. This chapter presents the methodologies used for in-situ data acquisition on prestressed concrete bridges. Specifically, the following sections describe the experimental setup and the process of data collection, focusing on the destructive testing performed on two prestressed concrete bridges scheduled for demolition.

## 4.1 In-Situ Acquisition

The in-situ data acquisition was conducted on two prestressed concrete bridges, Le Pastena and Cerqueta, located on the A24 highway in the province of L'Aquila, which were scheduled for demolition and reconstruction. An agreement between the Politecnico di Torino and Autostrade dei Parchi provided a unique opportunity to perform controlled breakage tests on the prestressing wires of these bridges, which had already been structurally weakened in preparation for their planned demolition. These tests provided valuable insights into the bridges' structural behavior under failure conditions, without physically destroying the bridges themselves.

The experiments and data collection were conducted on two spans of the Cerqueta bridge and one span of the Le Pastena bridge. Figures 4.1 and 4.2 show the general view of the tested bridge structures, highlighting the areas where the sensors were installed and data acquisition took place.

**(a)** General view of the left span of the Cerqueta bridge.

**(b)** Close-up of the right span of the Cerqueta bridge used for data collection.

**Figure 4.1:** Views of the Cerqueta bridge showing both the left and right spans where testing was performed.

**Figure 4.2:** View of the Le Pastena bridge.

### 4.1.1 Data Acquisition Setup

Since the bridge decks were inaccessible due to their considerable height and because an under-bridge inspection unit (a platform used to access the underside of a bridge from above) could not be used—the weakening of the bridges had already begun in preparation for demolition — a MEWP was utilized to reach the side beams. Two holes were drilled at different heights on the side beams to

expose the prestressing wires. The setup included the installation of eight acoustic emission (AE) sensors and two accelerometers (Models 805 and 805M1) to monitor the structural vibrations during testing.

**Sensor Placement:** The acoustic emission sensors were strategically positioned to capture relevant data: the first sensor was placed 1.5 m from the cutting area, the second 1.5 m from the first, and the remaining sensors were spaced 1 m apart. This arrangement allowed for comprehensive monitoring of the vibrational response during the destructive testing.

**(a)** Left span of the Cerqueta bridge. **(b)** Right span of the Cerqueta bridge.

**Figure 4.3:** Configuration of acoustic emission sensors on both spans of the Cerqueta bridge.

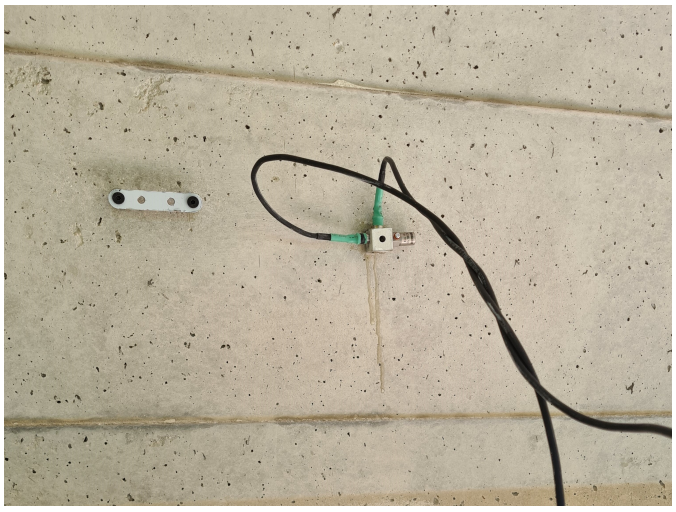


**Figure 4.4:** Close-up view of the installed accelerometers and their connection setup.

### Accelerometer Data Acquisition

For this study, we used two types of accelerometers, Model 805 and Model 805M1, manufactured by TE Connectivity Measurement Specialties. These accelerometers were chosen for their accuracy, durability, and suitability for structural monitoring. Using two models allowed us to compare performance under similar conditions and enhance data reliability through cross-validation.
Each accelerometer was assigned to capture a specific component: one recorded the longitudinal acceleration component, while the other captured the transverse acceleration component. This setup was essential for detailed vibrational analysis, as it allowed us to collect comprehensive data on the structural response during controlled cable cutting tests. With a sampling rate of 96,000 Hz and an effective frequency range of 0–48,000 Hz, high-resolution data acquisition enabled the detection of fine changes in structural behavior. Each data line recorded left and right channel values, supporting in-depth time-frequency analysis. The accelerometers were positioned to effectively capture their respective longitudinal and transverse vibrational data, maximizing data relevance.

This configuration provided a robust source of real-time data on the structural integrity and vibrational behavior of bridge structures during cutting tests, supporting subsequent analyses on structural response and integrity.

**Model 805 Accelerometer** The Model 805 is a compact, low-cost accelerometer, built with a 3-pin TO-5 header, and offers a flat frequency response up to 15 kHz, ideal for capturing a range of vibrations. Available in ±50g and ±500g ranges, it suits various vibration intensities in bridge monitoring. Key features of the Model 805 are summarized in Table 4.1, highlighting strengths in durability, signal clarity, and integration with data acquisition systems.

**Table 4.1:** Key features of the Model 805 accelerometer used for data acquisition.

| **Feature** | **Description** |
|---|---|
| Manufacturer | TE Connectivity Measurement Specialties. |
| Wide frequency range | 1 to 10,000 Hz (±3 dB), suitable for capturing both low and high-frequency vibrations. |
| Low noise levels | Ensures high-quality signal acquisition, minimizing interference and preserving data fidelity. |
| Stainless steel casing | Provides durability and protection from environmental conditions, essential for outdoor monitoring applications. |
| IEPE interface | Facilitates easy integration with data acquisition systems, enhancing setup efficiency and compatibility. |

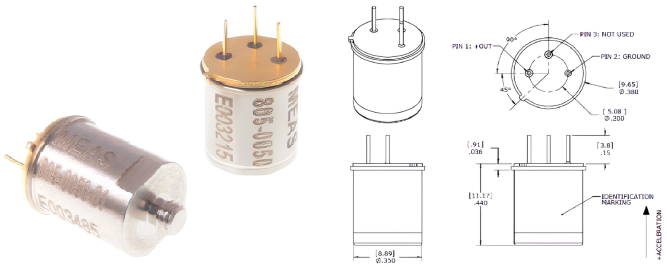



**Figure 4.5:** Model 805 Accelerometer by TE Connectivity Measurement Specialties. This diagram shows the structure and pin layout of the accelerometer used in the study.

**Model 805M1 Accelerometer** The Model 805M1, also made by TE Connectivity Measurement Specialties, is a similar device with a 3-wire voltage output. It comes in dynamic ranges from ±20g to ±500g and offers a flat frequency response up to 12 kHz. This accelerometer is known for its low power consumption (less than 0.80 mA), making it ideal for continuous monitoring in embedded systems. The key features of the Model 805M1 include:

**Table 4.2:** Key features of 805M1 accelerometer used for data acquisition.

| **Feature** | **Description** |
|---|---|
| Manufacturer | TE Connectivity Measurement Specialties. |
| Wide frequency range | 0.4 to 12,000 Hz (±3 dB). |
| Low power usage | Less than 0.80 mA, perfect for long-term use. |
| Grounded casing | Reduces interference and noise in the signal. |
| Easy mounting | Adhesive options make it simple to attach to surfaces. |

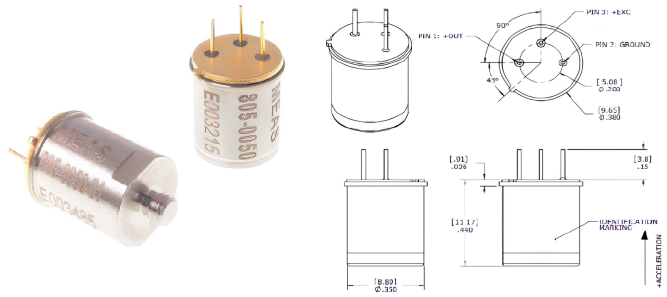



**Figure 4.6:** Model 805M1 Accelerometer by TE Connectivity Measurement Specialties. This figure shows the size and pin layout of the accelerometer.

Both accelerometers were securely mounted using adhesive to ensure they were

properly attached to the surface of the structure, which helped improve the accuracy of the vibrational data collected. These accelerometers were connected to a central data acquisition system that sampled the vibrations at high frequencies, enabling real-time monitoring of the bridge's behavior during the controlled cutting of prestressing wires, one at a time.

### 4.1.2 Data Collection Process

Connected to a central data acquisition system, the accelerometers recorded signals continuously during the controlled breakage of the prestressing wires. This continuous recording was essential for capturing the vibrations associated with the rupture events, providing real-time data on the dynamic response of the bridge structure during the tests. The high sensitivity of the accelerometers allowed us to detect even small changes in vibration, providing valuable data on the condition of the prestressing wires. This data will later be analyzed to evaluate the integrity of the structure.

#### Visualization of Audio Signal in Audacity

The extracted audio signals were analyzed using Audacity, an open-source digital audio editor, to ensure data quality and to identify significant events such as the breakage of prestressing wires. Figure 4.7 shows an example of a signal visualized in Audacity, representing the waveform of a breakage event captured during the tests. This visual analysis helped in verifying the integrity of the recorded signals and in pinpointing moments of structural significance.

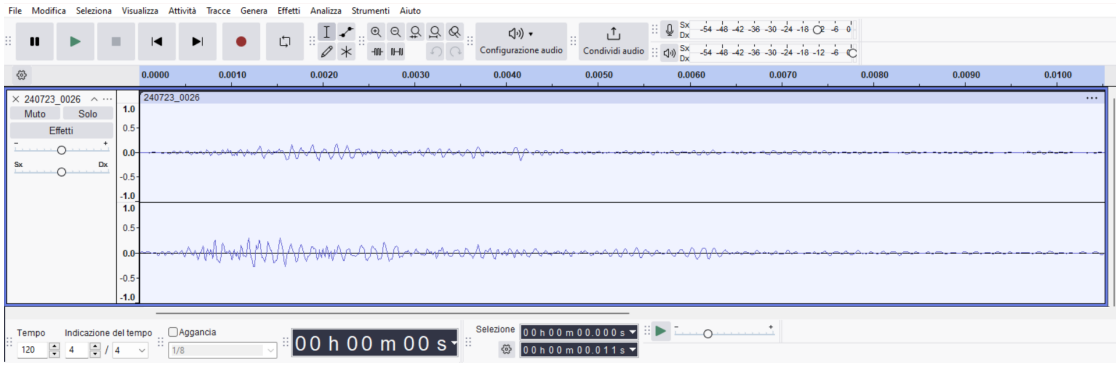



**Figure 4.7:** Screenshot of a breakage signal analyzed in Audacity. The visualization helps in verifying the integrity of the recorded signal and in identifying relevant structural events.

**Extracted Audio Signal Example**

The audio data collected during the in-situ testing was sampled at a frequency of 96,000 Hz and recorded in stereo format, with the left and right channels captured simultaneously. The extracted data was then converted into text format to facilitate further processing and analysis. The signal values are represented in a linear scale without specific units, as they are normalized sample values. Normalization ensures that the data is scaled consistently for analysis. Table 4.3 provides an example segment showcasing these values from both the left and right channels.
The interpolation method applied during processing was linear, ensuring continuous and smooth data for analysis. The table demonstrates how each line corresponds to simultaneous readings from both channels. This format aids in identifying patterns related to structural responses captured during events such as cable cutting and subsequent hammering activities.

**Table 4.3:** Example of an audio signal extracted from a recording during the bridge tests. The data is shown in a linear scale, recorded at 96000 Hz.

| Left Channel | Right Channel |
|---|---|
| 0.00018 | -0.02465 |
| -0.00374 | -0.01714 |
| -0.00543 | -0.01151 |
| ... | ... |

### 4.1.3 Event Monitoring

During the rupture of the cables, the data from the accelerometers were closely monitored. To facilitate controlled cutting and detailed data collection, we undertook preparations to expose and cut the prestressing wires. The process included drilling access holes and cutting the cables with a multi-tool by Dremel, generating vibrations that were recorded and classified as part of the dataset. After the cuts were made, hammering was used to dislodge and remove the cut cable sections, producing additional vibrational data for analysis. These steps were essential for comprehensive structural monitoring and were documented in detail.

Figures 4.8 and 4.9 present a visual overview of the preparation and cutting process on the Cerqueta and Le Pastena bridges, respectively. The images illustrate the exposed cables, cutting actions, and the arrangement of wires post-cut, showcasing the key aspects of the monitoring and preparation phases:

- **Cerqueta Bridge:** The series of images in Figure 4.8 highlight the initial exposure, cutting, and detailed view of prestressed strands during the testing

on the Cerqueta bridge. The arrangement and condition of the cables post-cut are also shown.

- **Le Pastena Bridge:** Figure 4.9 presents the process on the Le Pastena bridge, emphasizing the exposure and cutting of parallel wires. The close-up images detail the arrangement after the cutting process, showing the structural response to the controlled cuts.

**(a)** Opened section showing two cable cut locations on Cerqueta Bridge.

**(b)** Cut section showing prestressed strands on Cerqueta Bridge.

**(c)** Detail of exposed strands on Cerqueta Bridge.

**(d)** Close-up of cable arrangement after cut on Cerqueta Bridge.

**Figure 4.8:** Details of the cut sections on the prestressing wires of the Cerqueta Bridge using a multi-tool by Dremel. The preparation involved using a hammer to remove loosened cable pieces post-cut.

**(a)** Initial exposure of parallel wires.

**(b)** Cutting process of parallel wires.

**(c)** Close-up after cutting.

**(d)** Detailed view of wires post-cut.

**Figure 4.9:** Details of the cut sections on the Le Pastena bridge showing the parallel wires. The wires were cut using a multi-tool by Dremel, and a hammer was used for removing cut pieces.

The data collected during these tests will be analyzed using advanced signal processing and deep generative techniques, as outlined in Chapter 3. This analysis will enhance our understanding of the structural responses observed and contribute to the development of more effective SHM strategies.

# Chapter 5

# Results and Discussion

This chapter presents a comprehensive analysis of the data collected during the SHM process. The dataset comprises signals categorized into four classes, as summarized in Table 5.1. All signals are represented on a linear scale, with amplitudes expressed in arbitrary units. This classification allows for targeted analysis of each type of event captured during the SHM process.

**Table 5.1:** Dataset Summary: Signal Characteristics and Counts per Class

| **Characteristic** | **Value** |
|---|---|
| Samples per Signal | 1000 |
| Signal Length (s) | 0.0104 |
| Sampling Frequency (Hz) | 96000 |

| **Class** | **Max Mean Amplitude** | **Peak Amplitude** | **Signals** |
|---|---|---|---|
| Breakage | 0.1519 | 0.3333 | 200 |
| Hammering | 0.0089 | 0.0419 | 282 |
| Dremel | 0.0151 | 0.1606 | 466 |
| Noise | 0.0103 | 0.0359 | 400 |

The analyses presented in this chapter follow the methodologies described in Chapter 3, where signal representation techniques and deep generative models were detailed. All data processing and model training were implemented in Python using the Google Colab environment. This platform provided the computational resources required to manage large datasets and train deep generative models effectively.

In the time domain analysis, we evaluate various statistical parameters such as maximum amplitude, mean, standard deviation, skewness, and kurtosis to reveal

key features of each signal type. Visual representations like histograms, kernel density estimate (KDE) plots, and violin plots demonstrate the differences and relationships between these parameters. In the frequency domain, the STFT is applied to capture the time-frequency characteristics of the signals, and the resulting spectrograms are used as inputs for deep generative models. Then, the performance of three different types of GANs is then compared: Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and Least Squares GAN (LSGAN).

By using statistical analyses in both the time and frequency domains, followed by machine learning techniques, this chapter aims to find insights crucial for developing reliable automated monitoring systems in real-world SHM applications.

## 5.1 Time Domain Analysis

In this analysis, signals from the different classes—breakage, hammering, multi-tool by Dremel operation, and noise—were examined over a 0.0104-second segment, corresponding to a sample length of 1000 samples. This specific duration aligns with the typical timescale of prestressing wire rupture in concrete structures, making it relevant for capturing the transient events associated with structural failures. To ensure consistency, signals from the other classes were segmented to the same length for comparison.

Figure 5.1 shows two time domain signals for each class. From these updated visual plots, the following observations were made:

- **Breakage Signals**: Tend to display initial amplitude spikes associated with sudden, high-energy events, often stabilizing afterward. This behavior may relate to structural failures, such as the breaking of prestressed tendons in concrete bridges.
- **Hammering Signals**: Typically show relatively uniform amplitude changes with minimal peaks, reflecting steady impacts from a hammer.
- **Dremel Signals**: Often exhibit gradual increases in amplitude mid-signal, which may correlate with the operational pattern of the multi-tool by Dremel.
- **Noise Signals**: Usually consist of random fluctuations without significant peaks, representing background noise.

This time domain analysis provides an initial understanding of the signal nature in each class. The distinctive features of breakage, hammering, and dremel signals compared to noise highlight the importance of automated monitoring to distinguish between different structural events.

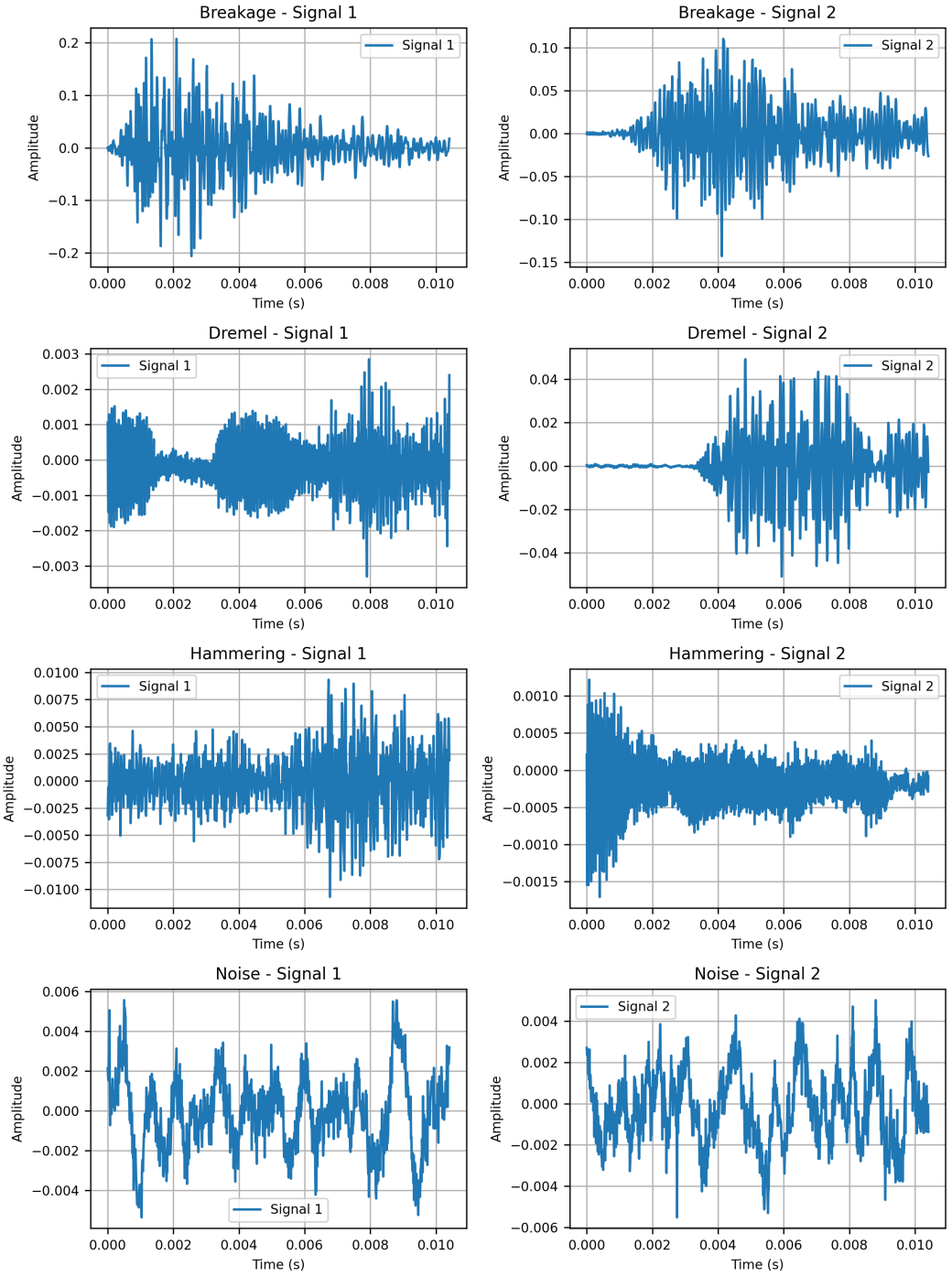



**Figure 5.1:** Time domain signals for different classes.

### 5.1.1 Time Domain Statistical Analysis

To further compare the signal classes, we analyzed key statistical parameters and visualized them using histograms, kernel density estimates (KDE), and violin plots. Figure 5.2 presents histograms that provide an initial overview of the distributions for each class, highlighting the following key characteristics.

- **Maximum and Minimum Amplitude**: Breakage signals generally show higher maximum and minimum amplitude values, indicating sudden, high-energy events. In contrast, noise signals display lower values, consistent with random background fluctuations.
- **Mean Amplitude**: Noise signals have a mean amplitude close to zero, suggesting balanced fluctuations without strong deviations. Breakage signals display a skewed mean, indicative of abrupt energy bursts.
- **Standard Deviation**: Breakage signals have wider distributions, indicating greater variability compared to noise signals, which show a narrower spread.
- **Skewness and Kurtosis**: Breakage signals are highly skewed and have higher kurtosis values, suggesting infrequent but sharp peaks. This contrasts with the relatively balanced and lower kurtosis of noise signals.

While histograms provide a basic overview of statistical characteristics, KDE and violin plots offer deeper insights into the distribution and density of these parameters across signal classes. Figure 5.3 shows KDE plots with a bandwidth adjustment (`bw_adjust=0.3`), estimating the probability density for each statistical parameter and offering a nuanced view of data distribution. This adjustment yields sharper, distinct peaks that highlight fine inter-class differences. For instance, breakage signals show sharp yet lower peaks due to their sudden, singular energy bursts, resulting in fewer high-value points and a less dense distribution compared to repetitive signals like hammering or dremel. Conversely, noise signals exhibit a wider spread, reflecting their random, balanced nature.

Figure 5.4 presents the violin plots that capture both the distribution and the underlying probability density. These plots provide a more fine view compared to histograms and KDE plots, highlighting differences in the data spread and variability within each class. Breakage signals, for instance, have extended tails, demonstrating variability and the presence of extreme values. In contrast, noise signals exhibit a more compact, symmetrical shape, which reflects their stable and balanced nature. The elongated nature of the breakage signals' distribution, particularly in the maximum and minimum amplitude plots, underscores their tendency for extreme events. Additionally, the skewness and kurtosis plots show that breakage signals possess significant asymmetry and rare high peaks.

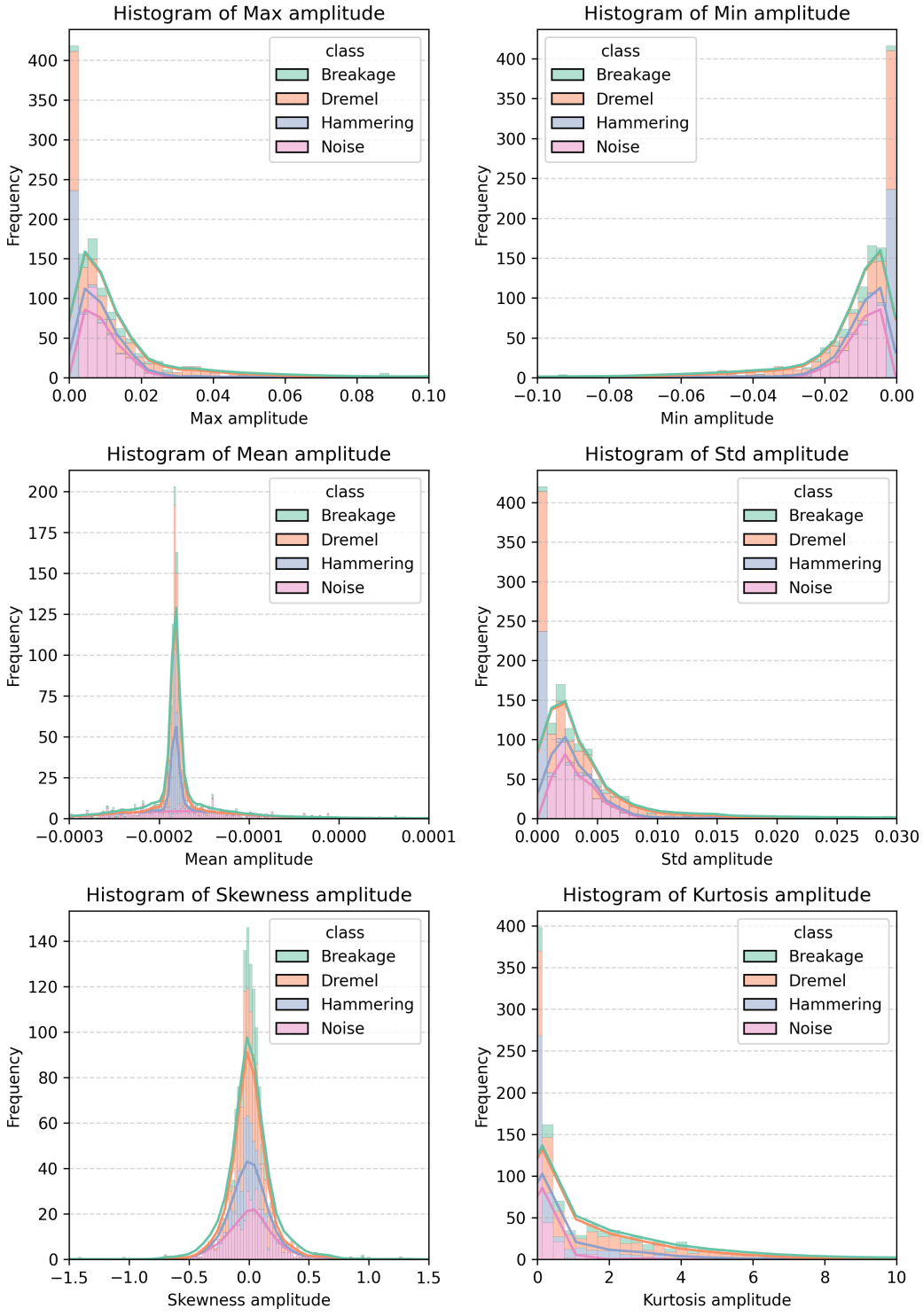



**Figure 5.2:** Histograms of various statistical parameters for each class in the time domain.

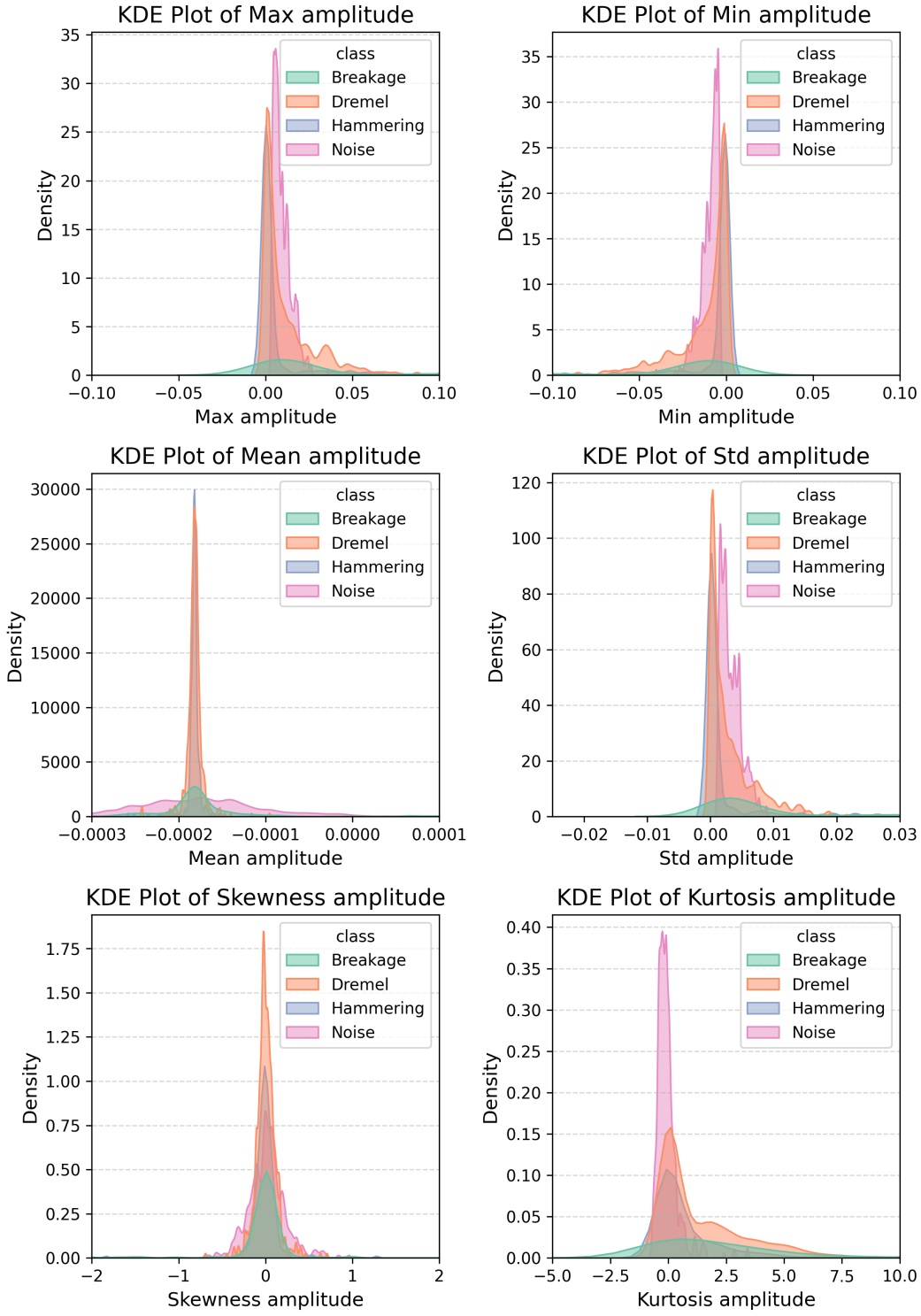



**Figure 5.3:** Kernel density estimate (KDE) plots showing the probability density distribution for each statistical parameter by class.

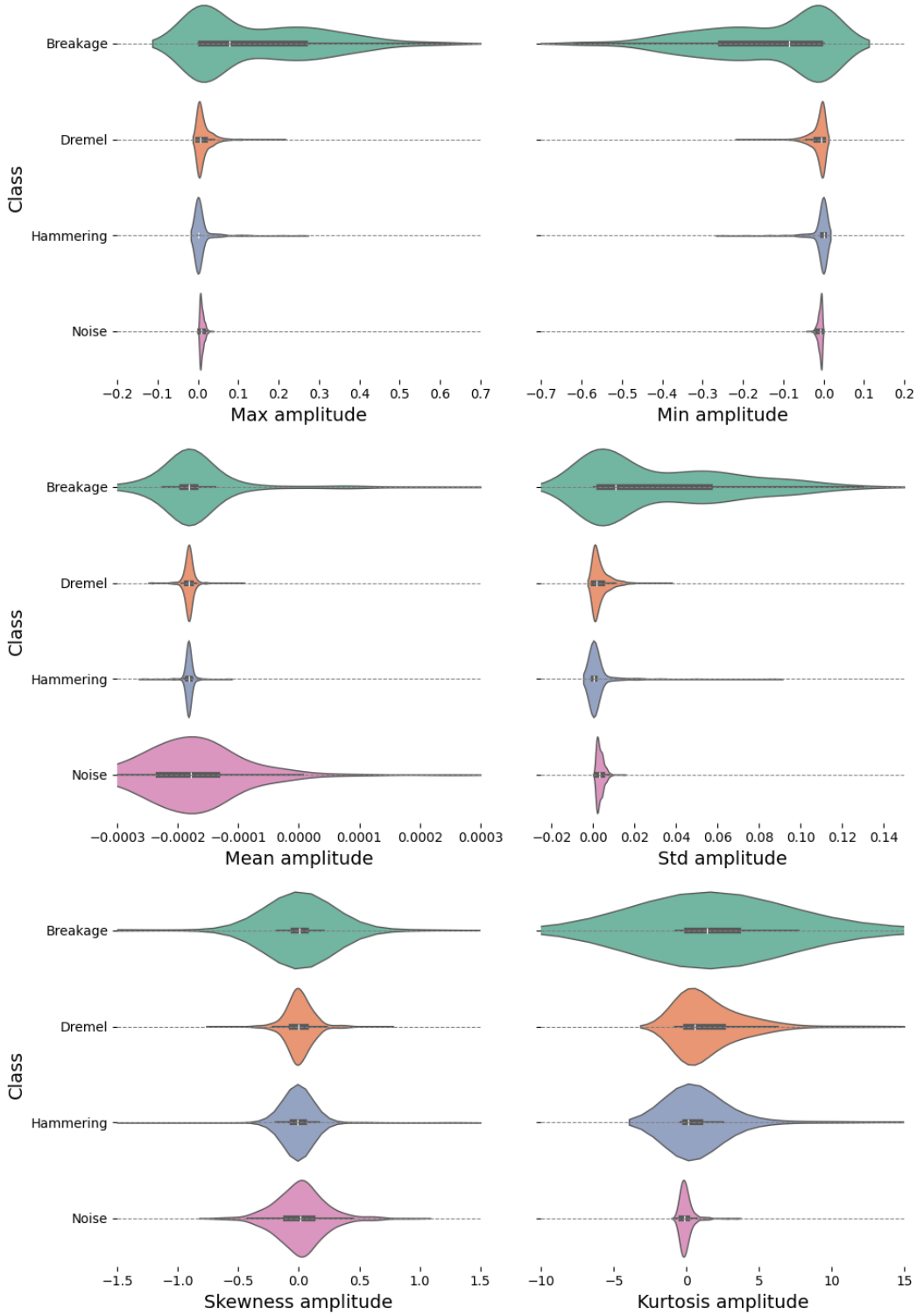



**Figure 5.4:** Violin plots showing the distribution of statistical parameters by class in the time domain.

## 5.2 Frequency Domain Analysis

Analyzing the frequency domain properties of signals provides essential insights into their spectral characteristics, which are crucial for distinguishing various structural events. This analysis was conducted using the Fourier Transform, a tool that allows for a detailed examination of each signal's frequency components and highlights the unique characteristics of different signal classes. By examining the amplitude spectrum, derived from applying the Fourier Transform, we can gain a deeper understanding of how energy is distributed across frequencies, offering insights into the underlying physical phenomena of each class.

Figure 5.5 illustrates the frequency domain representation of selected signals from each class. The amplitude spectrum was calculated for each signal class, with notable patterns observed across the different types of events. Key observations include:

- **Breakage Signals**: These signals show clear, prominent peaks in the low frequency range. This concentration of energy suggests a sudden release of energy typical of a breakage event. The amplitude gradually decreases at higher frequencies, indicating energy dissipation.
- **Hammering Signals**: The amplitude spectrum of hammering signals is characterized by evenly distributed peaks within the low-frequency range, suggesting a consistent and repetitive energy input, as expected from manual impacts.
- **Dremel Signals**: These signals exhibit distinct, sharp peaks at specific frequency values. These peaks correspond to the operational speed of the multi-tool by Dremel, making this class easily identifiable based on its frequency characteristics.
- **Noise Signals**: The noise signals show a relatively flat frequency spectrum with no significant peaks, which aligns with the nature of ambient noise that is more random and lacks a structured frequency pattern

The Fourier Transform analysis underscores the distinct frequency characteristics of breakage and Dremel signals due to their sharp frequency peaks, which are particularly useful for differentiating between these types of structural events. The amplitude spectra, as visualized, help in identifying the unique energy profiles of each class, thereby enhancing the effectiveness of signal classification within SHM applications.

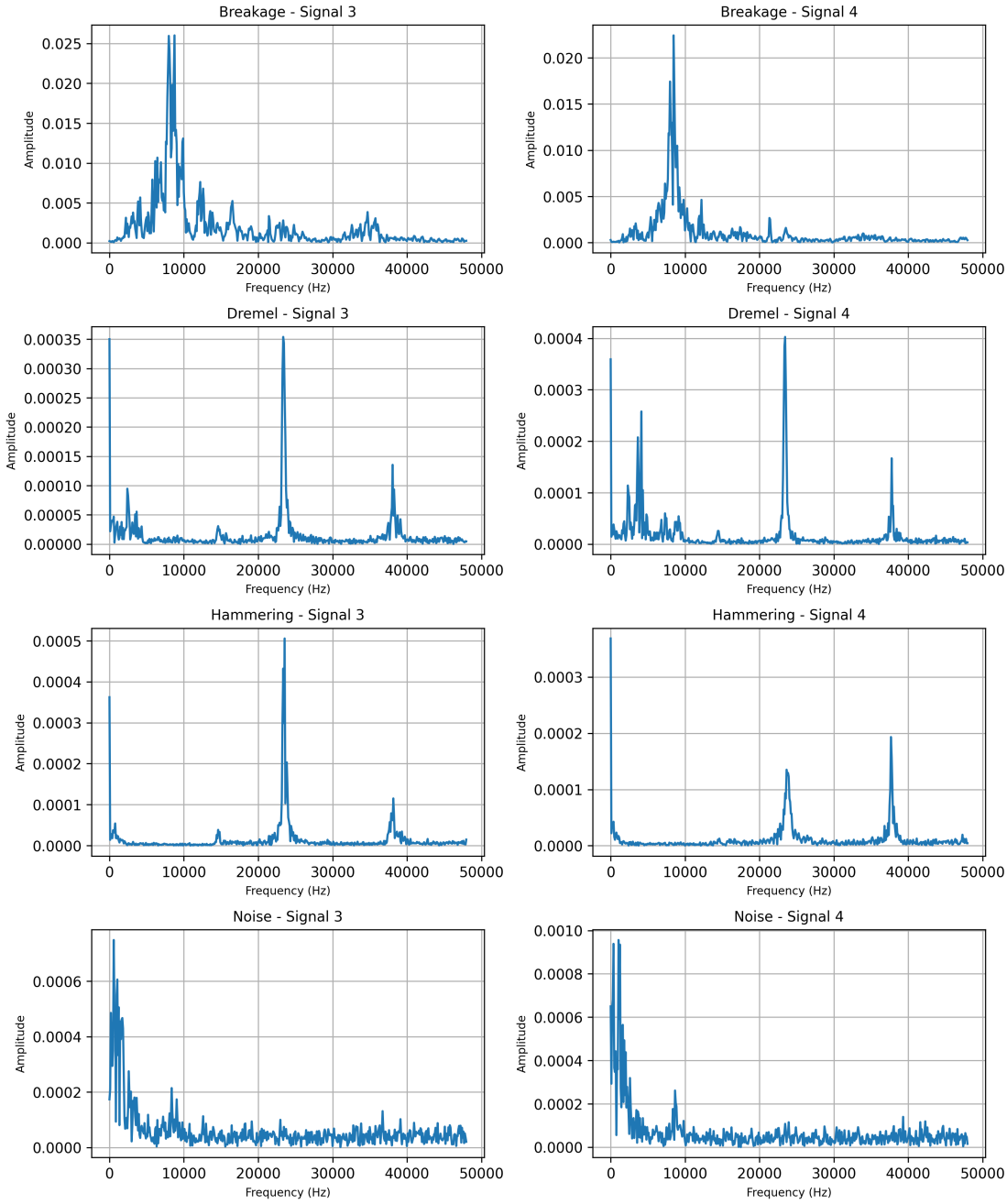



**Figure 5.5:** Frequency domain grid plot showing the amplitude spectrum of signals from different classes. Each subplot represents the frequency characteristics of selected signals from the breakage, hammering, dremel, and noise classes.

### 5.2.1 Frequency Domain Statistical Analysis

To provide a comprehensive comparison of the frequency domain characteristics across different signal classes, we generated both violin plots and KDE plots of various statistical metrics. These visualizations offer insights into the distribution of parameters such as maximum magnitude, total power, spectral entropy, skewness, and kurtosis for each class. Figure 5.6 represents the violin plots that illustrate the spread and density of statistical parameters such as maximum magnitude, total power, spectral entropy, skewness, and kurtosis. Key observations include:

- **Max Magnitude and Total Power**: Breakage signals show higher values for both parameters, indicating their intense nature, while noise signals display lower values, consistent with random, low-energy characteristics.

- **Spectral Entropy**: Noise signals have the highest spectral entropy, indicating a more complex and random frequency distribution. This contrasts with the structured content observed in breakage and hammering signals.

- **Skewness and Kurtosis**: Breakage signals show higher skewness and kurtosis, indicating sharp and infrequent peaks, reflecting concentrated energy at specific frequencies. Noise signals show a more balanced distribution.

KDE Plots (Figure 5.7) presents the KDE plots that provide smooth probability density estimates for the same statistical parameters, highlighting the distribution shapes and density for each class. Key observations from the KDE plots include:

- **Max Magnitude and Total Power**: The KDE plots confirm that breakage signals have a higher density of maximum magnitude and total power values compared to other classes. The KDE visualization smooths out individual data points, making it easier to see the overall trend.

- **Spectral Entropy**: The KDE plots show that noise signals have a wider distribution of spectral entropy, reinforcing their complex frequency nature.

- **Skewness and Kurtosis**: The KDE plots emphasize the sharp, skewed distributions of breakage signals, showcasing their unique frequency characteristics compared to the more symmetric distribution of noise signals.

Together, the violin plots and KDE plots provide a detailed comparison of how the statistical properties of different signal classes vary in the frequency domain. These visualizations are essential for identifying distinguishing features that aid in the classification of structural events.

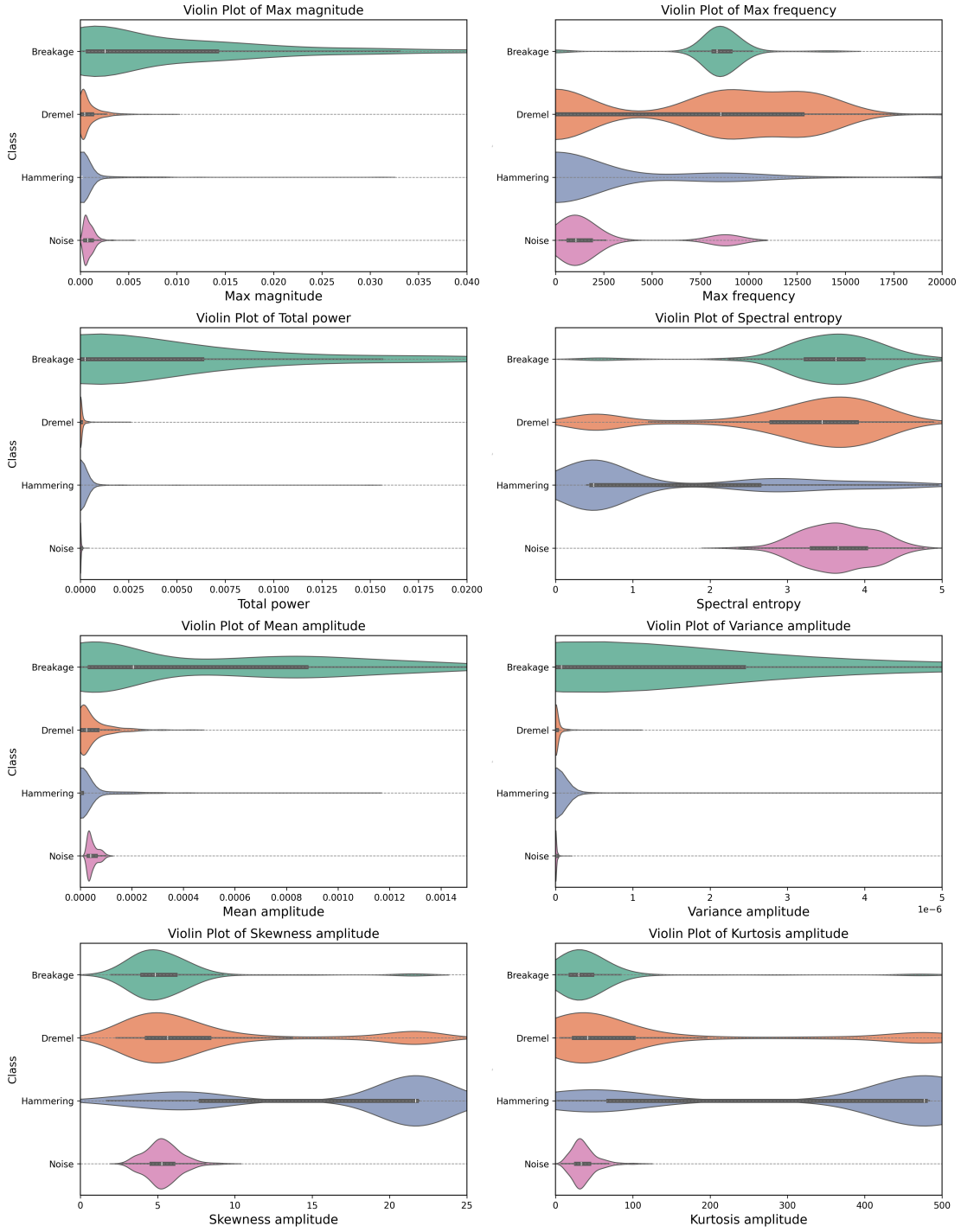



**Figure 5.6:** Violin plots of statistical parameters in the frequency domain for different signal classes.

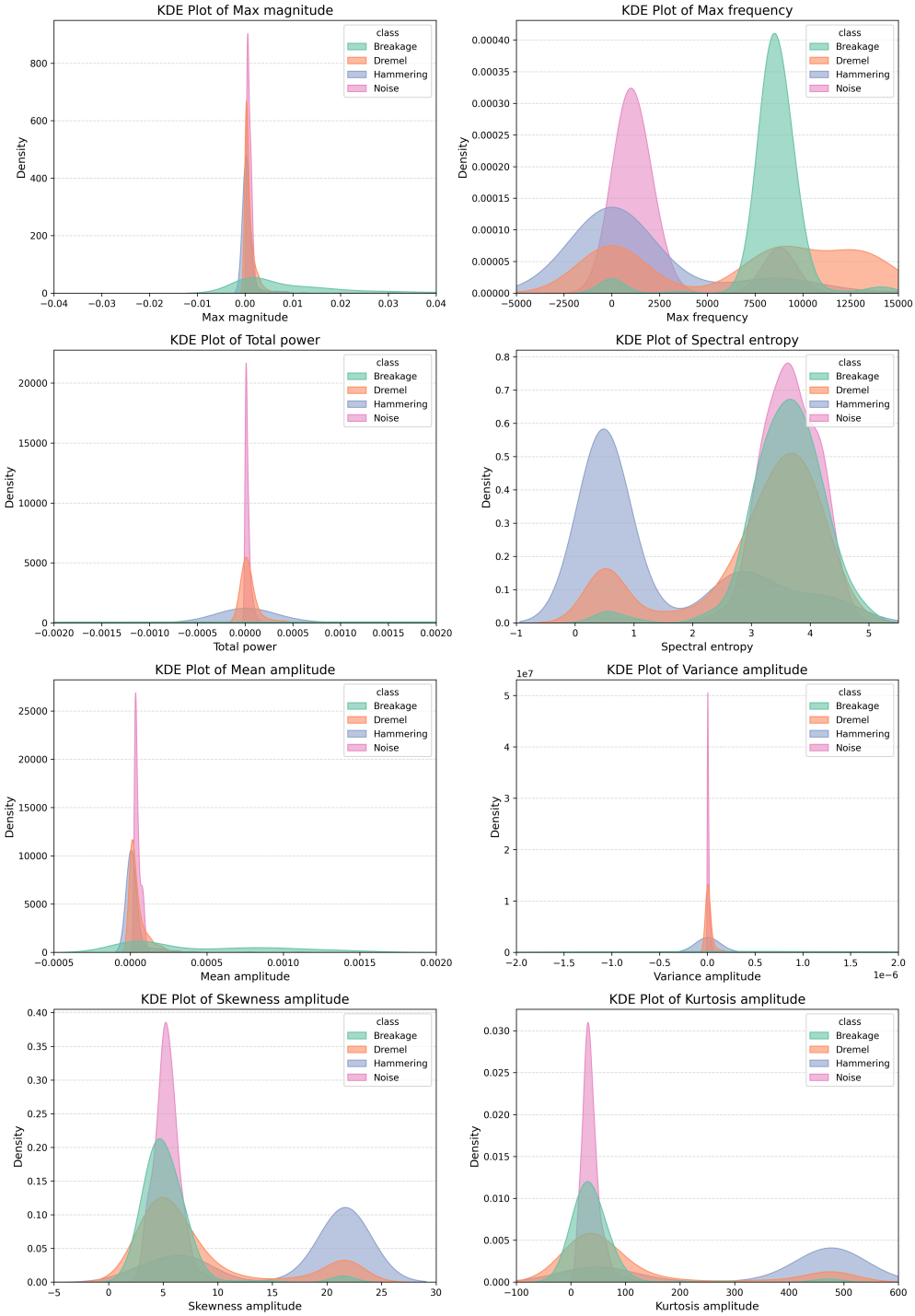



**Figure 5.7:** Kernel density estimate (KDE) plots of statistical parameters in the frequency domain for different signal classes.

## 5.3 Spectrograms Analysis

Spectrograms offer a powerful visual representation of how the frequency content of signals changes over time, making them an essential method for analyzing structural events in SHM. In this section, we detail the process of generating and analyzing spectrograms for each signal class. First, we applied the Short-Time Fourier Transform (STFT) to the signals to convert them from the time domain to the time-frequency domain. This transformation allows us to observe how the frequency components of a signal evolve over time, which is crucial for detecting and characterizing transient events in SHM.

### 5.3.1 Selection of Window Size in STFT

The choice of window size in the STFT significantly affects the balance between time and frequency resolution in the resulting spectrogram. To determine the optimal window size for our analysis, we experimented with four different window sizes: 64, 128, 256, and 512. Each window size presents a trade-off between time resolution and frequency resolution:

- **Window Size 64:** Provides high time resolution, capturing rapid changes in the signals, but with lower frequency resolution. As a result, the frequency components appear broader and less defined. This is especially evident in the breakage and hammering signals, where fine frequency details are blurred.
- **Window Size 128:** Achieves a balance between time and frequency resolution. This window size reveals the distinct spectral patterns in each signal class more effectively, offering clearer insights into the frequency components while still retaining sufficient time resolution for event detection.
- **Window Size 256:** Enhances frequency resolution at the cost of reduced time resolution. This size provides a more detailed view of the frequency content but introduces time smearing, making it harder to detect the precise onset of short-duration events, such as hammering impacts.
- **Window Size 512:** Maximizes frequency resolution, producing the sharpest frequency components. However, the trade-off is a significant reduction in time resolution, leading to time smearing that makes it difficult to capture transient events accurately. In breakage and hammering signals, important details about the event onset are lost due to the time smearing effect.

After evaluating the spectrograms produced with different window sizes, a window size of 128 was selected for further analysis. This choice provides a satisfactory balance between time and frequency resolution, capturing essential features of the

signals while maintaining adequate temporal detail for detecting structural events.

Figures 5.8, 5.9, 5.10, and 5.11 display spectrograms for two signals from each class with the four different window sizes applied. Each spectrogram's x-axis represents time (in seconds), the y-axis represents frequency (in Hz), and the color intensity corresponds to the signal's amplitude measured in decibels (dB).

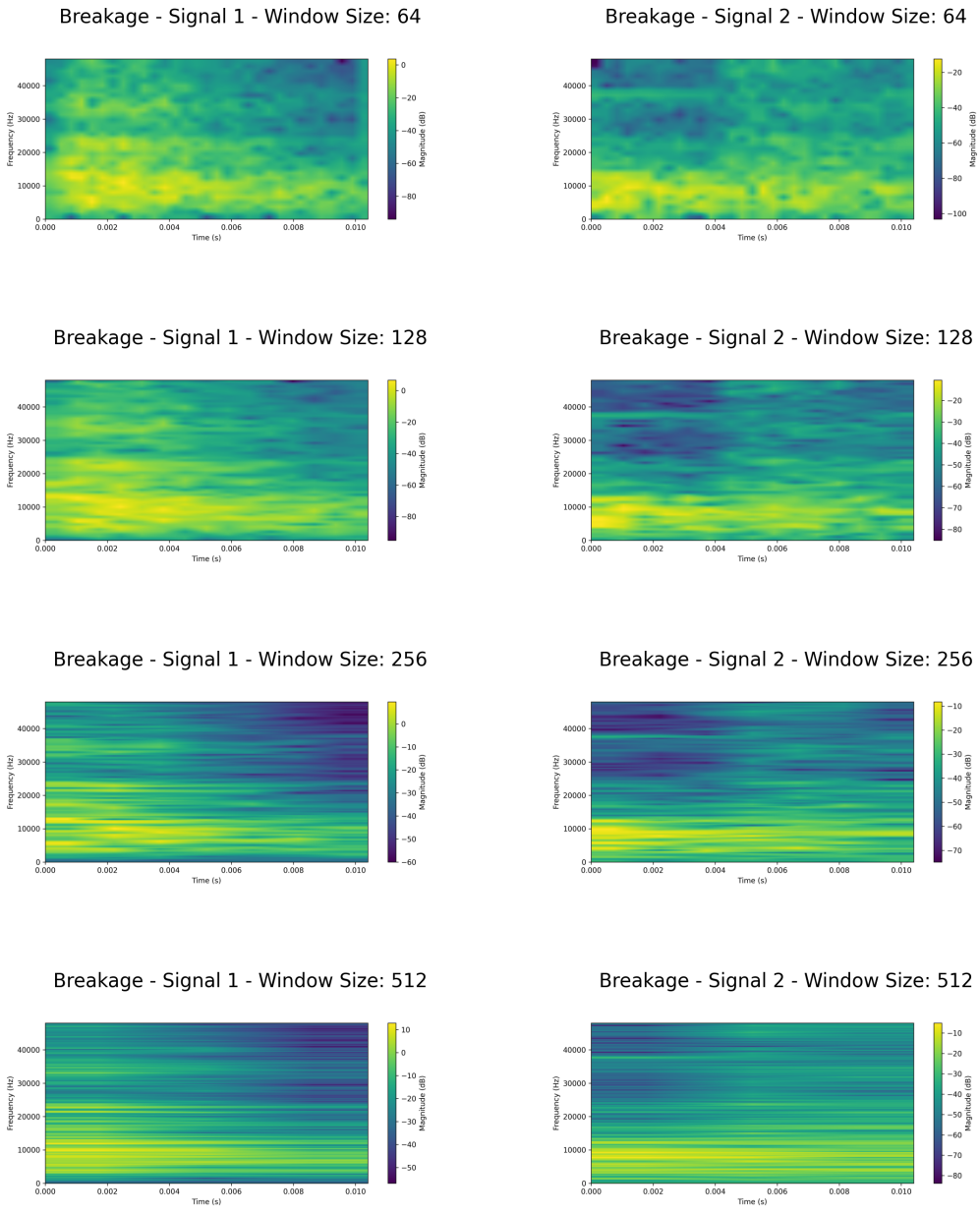



**Figure 5.8:** Spectrograms for breakage signals with different window sizes.

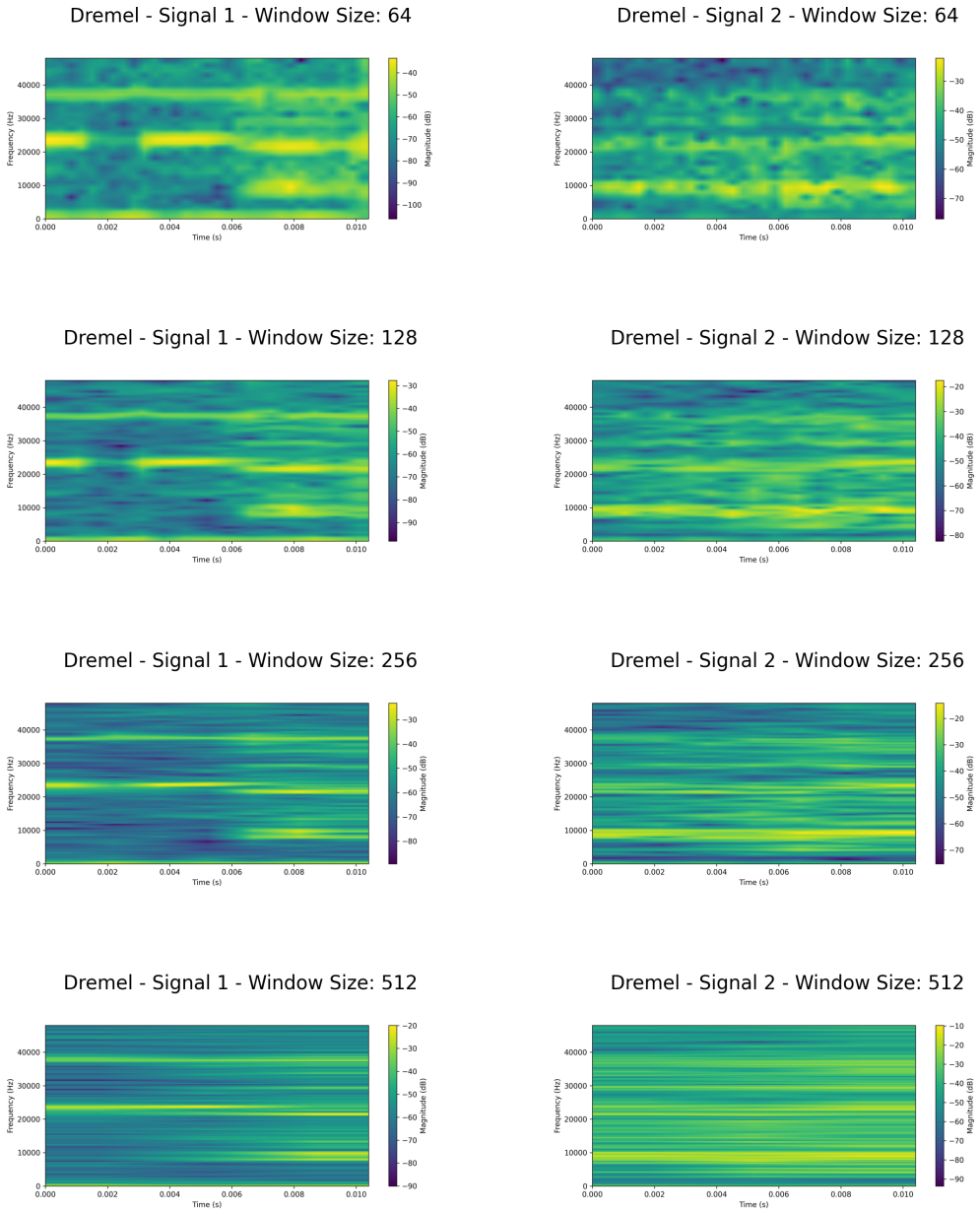



**Figure 5.9:** Spectrograms for dremel signals with different window sizes.

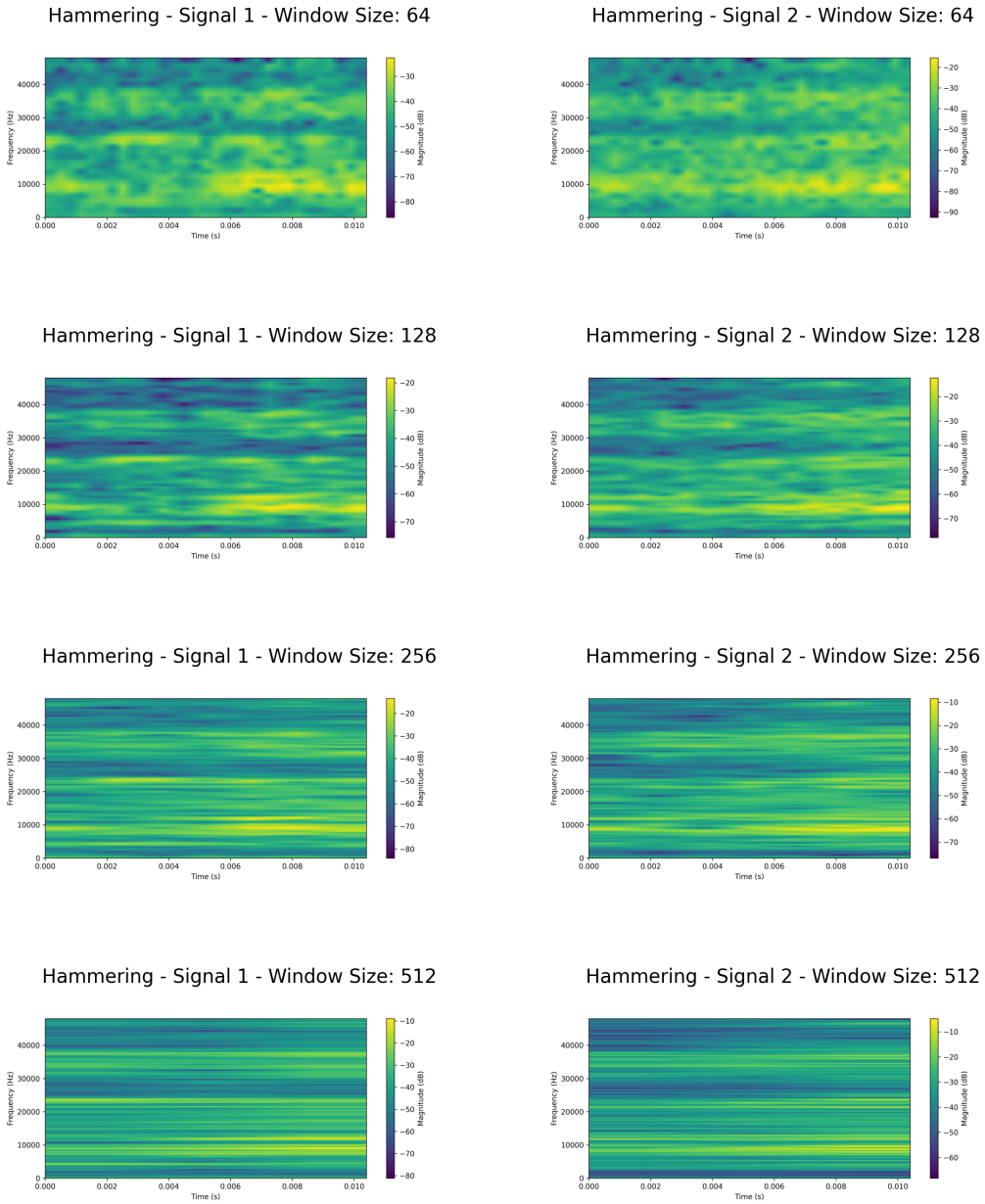



**Figure 5.10:** Spectrograms for hammering signals with different window sizes.

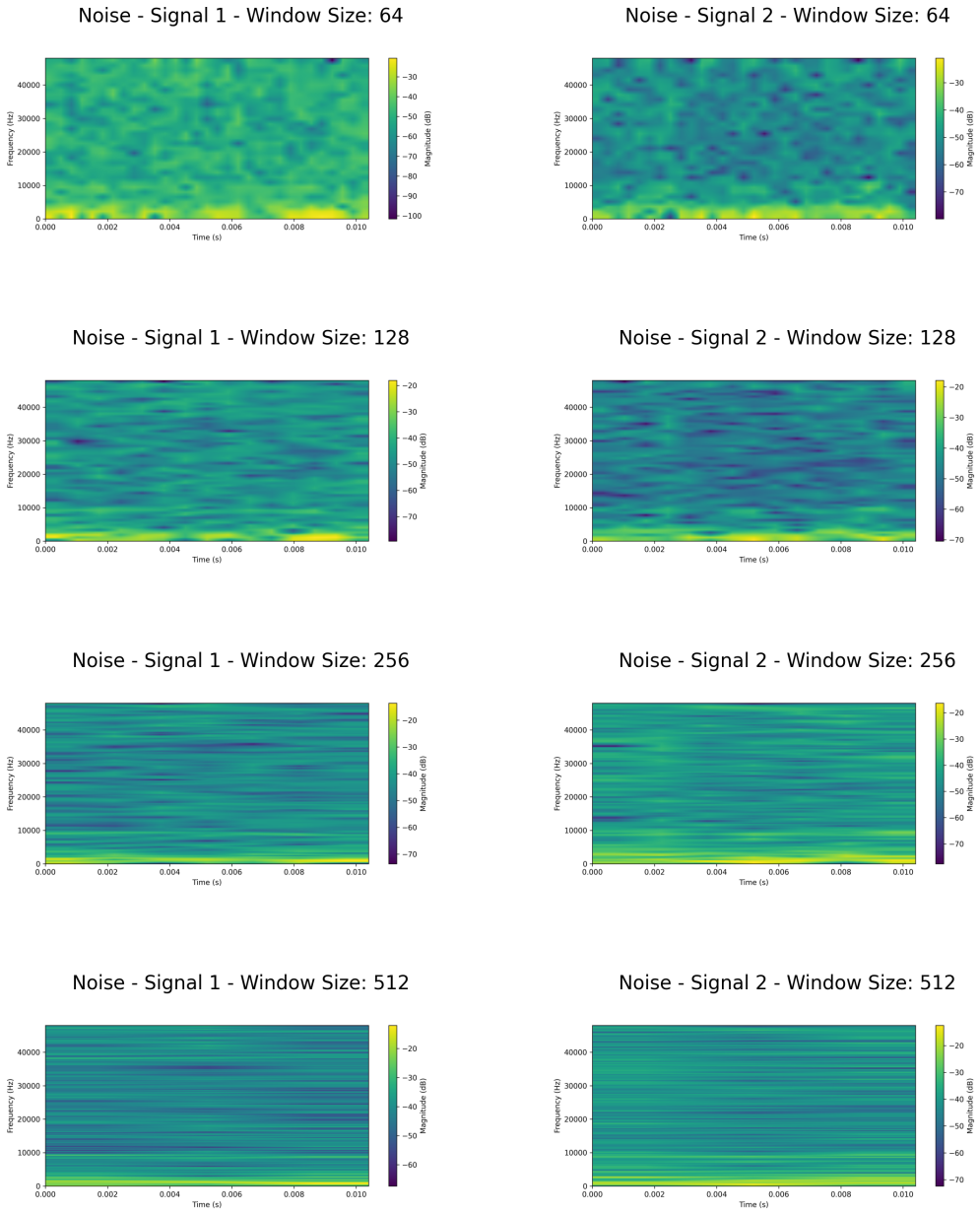



**Figure 5.11:** Spectrograms for noise signals with different window sizes.

### Spectrogram Extraction and Conversion to Numpy Arrays

With the STFT applied and the optimal window size determined, the next step involved extracting the spectrograms from the signals and converting them into NumPy arrays for further analysis and use in deep learning models. This process is essential for transforming the visual spectrogram data into a format suitable for

computational processing.

**STFT Parameters:** To extract meaningful information from the time-frequency domain, specific parameters were set:

- **Window size:** For this study, a window size of 128 was used, as determined from the previous analysis.
- **Overlap:** A 50% overlap between consecutive windows is used to enhance the temporal resolution without compromising the spectral detail.
- **Sampling rate:** The signals were sampled at a rate of 96000 Hz, allowing for the capture of fine details across a wide frequency range.

**Frequency Bins Calculation:** Using a window size of 128, the STFT produces $\frac{128}{2} + 1 = 65$ frequency bins, representing frequencies from 0 Hz to the Nyquist frequency. The Nyquist frequency, defined as half the sampling rate, is given by:

$$\text{Nyquist frequency} = \frac{\text{sampling rate}}{2} = \frac{96000}{2} = 48000\text{Hz} \tag{5.1}$$

Each frequency bin corresponds to a range of approximately 738.46 Hz, calculated as:

$$\text{Bin frequency range} = \frac{48000}{65} \approx 738.46, \text{Hz} \tag{5.2}$$

**Time Frames Calculation:** The number of time frames in the spectrogram depends on the signal length, window size, and hop size (which is half the window size due to 50% overlap):

$$\text{Hop size} = \frac{\text{window size}}{2} = \frac{128}{2} = 64 \tag{5.3}$$

The number of frames $N$ can then be calculated using the following formula:

$$N = 1 + \left\lceil \frac{\text{signal length} - \text{window size}}{\text{hop size}} \right\rceil \tag{5.4}$$

For example, with a signal length of 1000 samples:

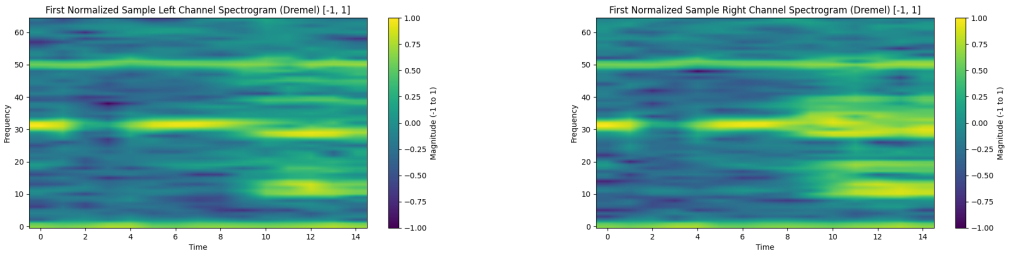
$$N = 1 + \left\lceil \frac{1000 - 128}{64} = 1 + 13.625 \approx 15\,\text{frames} \right\rceil \tag{5.5}$$

Thus, the resulting shape of each spectrogram array is approximately $(15, 65)$.

**Conversion to Numpy Arrays:** Once the STFT is computed, the resulting spectrogram is stored as a 2D array where each element corresponds to the magnitude of a specific frequency bin at a given time frame. These 2D arrays are saved in the ‘.npy‘ format, which is efficient for further processing and input to DL models. The arrays are normalized to the range [-1,1] to standardize the data and facilitate training in neural networks.

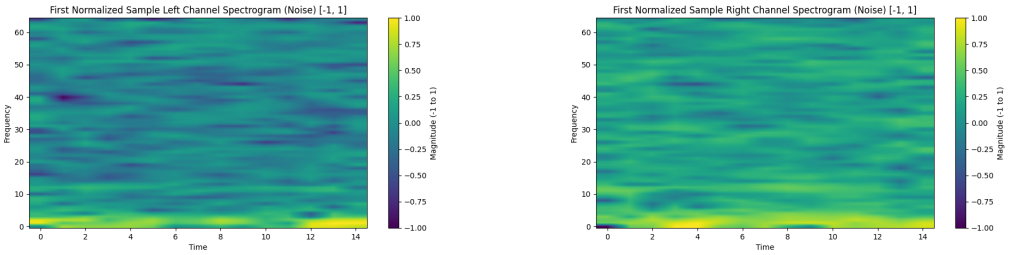
### 5.3.2 Normalization and Visualization of Spectrograms

To prepare the spectrogram data for input into deep learning models, we normalized the NumPy arrays to the range [-1, 1]. This normalization ensures consistency across all signal classes and facilitates more stable and efficient training of neural networks.
Figures 5.12, 5.13, 5.14, and 5.15 display examples of the normalized spectrograms for each signal class. These visualizations highlight how the frequency content has been uniformly scaled while preserving the essential characteristics of the original signals.



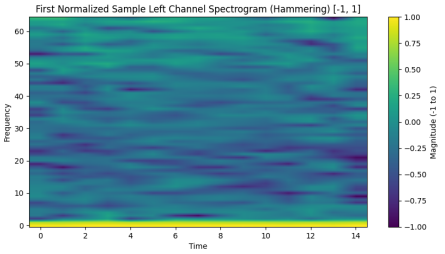

**(a)** Left channel spectrogram. **(b)** Right channel spectrogram.

**Figure 5.12:** Normalized spectrograms for the Dremel signal class.

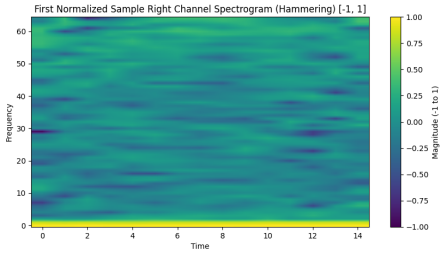



**(a)** Left channel spectrogram. **(b)** Right channel spectrogram.

**Figure 5.13:** Normalized spectrograms for the Noise signal class.

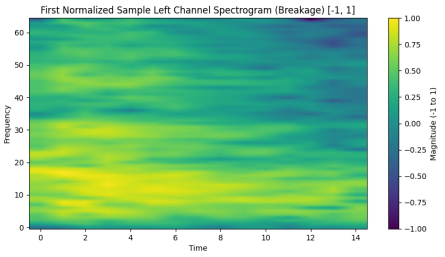



**(a)** Left channel spectrogram.

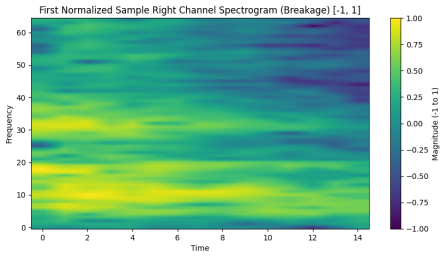



**(b)** Right channel spectrogram.

**Figure 5.14:** Normalized spectrograms for the Hammering signal class.




**(a)** Left channel spectrogram.




**(b)** Right channel spectrogram.

**Figure 5.15:** Normalized spectrograms for the Breakage signal class.

## 5.4 Generative Adversarial Networks Results

This section presents the results obtained from the three types of GANs used in this research: DCGAN, WGAN, and LSGAN. Each model was trained using the synthetic data generated from the STFT of the SHM signals, focusing specifically on the Dremel class. The generated spectrograms were evaluated based on their visual quality and the corresponding loss functions.

To enhance the training process and increase the robustness of the GANs, two types of noise were employed during training. The first type is the latent noise used as the input to the generator, denoted as $\mathbf{z}$. This latent vector $\mathbf{z} \in \mathbb{R}^{100}$ is sampled from a uniform distribution and serves as the starting point for generating synthetic data. The randomness introduced by $\mathbf{z}$ is crucial for producing diverse outputs from the generator. The second type of noise is Gaussian noise, denoted as $\mathcal{N}(0, 0.1)$, which was added to the real samples before they were input into the discriminator. This Gaussian noise, with a mean of 0 and a standard deviation of 0.1, serves several purposes. By introducing small random variations to the real

data, it creates a more realistic learning environment that mimics sensor fluctuations or environmental changes commonly found in real-world SHM applications. This helps the discriminator avoid overfitting by learning general patterns rather than memorizing the exact details of the real samples. Moreover, the addition of Gaussian noise mitigates mode collapse by ensuring greater variability in the input data, thus improving the overall robustness and generalization ability of the GAN. Together, the latent vector $\mathbf{z}$ as the input for the generator and the Gaussian noise $\mathcal{N}(0, 0.1)$ added to the discriminator inputs enable the GAN to capture the variability found in real SHM scenarios more effectively. This results in the generation of synthetic data that better reflects the diversity and quality of real-world SHM conditions.

The following subsections detail the performance of each GAN variant, outlining their respective advantages and challenges. Visual outputs from the generators, alongside the training dynamics of both discriminator and generator, are also presented to offer a clearer understanding of each model's effectiveness in generating realistic spectrograms for SHM tasks. The analysis of GAN-generated samples aims to determine which model best replicates the distinct characteristics of breakage, hammering, dremel, and noise signals. A comprehensive comparison will then be conducted, focusing on visual quality and the quantitative FID metric.

### 5.4.1 Deep Convolutional Generative Adversarial Networks Results

This section presents the architecture, training setup, and performance analysis of the DCGAN model, focusing on its ability to generate synthetic spectrograms for the Dremel class.

#### DCGAN Model Architecture

The DCGAN model consists of two main components: the generator and the discriminator. Each component's architecture is designed to progressively refine features, with specific layers dedicated to enhancing stability and generalization. Tables 5.2 and 5.3 provide a detailed breakdown of each layer, including output shape and parameter count.

The generator architecture, as shown in Table 5.2), is configured to upsample an input noise vector into a structured spectrogram resembling the original Dremel signal. Starting with a dense layer, the network reshapes the noise vector and uses convolutional layers with batch normalization, ReLU activations, and dropout to build high-quality, realistic signal features.

**Table 5.2:** Generator Architecture for the DCGAN

| **Layer (type)** | **Output Shape** | **Param #** |
|---|---|---|
| Input Layer | (None, 100) | 0 |
| Dense | (None, 33792) | 3,412,992 |
| Reshape | (None, 8, 33, 128) | 0 |
| Batch Normalization | (None, 8, 33, 128) | 512 |
| ReLU | (None, 8, 33, 128) | 0 |
| UpSampling2D | (None, 16, 66, 128) | 0 |
| Conv2D | (None, 16, 66, 128) | 147,584 |
| Batch Normalization | (None, 16, 66, 128) | 512 |
| Conv2D | (None, 16, 66, 128) | 147,584 |
| Batch Normalization | (None, 16, 66, 128) | 512 |
| Dropout | (None, 16, 66, 128) | 0 |
| Add | (None, 16, 66, 128) | 0 |
| Conv2D | (None, 15, 65, 64) | 32,832 |
| Batch Normalization | (None, 15, 65, 64) | 256 |
| Conv2D | (None, 15, 65, 1) | 577 |
| **Total** | **-** | **3,743,361** |

The discriminator, as depicted in Table 5.3, evaluates the authenticity of spectrograms by progressively downsampling the input and extracting hierarchical features through a series of convolutional, batch normalization, and dropout layers. The final dense layer outputs a binary classification indicating real or generated data.

**Table 5.3:** Discriminator Architecture for the DCGAN

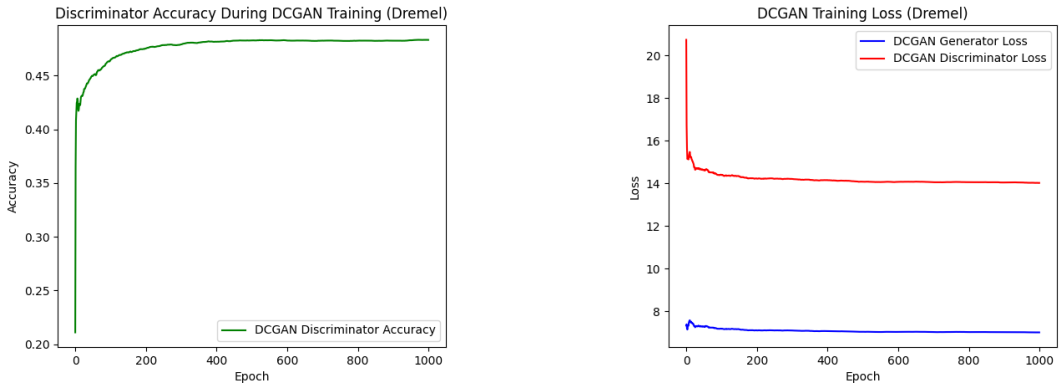
| **Layer (type)** | **Output Shape** | **Param #** |
|---|---|---|
| Conv2D | (None, 8, 33, 64) | 640 |
| Batch Normalization | (None, 8, 33, 64) | 256 |
| Dropout | (None, 8, 33, 64) | 0 |
| Conv2D | (None, 4, 17, 128) | 73,856 |
| Batch Normalization | (None, 4, 17, 128) | 512 |
| Dropout | (None, 4, 17, 128) | 0 |
| Conv2D | (None, 2, 9, 256) | 295,168 |
| Batch Normalization | (None, 2, 9, 256) | 1,024 |
| Dropout | (None, 2, 9, 256) | 0 |
| Conv2D | (None, 1, 5, 512) | 1,180,160 |
| Batch Normalization | (None, 1, 5, 512) | 2,048 |
| Dropout | (None, 1, 5, 512) | 0 |
| Flatten | (None, 2560) | 0 |
| Dense | (None, 1) | 2,561 |
| **Total** | **-** | **1,556,225** |

### Performance Analysis

The performance of the DCGAN model was assessed through two primary metrics: discriminator accuracy and the loss values for both the generator and discriminator over 1000 epochs.

Figure 5.16 illustrates the evolution of the discriminator's accuracy and the loss functions of both networks. Initially, the discriminator's accuracy increases rapidly, stabilizing around 0.45 after 200 epochs, indicating that it effectively differentiates real from generated samples up to a certain level. The generator's loss shows a steady decrease, suggesting that its ability to create realistic samples improves over time, while the discriminator's loss plateaus, indicating stable learning dynamics.

These observations highlight the dynamics between the generator and discriminator during training, with both networks adjusting to each other and reaching a relatively stable state as training progresses.




**(a)** Discriminator accuracy during training.

**(b)** Training loss of the generator and discriminator.

**Figure 5.16:** Results of DCGAN training for the Dremel class.

### Training Instability and Observations

Despite extending the training to 1000 epochs, the DCGAN struggled to generate fully realistic spectrograms, as shown in Figure 5.17. The generated samples exhibit some structural features, but they fall short in replicating the distinct characteristics of real signals, such as sharp frequency peaks or coherent frequency bands. This limitation is largely attributed to the model's struggle with mode collapse, resulting in limited diversity in outputs and repetitiveness in generated samples.

The discriminator's accuracy plateaued at around 50%, suggesting that it became unable to effectively distinguish between real and synthetic samples as training progressed. This stagnation indicates that the generator and discriminator reached an equilibrium where neither network improved significantly, reflecting a need for a more robust architecture or different training strategies to overcome these challenges.

In conclusion, while the DCGAN demonstrated some capacity to learn and replicate general patterns, the instability in training and the limited fidelity of the generated spectrograms highlight the necessity for exploring alternative GAN models. The following sections will present the performance of WGAN and LSGAN, which employ modified architectures and training techniques to address the issues observed with the DCGAN.

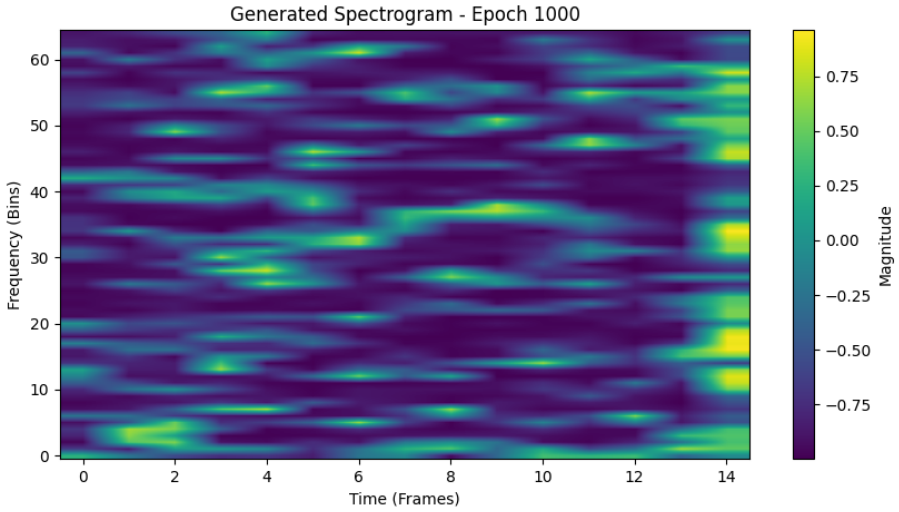



**Figure 5.17:** Spectrogram generated by the DCGAN after 1000 epochs, that lacks the distinctive features of real spectrograms, such as well-defined frequency bands and sharp transitions, highlighting the limitations of the current model.

### 5.4.2 Wasserstein Generative Adversarial Networks Results

This section provides an overview of the architecture, training behavior, and performance of the Wasserstein GAN (WGAN) applied to the Dremel class, over 1000 epochs. The WGAN model showed improved stability and generated higher quality samples compared to the DCGAN.

### WGAN Model Architecture

To clarify the architectural details of the WGAN utilized in this study, Tables 5.4 and 5.5 present the layer-by-layer composition of both the generator and discriminator models. Each table outlines the output shape and parameter count for each layer, providing insight into the model's structure.

The generator architecture, as shown in Table 5.4, begins with a dense layer that reshapes the input noise vector into a higher-dimensional representation. This is followed by a series of convolutional, batch normalization, and ReLU layers, which facilitate feature learning and upsampling. The architecture's design aims to gradually increase the spatial dimensions of the synthetic spectrogram, allowing it to capture and refine the essential features associated with the target signal class. Dropout layers are incorporated to enhance training stability by reducing overfitting.

**Table 5.4:** Generator Architecture for the WGAN.

| **Layer (type)** | **Output Shape** | **Param #** |
|---|---|---|
| Input Layer | (None, 100) | 0 |
| Dense | (None, 33792) | 3,412,992 |
| Reshape | (None, 8, 33, 128) | 0 |
| Batch Normalization | (None, 8, 33, 128) | 512 |
| ReLU | (None, 8, 33, 128) | 0 |
| UpSampling2D | (None, 16, 66, 128) | 0 |
| Conv2D | (None, 16, 66, 128) | 147,584 |
| Batch Normalization | (None, 16, 66, 128) | 512 |
| ReLU | (None, 16, 66, 128) | 0 |
| Dropout | (None, 16, 66, 128) | 0 |
| Conv2D | (None, 16, 66, 128) | 147,584 |
| Batch Normalization | (None, 16, 66, 128) | 512 |
| ReLU | (None, 16, 66, 128) | 0 |
| Conv2D | (None, 15, 65, 64) | 32,832 |
| Batch Normalization | (None, 15, 65, 64) | 256 |
| Conv2D | (None, 15, 65, 1) | 577 |
| **Total** | **-** | **3,742,593** |

The discriminator architecture, as depicted in Table 5.5, includes convolutional and LeakyReLU layers that progressively reduce the spatial dimensions of the input, enabling it to extract hierarchical features from the spectrograms. Dropout layers are used to improve the model's generalization ability, while the final dense layer outputs a single value representing the Wasserstein distance, which guides the generator's training process.

**Table 5.5:** Discriminator Architecture for the WGAN.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| Conv2D | (None, 8, 33, 64) | 640 |
| LeakyReLU | (None, 8, 33, 64) | 0 |
| Dropout | (None, 8, 33, 64) | 0 |
| Conv2D | (None, 4, 17, 128) | 73,856 |
| LeakyReLU | (None, 4, 17, 128) | 0 |
| Dropout | (None, 4, 17, 128) | 0 |
| Flatten | (None, 8704) | 0 |
| Dense | (None, 1) | 8,705 |
| **Total** | **-** | **83,201** |

### Performance Analysis

The WGAN demonstrated stable training dynamics compared to the DCGAN, as seen in the generator and discriminator loss trends over 1000 epochs. The discriminator's loss, fluctuating around negative values, indicates consistent learning driven by the Wasserstein distance. Figure 5.18 shows the progression of both losses, revealing a dynamic interplay that contributed to high-quality synthetic samples. The generator's loss became increasingly negative over time, reflecting its improvement in producing realistic data.

Throughout training, the generator and discriminator losses evolved as follows:

- **Early stages**: The generator's loss was initially low (e.g., -0.0019 at epoch 10), while the discriminator's was high (e.g., 7.63), indicating an initial adjustment phase.
- **Middle stages**: The generator's loss turned more negative (e.g., -1.78 at epoch 170), with the discriminator's loss also decreasing (around -6.49), showing progress in sample realism.
- **Late stages**: By epoch 1000, the generator's loss reached -7.98, while the discriminator's loss stabilized around -6.76, suggesting refined outputs with minor stability challenges.

After 1000 epochs, the WGAN produced spectrograms with clearer structures and reduced noise, as illustrated in Figure 5.19. These spectrograms exhibit distinct patterns and higher fidelity, indicating that the WGAN effectively captured the essential characteristics of the Dremel class. While minor artifacts and noise remain, the quality and structure of the generated samples improved substantially over earlier epochs.

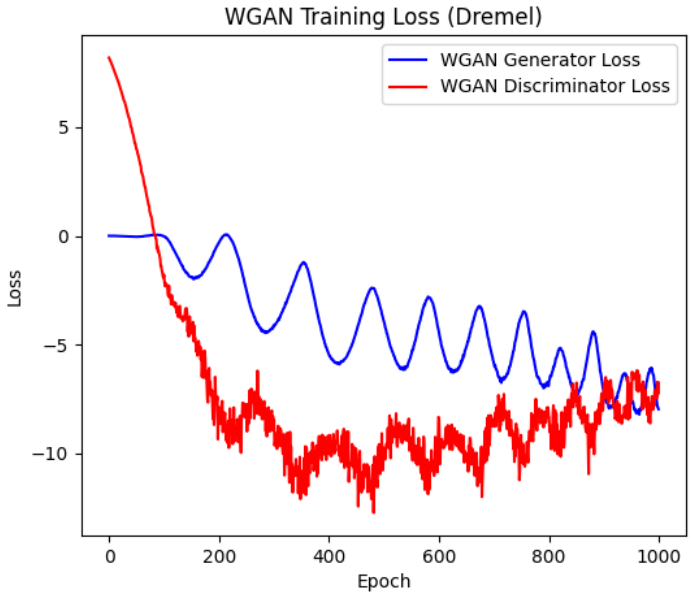



**Figure 5.18:** Discriminator and Generator Loss during WGAN training over 1000 epochs for the Dremel class. The discriminator's loss shows a consistent decrease, while the generator's loss fluctuates and stabilizes as training progresses.

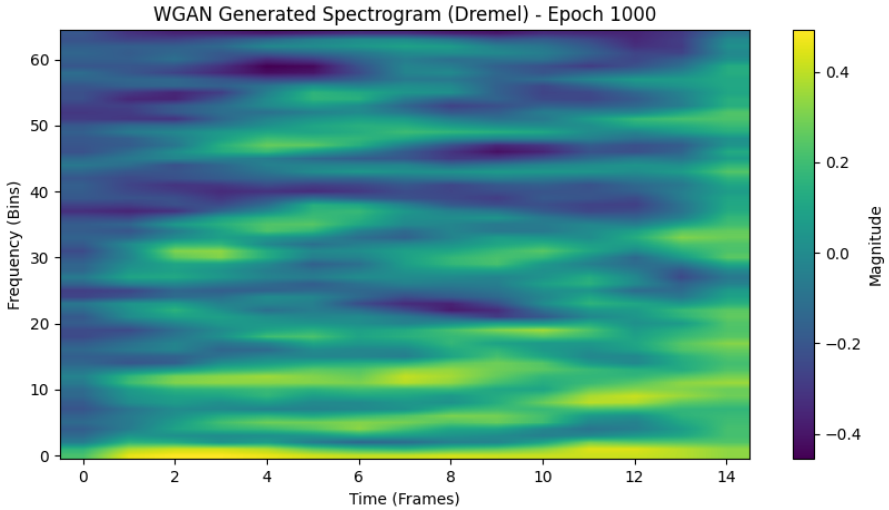
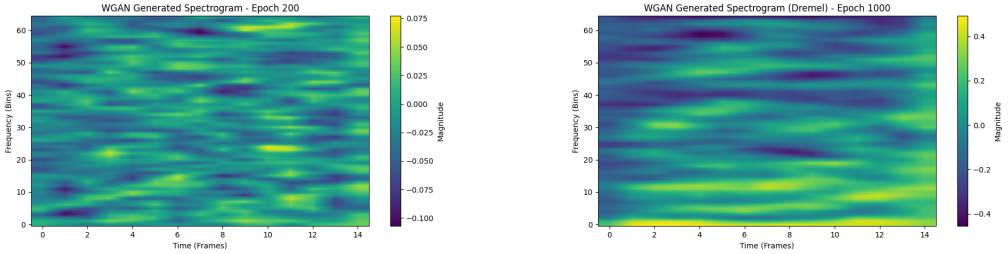



**Figure 5.19:** Sample of a synthetic spectrogram generated by the WGAN after 1000 epochs for the Dremel class. The generated spectrogram shows more structured patterns and reduced noise.

### Training Stability and Observations

Throughout the training, the WGAN displayed greater stability than the DCGAN, avoiding significant mode collapse and achieving a steady progression in sample quality. The generated spectrograms after 1000 epochs exhibited clearer and more accurate representations of the Dremel class, showing that the Wasserstein distance and regular updates to the discriminator contributed to a robust training process. As illustrated in Figure 5.20, the quality of the generated spectrograms improved noticeably from epoch 100 to epoch 1000, with later samples featuring well-defined patterns and reduced noise.

As shown in the spectrograms, the WGAN at 1000 epochs significantly improves upon the results at 100 epochs. The generated samples exhibit much clearer structure, and the noise levels are greatly reduced, though some artifacts still remain. These improvements suggest that the WGAN is more effective than the DCGAN for this dataset. In conclusion, the WGAN demonstrated superior stability and image quality compared to the DCGAN, underscoring the importance of the Wasserstein distance and frequent discriminator updates for enhancing model performance.




**(a)** WGAN Generated spectrogram at 100 epochs.

**(b)** WGAN Generated spectrogram at 1000 epochs.

**Figure 5.20:** Comparison of spectrograms generated by WGAN at 100 and 1000 epochs.

### 5.4.3 Least Squares Generative Adversarial Networks Results

This section provides an overview of the architecture, training behavior, and performance of the Least Squares GAN (LSGAN) applied to the Dremel class over 1000 epochs. The LSGAN model demonstrated greater stability and produced higher quality samples compared to both the DCGAN and WGAN.

### LSGAN Model Architecture

To provide a comprehensive understanding of the LSGAN architecture, Tables 5.6 and 5.7 detail the structures of the generator and discriminator models. Each table includes the layer composition, output shapes, and parameter counts.

The generator architecture, shown in Table 5.6, starts with a dense layer that expands the noise input into a high-dimensional representation, followed by transposed convolutional, batch normalization, and LeakyReLU layers. This setup gradually upsamples the noise, creating a synthetic spectrogram with characteristics of the Dremel class. Batch normalization stabilizes training, while LeakyReLU activations help capture essential patterns.

**Table 5.6:** Generator Architecture for the LSGAN.

| **Layer (type)** | **Output Shape** | **Param #** |
|---|---|---|
| Dense | (None, 62400) | 6,302,400 |
| LeakyReLU | (None, 62400) | 0 |
| Reshape | (None, 15, 65, 64) | 0 |
| Conv2DTranspose | (None, 15, 65, 64) | 36,928 |
| LeakyReLU | (None, 15, 65, 64) | 0 |
| BatchNormalization | (None, 15, 65, 64) | 256 |
| Conv2DTranspose | (None, 15, 65, 1) | 577 |
| **Total** | **-** | **6,340,161** |

The discriminator, as depicted in Table 5.7, classifies real and synthetic samples through convolutional and LeakyReLU layers, with Dropout layers for regularization. It reduces input dimensions progressively, extracting hierarchical features that aid in distinguishing real from generated spectrograms. The final dense layer outputs a single value, guiding the generator to produce samples that resemble real data.

**Table 5.7:** Discriminator Architecture for the LSGAN.

| **Layer (type)** | **Output Shape** | **Param #** |
|---|---|---|
| Conv2D | (None, 8, 33, 64) | 640 |
| LeakyReLU | (None, 8, 33, 64) | 0 |
| Dropout | (None, 8, 33, 64) | 0 |
| Conv2D | (None, 4, 17, 128) | 73,856 |
| LeakyReLU | (None, 4, 17, 128) | 0 |
| Dropout | (None, 4, 17, 128) | 0 |
| Flatten | (None, 8704) | 0 |
| Dense | (None, 1) | 8,705 |
| **Total** | **-** | **83,201** |

### Performance Analysis

The LSGAN's training on the Dremel class data over 1000 epochs demonstrated a stable and effective learning process, with clear improvements in model performance as training progressed. Initially, the discriminator's loss was notably higher than the generator's, reflecting the early stages where both networks were adjusting to distinguish between real and generated samples. This phase exhibited significant fluctuations as both networks learned to adapt. As training advanced, the generator started producing more realistic samples, resulting in a gradual decrease in its loss. Meanwhile, the discriminator maintained relatively stable loss values, indicating effective adaptation to both real and synthetic data. This balance between the networks enabled steady progress, with neither network excessively dominating the other. Towards the later stages, both generator and discriminator losses stabilized, with the discriminator's loss settling around 0.5 and the generator's around 0.25. These consistent loss values suggest the training achieved a balance, successfully avoiding issues like mode collapse. The stable loss curves also highlight the effectiveness of the least-squares loss function in LSGAN, promoting balanced training dynamics and ensuring high-quality spectrogram generation.

Figure 5.21 shows the evolution of loss values, highlighting minimal fluctuations as training converges.

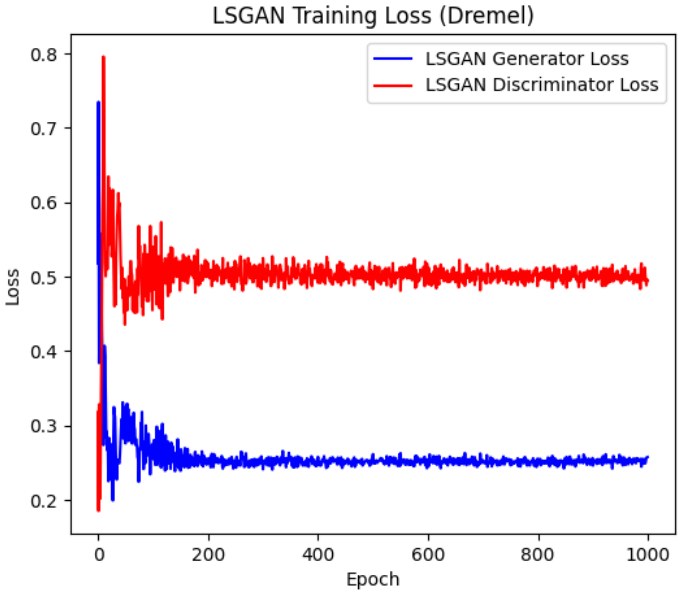



**Figure 5.21:** Training loss for the generator and discriminator during LSGAN training over 1000 epochs, showing stable learning with minimal fluctuations.

After 1000 epochs, the LSGAN generated synthetic spectrograms with improved structural clarity and reduced noise, as illustrated in Figure 5.22. These samples demonstrate the model's ability to capture the characteristic frequency patterns of the Dremel class, including distinct frequency bands and coherent structures. Compared to the initial epochs, the spectrograms now display enhanced visual fidelity, reflecting the effectiveness of the LSGAN's architecture and loss function in producing high-quality synthetic data.

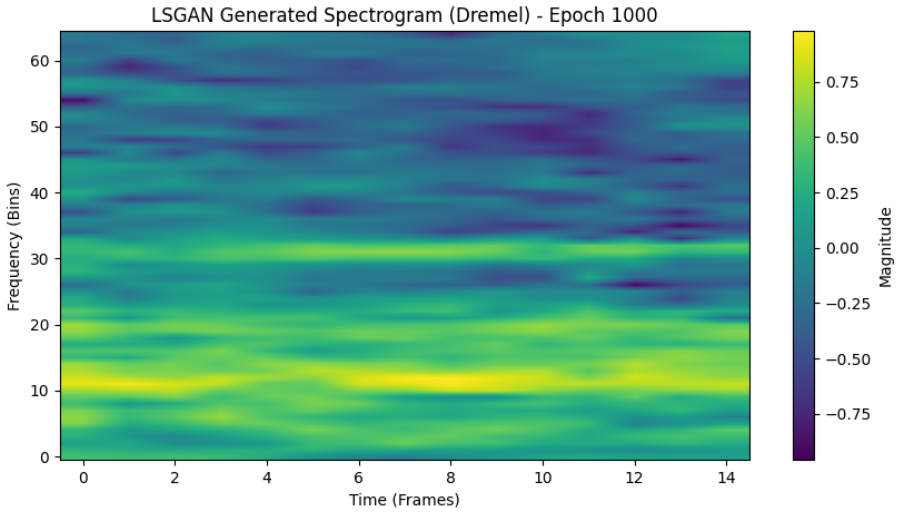



**Figure 5.22:** Sample spectrogram generated by the LSGAN after 1000 epochs, showing clear structure and well-defined frequency components.

### Training Stability and Observations

The LSGAN demonstrated stable training across 1000 epochs, maintaining consistent generator and discriminator performance. The addition of noise to the real samples contributed to robust learning, smoothing the loss curves and improving the model's resilience. The model effectively mitigated issues such as mode collapse, ensuring a diverse range of generated outputs that capture the essential characteristics of the Dremel class.

In conclusion, the LSGAN's least-squares loss function and robust training dynamics allowed it to produce high-quality spectrograms with minimal fluctuations in loss values. The generated samples displayed strong structural fidelity, making the LSGAN a promising model for generating realistic SHM data. Future refinements, such as further tuning of parameters or exploring architectural modifications, may enhance its performance even further.

## 5.5 Performance Comparison of DCGAN, WGAN and LSGAN

This section provides a comprehensive comparison between the DCGAN, WGAN, and LSGAN models, examining their training losses, FID scores, and final generated samples.

### 5.5.1 Training Losses

Figures 5.23, 5.24, 5.25, and 5.26 show the generator and discriminator losses over 1000 epochs for each model when trained on the Dremel, Breakage, Hammering, and Noise datasets, respectively.

The comparison highlights that the WGAN model exhibits smoother loss curves with significantly less fluctuation in both the generator and discriminator losses, particularly after the 400-epoch mark. This stability across epochs reflects the effectiveness of the Wasserstein distance in promoting gradual and steady learning, which is essential for generating high-quality samples. The WGAN's design, which updates the discriminator more frequently and minimizes oscillations, proves beneficial in maintaining training consistency, especially on complex datasets such as Dremel and Noise.
Conversely, the DCGAN displays more pronounced fluctuations and instability, with the discriminator loss often approaching values indicative of random guessing after extended training. This instability suggests that the DCGAN struggles with training balance and may suffer from issues such as mode collapse, where the generator fails to produce diverse samples.
The LSGAN maintains stable losses throughout the training, which is attributed to the least-squares loss function that promotes smoother gradients and helps avoid sharp fluctuations. This mechanism aids in preventing mode collapse and enhances the overall stability of the training process. The least-squares approach allows the LSGAN to generate samples that better capture the distribution of real data, making it a preferable choice in cases where consistent model performance is crucial.

Each figure includes a zoomed view of the LSGAN losses, providing a closer look at the model's behavior in the later stages of training. The zoomed-in view reveals slight fluctuations in the LSGAN's losses, especially as it approaches convergence. These minor oscillations are natural and minimal compared to the more significant fluctuations observed in the DCGAN and WGAN models. Overall, the stability of the LSGAN's losses, with generator and discriminator losses remaining relatively close and synchronized, underscores the effectiveness of the least-squares loss function. This approach not only stabilizes training but also ensures balanced

and consistent progress, reducing the likelihood of mode collapse.

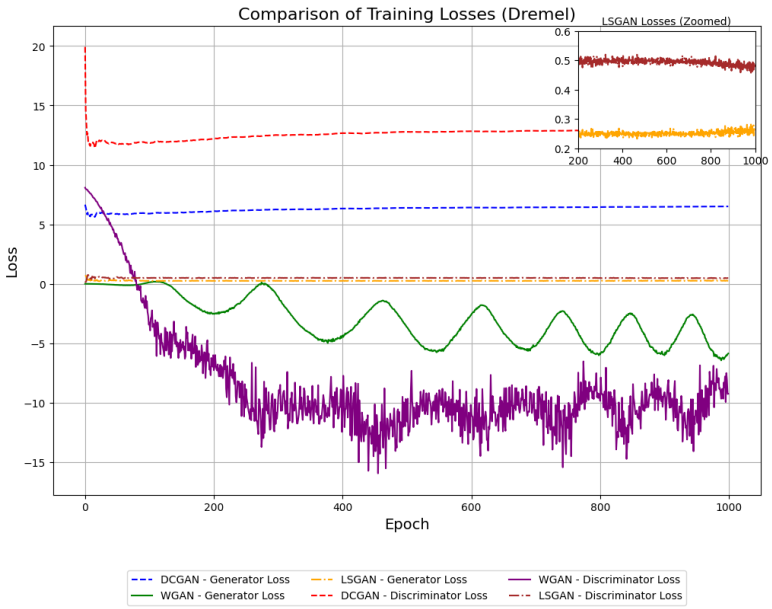



**Figure 5.23:** Comparison of generator and discriminator losses for DCGAN, WGAN, and LSGAN over 1000 epochs on the Dremel dataset (466 signals).

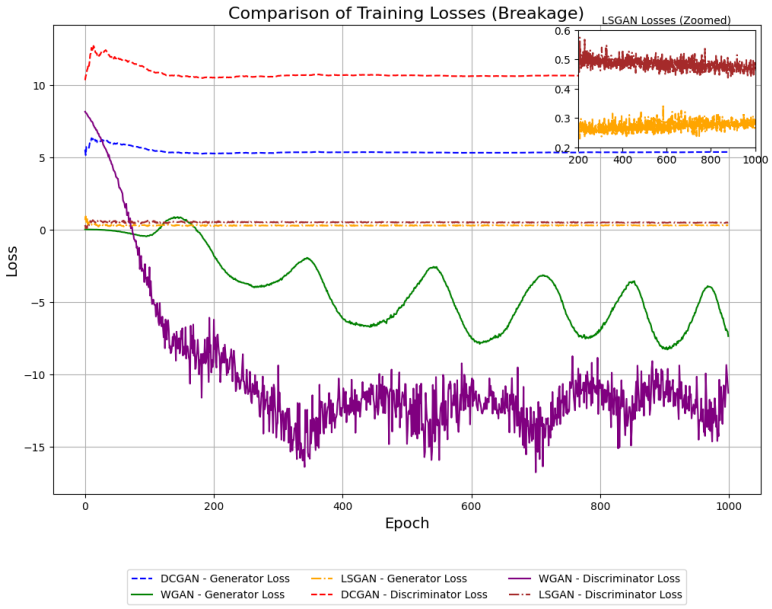



**Figure 5.24:** Comparison of generator and discriminator losses for DCGAN, WGAN, and LSGAN over 1000 epochs on the Breakage dataset (200 signals).

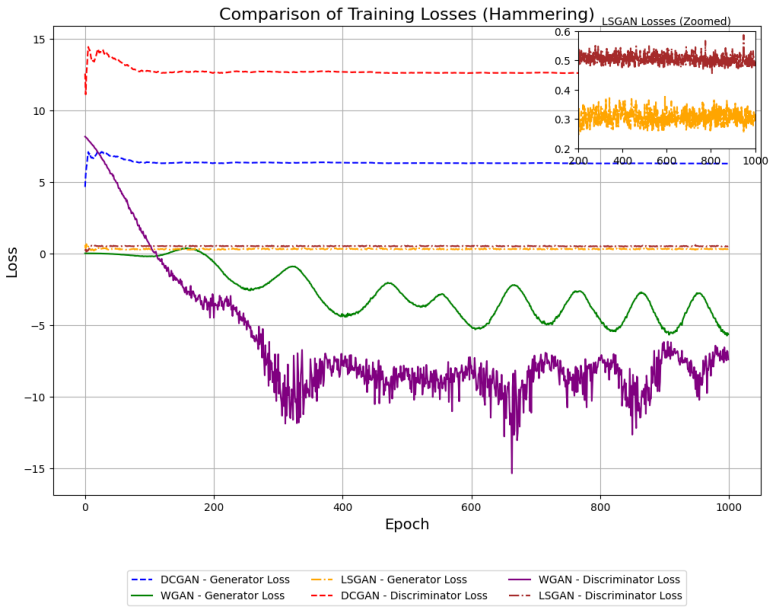



**Figure 5.25:** Comparison of generator and discriminator losses for DCGAN, WGAN, and LSGAN over 1000 epochs on the Hammering dataset (282 signals).

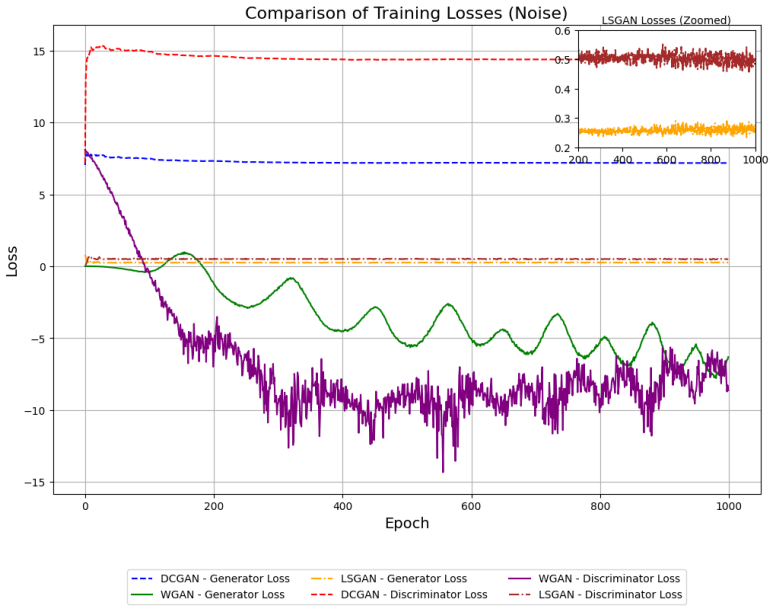



**Figure 5.26:** Comparison of generator and discriminator losses for DCGAN, WGAN, and LSGAN over 1000 epochs on the Noise dataset (400 signals).

### 5.5.2 Fréchet Inception Distance Score

Table 5.8 shows the FID scores for the three models on the Dremel, Breakage, Hammering, and Noise datasets, respectively. The results indicate that the DCGAN consistently produced the highest FID scores, suggesting that the generated data is far from realistic. In contrast, the WGAN achieved significantly lower FID scores, demonstrating an improvement in the quality of generated samples. The LSGAN achieved the lowest FID scores for all datasets, suggesting that it generated the most realistic samples among the three models. As explained in Section 3.3, the FID score provides an objective measure of how closely the generated data matches the real data by comparing their distributions in the feature space of a pre-trained network. Lower FID scores indicate a better match, meaning that the LSGAN was able to generate more realistic spectrograms compared to the other models.

**Table 5.8:** FID Scores for DCGAN, WGAN, and LSGAN on Dremel (466 signals), Breakage (200 signals), Hammering (282 signals), and Noise (400 signals) datasets. Lower scores indicate higher-quality generated data.

| Model | Dremel FID | Breakage FID | Hammering FID | Noise FID |
|---|---|---|---|---|
| DCGAN | 408.51 | 478.66 | 406.12 | 400.26 |
| WGAN | 77.61 | 89.41 | 84.54 | 59.39 |
| LSGAN | 30.33 | 39.52 | 34.33 | 32.66 |

The results clearly show that the LSGAN outperforms both the DCGAN and WGAN in terms of both training stability and image quality for all datasets. The LSGAN's least-squares loss function provides smoother gradients, which helps improve the generator's performance and produce more realistic spectrograms. The WGAN, while significantly better than the DCGAN, still suffers from minor fluctuations in the generator's loss, which may be improved with further tuning of the hyperparameters. The DCGAN, however, exhibits poor stability and generates lower-quality samples, indicating that it is less suitable for this specific task.

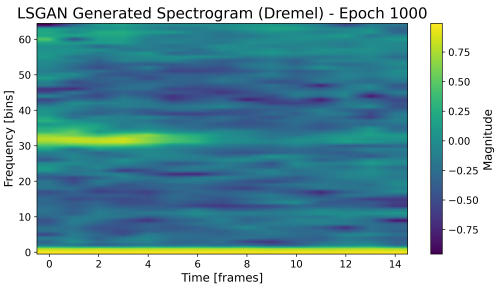
In conclusion, while both the WGAN and LSGAN offer significant improvements over the DCGAN, the LSGAN is the most effective model for generating high-quality spectrograms in this SHM application, as demonstrated by its superior FID scores and more stable training dynamics across different datasets.

### 5.5.3 Final Generated Samples

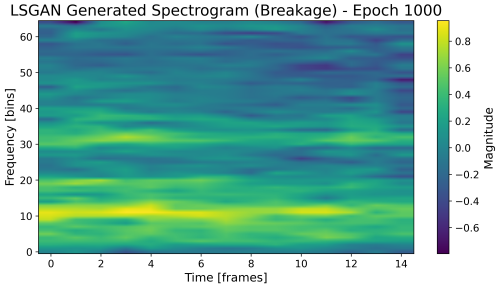
Figure 5.27 showcases examples of the final generated spectrograms from the LSGAN model at epoch 1000 for the Dremel, Breakage, Hammering, and Noise datasets. These samples demonstrate the model's ability to capture characteristic

frequency patterns, with distinct frequency bands and coherent structure. Compared to the initial training epochs, the spectrograms generated by the LSGAN exhibit higher visual fidelity, clearer structures, and reduced noise.

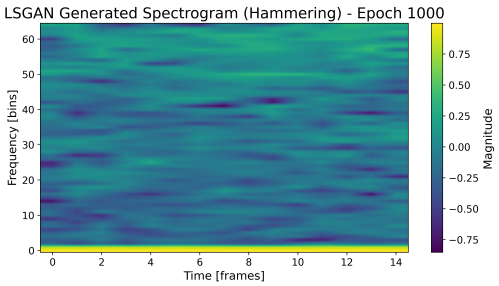
The stability observed in the LSGAN's training process, combined with its low FID scores, reflects its capability to produce realistic and high-quality synthetic samples. This makes the LSGAN a more suitable model than both the DCGAN and WGAN for SHM applications.



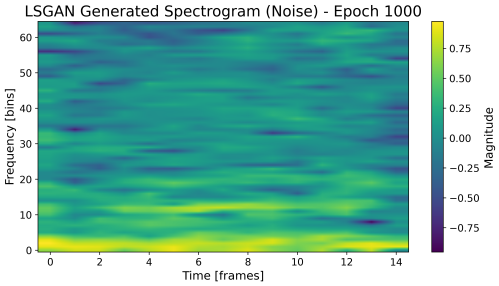

**(a)** LSGAN Generated Spectrogram (Dremel)




**(b)** LSGAN Generated Spectrogram (Breakage)




**(c)** LSGAN Generated Spectrogram (Hammering)




**(d)** LSGAN Generated Spectrogram (Noise)

**Figure 5.27:** Examples of LSGAN generated spectrograms at epoch 1000 for different signal classes: Dremel, Breakage, Hammering, and Noise.

# Chapter 6

# Conclusion and Future Developments

This thesis explored advanced methodologies in Structural Health Monitoring (SHM) to enhance the detection and prediction of structural anomalies in prestressed concrete bridges. The research focused on three main areas: signal representation techniques, data acquisition, and the application of deep generative models. Time-frequency analysis was implemented using the Short-Time Fourier Transform (STFT) to capture both temporal and spectral features of sensor signals. In-situ data collection was conducted on two prestressed concrete bridges, Le Pastena and Cerqueta, located on the A24 highway in the province of L'Aquila and scheduled for demolition. Using accelerometers and acoustic emission sensors, real-time data was collected during controlled destructive testing involving the cutting of prestressing wires, creating a valuable dataset for SHM.

The comparative analysis involved three types of Generative Adversarial Networks (GANs), Deep Convolutional GAN (DCGAN), Wasserstein GAN (WGAN), and Least Squares GAN (LSGAN), applied to generate synthetic spectrograms for data augmentation (DA) purposes. The LSGAN outperformed both the DCGAN and WGAN in generating high-quality spectrograms, demonstrating stable training dynamics and achieving the lowest Fréchet Inception Distance (FID) scores across all signal classes. This outcome suggests significant potential for LSGAN-generated synthetic data to improve SHM applications, particularly in scenarios where real-world data is limited.

The analysis of time and frequency domains revealed general patterns among the four signal classes (breakage, hammering, dremel, and noise). In the time

domain, signals showed varying amplitude trends, while frequency domain analysis identified distinct spectral profiles across classes. For instance, breakage signals tended to concentrate energy in lower frequencies, while dremel class signals displayed sharper peaks aligned with operational speed. Spectrogram analysis, optimized with a window size of 128, achieved a balanced representation, improving input quality for GAN performance. This approach facilitated deeper insights that support the development of automated systems for reliable anomaly detection.

This study contributes to SHM by analyzing the use of generative deep learning (GDL) models to create real-world synthetic datasets that can potentially improve the detection and classification of anomalies in structural monitoring. The LSGAN's capability to generate realistic synthetic data could address data scarcity, a critical limitation in SHM, allowing for better deep learning (DL) model training.

While the findings are promising, the research also identified challenges. Data scarcity, particularly for certain signal classes like breakage and hammering, limited model generalization. Computational demands associated with GAN training also constrained scalability. Additionally, data collection was conducted under specific conditions, potentially affecting model performance in varied environments.

To address these limitations, future research could focus on enhancing synthetic data quality to better mimic real-world failures through approaches like domain adaptation or transfer learning. Exploring alternative DL architectures could further improve data generation capabilities and prediction accuracy. Developing real-time SHM systems capable of on-the-fly data processing would enable predictive maintenance and timely interventions. Expanding datasets by collecting data from varied structures and environments, as well as collaborating with industry partners for data sharing and field validation, could improve model robustness and confirm their practical applicability. Moreover, integrating Explainable AI (XAI) techniques could enhance model transparency, allowing SHM operators to better understand decision-making processes. The use of unmanned aerial vehicles (Unmanned Aerial Vehicles (UAVs)s) equipped with IoT sensors also presents a promising advancement for monitoring hard-to-access areas, reducing reliance on manual inspections while improving safety and efficiency.

This thesis demonstrates the potential of combining signal processing and GDL in SHM. By replicating complex structural event patterns, these methods contribute to more accurate anomaly detection. The success of LSGAN in this research highlights the potential of GANs and generative models for data-scarce fields like SHM. Continued research and industry collaboration will advance SHM, ultimately supporting safer and more resilient infrastructure management, especially for bridges.

# Appendix A

# DCGAN Architecture

In this appendix, we present the architecture and code for the Deep Convolutional Generative Adversarial Network (DCGAN) implemented in Python using TensorFlow and Keras. This architecture is designed to generate synthetic spectrogram data for SHM applications.

## A.1 DCGAN Generator and Discriminator Code

**Listing A.1:** DCGAN Generator and Discriminator Architecture

```
def build_dcgan_generator(z_dim):
    model = models.Sequential()

    model.add(layers.Dense(8 * 33 * 128, kernel_initializer='
        he_normal', input_dim=z_dim))
    model.add(layers.Reshape((8, 33, 128)))
    model.add(layers.BatchNormalization())
    model.add(layers.ReLU())

    model.add(layers.UpSampling2D(size=(2, 2)))
    model.add(layers.Conv2D(128, kernel_size=3, padding="same",
        kernel_initializer='he_normal', activation='relu'))
    model.add(layers.BatchNormalization())

    model.add(layers.Conv2D(128, kernel_size=3, padding="same",
        kernel_initializer='he_normal', activation='relu'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dropout(0.3))
    model.add(layers.Add())

```

```
18      model.add(layers.Conv2D(64, kernel_size=(2, 2), strides=(1, 1),
            padding="valid", kernel_initializer='he_normal', activation='
            relu'))
19      model.add(layers.BatchNormalization())
20
21      model.add(layers.Conv2D(1, kernel_size=3, padding="same",
            activation="tanh"))
22
23      return model
```

## A.2 Training Configuration

**Listing A.2:** DCGAN Training Loop

```
1  def train_gan(epochs, batch_size, z_dim, gan_generator,
       dcgan_discriminator, gan, X_train, visualize_interval=100):
2      real = np.ones((batch_size, 1))
3      fake = np.zeros((batch_size, 1))
4      gan_metrics = {'generator_loss': [], 'discriminator_loss': [], '
           discriminator_accuracy': []}
5
6      for epoch in range(epochs):
7          idx = np.random.randint(0, X_train.shape[0], batch_size)
8          real_images = X_train[idx]
9          real_images += np.random.normal(0, 0.1, real_images.shape)
10
11         noise = np.random.normal(0, 1, (batch_size, z_dim))
12         fake_images = gan_generator.predict(noise)
13
14         d_loss_real, d_acc_real = dcgan_discriminator.train_on_batch(
               real_images, real)
15         d_loss_fake, d_acc_fake = dcgan_discriminator.train_on_batch(
               fake_images, fake)
16
17         d_loss = np.add(d_loss_real, d_loss_fake)
18         d_acc = np.add(d_acc_real, d_acc_fake) / 2
19
20         noise = np.random.normal(0, 1, (batch_size, z_dim))
21         g_loss = gan.train_on_batch(noise, real)
22
23         if isinstance(g_loss, list):
24             g_loss = g_loss[0]
25         if isinstance(d_loss, list):
26             d_loss = d_loss[0]
27
28         gan_metrics['generator_loss'].append(g_loss)
29         gan_metrics['discriminator_loss'].append(d_loss)
```

```
30          gan_metrics['discriminator_accuracy'].append(d_acc)
31
32          if (epoch + 1) % 10 == 0:
33              print(f'Epoch {epoch+1}/{epochs}, Generator Loss: {g_loss
                    :.4f}, Discriminator Loss: {d_loss:.4f}, Discriminator
                     Accuracy: {d_acc*100:.2f}%')
34
35      print("DCGAN training completed!")
36      return gan_metrics
```

# Appendix B

# WGAN Architecture

In this appendix, we present the architecture and code for the Wasserstein Generative Adversarial Network (WGAN) implemented in Python using TensorFlow and Keras. This implementation is used in the research for generating synthetic spectrogram data in SHM.

## B.1 WGAN Generator and Discriminator Code

**Listing B.1:** WGAN Generator and Discriminator Architecture

```
def build_wgan_generator(z_dim):
    model = models.Sequential()
    model.add(layers.Dense(8 * 33 * 128, activation="relu", input_dim
        =z_dim))
    model.add(layers.Reshape((8, 33, 128)))
    model.add(layers.UpSampling2D(size=(2, 2)))
    model.add(layers.Conv2D(128, kernel_size=3, padding="same"))
    model.add(layers.BatchNormalization())
    model.add(layers.ReLU())
    model.add(layers.Conv2D(128, kernel_size=3, padding="same"))
    model.add(layers.BatchNormalization())
    model.add(layers.ReLU())
    model.add(layers.Conv2D(64, kernel_size=(2, 2), strides=(1, 1),
        padding="valid"))
    model.add(layers.Conv2D(1, kernel_size=3, padding="same",
        activation="tanh"))
    return model

def build_wgan_discriminator(input_shape):
    model = models.Sequential()
    model.add(layers.Conv2D(64, kernel_size=3, strides=2, input_shape
        =input_shape, padding="same"))
```

```
    model.add(layers.LeakyReLU(alpha=0.2))
    model.add(layers.Dropout(0.25))
    model.add(layers.Conv2D(128, kernel_size=3, strides=2, padding="
        same"))
    model.add(layers.LeakyReLU(alpha=0.2))
    model.add(layers.Dropout(0.25))
    model.add(layers.Flatten())
    model.add(layers.Dense(1))
    return model
```

## B.2 Training Loop and Gradient Penalty

**Listing B.2:** WGAN Training Loop and Gradient Penalty

```
def gradient_penalty(wgan_discriminator, real_samples, fake_samples):
    batch_size = tf.shape(real_samples)[0]
    epsilon = tf.random.uniform([batch_size, 1, 1, 1], 0.0, 1.0)
    interpolated_samples = epsilon * real_samples + (1 - epsilon) *
        fake_samples
    with tf.GradientTape() as tape:
        tape.watch(interpolated_samples)
        interpolated_predictions = wgan_discriminator(
            interpolated_samples)
    gradients = tape.gradient(interpolated_predictions, [
        interpolated_samples])[0]
    grad_l2_norm = tf.sqrt(tf.reduce_sum(tf.square(gradients), axis
        =[1, 2, 3]))
    gradient_penalty = tf.reduce_mean((grad_l2_norm - 1.0) ** 2)
    return gradient_penalty

def train_wgan(wgan_generator, wgan_discriminator, X_train,
    batch_size, z_dim, epochs):
    wgan_generator_optimizer = tf.keras.optimizers.Adam(learning_rate
        =1e-4, beta_1=0.5, beta_2=0.9)
    discriminator_optimizer = tf.keras.optimizers.Adam(learning_rate
        =1e-4, beta_1=0.5, beta_2=0.9)
    for epoch in range(epochs):
        for _ in range(5):
            idx = np.random.randint(0, X_train.shape[0], batch_size)
            real_samples = X_train[idx]
            noise_real_samples = real_samples + np.random.normal(0,
                0.1, real_samples.shape)
            noise = tf.random.normal([batch_size, z_dim])
            fake_samples = wgan_generator(noise)
            with tf.GradientTape() as discriminator_tape:
                real_predictions = wgan_discriminator(
                    noise_real_samples)
```

```
25                fake_predictions = wgan_discriminator(fake_samples)
26                gp = gradient_penalty(wgan_discriminator,
                      noise_real_samples, fake_samples)
27                discriminator_loss = wgan_gp_loss(real_predictions,
                      fake_predictions, gp)
28            discriminator_gradients = discriminator_tape.gradient(
                  discriminator_loss, wgan_discriminator.
                  trainable_variables)
29            discriminator_optimizer.apply_gradients(zip(
                  discriminator_gradients, wgan_discriminator.
                  trainable_variables))
30        noise = tf.random.normal([batch_size, z_dim])
31        with tf.GradientTape() as generator_tape:
32            fake_samples = wgan_generator(noise)
33            fake_predictions = wgan_discriminator(fake_samples)
34            generator_loss = -tf.reduce_mean(fake_predictions)
35        generator_gradients = generator_tape.gradient(generator_loss,
              wgan_generator.trainable_variables)
36        wgan_generator_optimizer.apply_gradients(zip(
              generator_gradients, wgan_generator.trainable_variables))
```

# Appendix C

# LSGAN Model Architecture

In this appendix, we present the architecture and code for Least Squares Generative Adversarial Network (LSGAN) mplemented in Python using TensorFlow and Keras. This implementation is used in the research for generating synthetic spectrogram data in SHM.

## C.1 LSGAN Generator and Discriminator Architecture

**Listing C.1:** LSGAN Generator and Discriminator Architecture

```
def build_lsgan_generator(z_dim):
    lsgan_generator = Sequential()
    lsgan_generator.add(Dense(15 * 65 * 64, input_dim=z_dim))
    lsgan_generator.add(LeakyReLU(alpha=0.2))
    lsgan_generator.add(Reshape((15, 65, 64)))
    lsgan_generator.add(Conv2DTranspose(64, kernel_size=3, strides=1,
        padding='same'))
    lsgan_generator.add(LeakyReLU(alpha=0.2))
    lsgan_generator.add(BatchNormalization())
    lsgan_generator.add(Conv2DTranspose(1, kernel_size=3, strides=1,
       padding='same', activation='tanh'))
    return lsgan_generator

def build_lsgan_discriminator(input_shape):
    lsgan_discriminator = Sequential()
    lsgan_discriminator.add(Conv2D(64, kernel_size=(3,3), strides
       =(2,2), padding='same', input_shape=input_shape))
    lsgan_discriminator.add(LeakyReLU(alpha=0.2))
    lsgan_discriminator.add(Dropout(0.3))
    lsgan_discriminator.add(Conv2D(128, kernel_size=(3,3), strides
       =(2,2), padding='same'))
```

```
    lsgan_discriminator.add(LeakyReLU(alpha=0.2))
    lsgan_discriminator.add(Dropout(0.3))
    lsgan_discriminator.add(Flatten())
    lsgan_discriminator.add(Dense(1, activation='linear'))
    return lsgan_discriminator
```

## C.2 Training Configuration

**Listing C.2:** LSGAN Training Loop

```
def train_lsgan(lsgan_generator, lsgan_discriminator, X_train, epochs
    , batch_size, z_dim):
    lsgan_generator_optimizer = Adam(learning_rate=0.0002, beta_1
        =0.5, beta_2=0.9)
    discriminator_optimizer = Adam(learning_rate=0.0002, beta_1=0.5,
        beta_2=0.9)
    lsgan_losses = {'generator': [], 'discriminator': []}
    batches_per_epoch = int(X_train.shape[0] / batch_size)
    for epoch in range(epochs):
        for batch_num in range(batches_per_epoch):
            start = batch_num * batch_size
            end = start + batch_size
            real_images = X_train[start:end]
            noise = tf.random.normal([batch_size, z_dim])
            generated_images = lsgan_generator(noise)
            with tf.GradientTape() as tape:
                real_output = lsgan_discriminator(real_images)
                fake_output = lsgan_discriminator(generated_images)
                discriminator_loss = lsgan_discriminator_loss(
                    real_output, fake_output)
            gradients_of_discriminator = tape.gradient(
                discriminator_loss, lsgan_discriminator.
                trainable_variables)
            discriminator_optimizer.apply_gradients(zip(
                gradients_of_discriminator, lsgan_discriminator.
                trainable_variables))
            with tf.GradientTape() as tape:
                generated_images = lsgan_generator(noise)
                fake_output = lsgan_discriminator(generated_images)
                generator_loss = lsgan_generator_loss(fake_output)
            gradients_of_generator = tape.gradient(generator_loss,
                lsgan_generator.trainable_variables)
            lsgan_generator_optimizer.apply_gradients(zip(
                gradients_of_generator, lsgan_generator.
                trainable_variables))
        lsgan_losses['generator'].append(generator_loss.numpy())
```

```
26            lsgan_losses['discriminator'].append(discriminator_loss.numpy
                  ())
27            if (epoch + 1) % 10 == 0:
28                print(f'Epoch {epoch+1}/{epochs}, Generator Loss: {
                      generator_loss.numpy()}, Discriminator Loss: {
                      discriminator_loss.numpy()}')
29        print("LSGAN training completed!")
30        return lsgan_losses
```

# Bibliography


[1] Sasan Farhadi, Mauro Corrado, Oscar Borla, and Giulio Ventura. «Prestressing wire breakage monitoring using sound event detection». In: *Computer-Aided Civil and Infrastructure Engineering* 39.2 (2024), pp. 186–202. DOI: 10.1111/mice.13079 (cit. on p. 1).

[2] Sasan Farhadi, Mauro Corrado, and Giulio Ventura. «Automated acoustic event-based monitoring of prestressing tendons breakage in concrete bridges». In: *Computer-Aided Civil and Infrastructure Engineering* 39.1 (2024), pp. 67–83 (cit. on pp. 1, 2, 12, 13, 19, 28).

[3] J. Ko and Y. Ni. «Technology developments in structural health monitoring of large-scale bridges». In: *Engineering Structures* 27.12 (2005), pp. 1715–1725 (cit. on pp. 2, 13).

[4] H. Sohn, C. R. Farrar, F. M. Hemez, D. D. Shunk, D. W. Stinemates, B. R. Nadler, and J. J. Czarnecki. *A review of structural health monitoring literature: 1996–2001.* Tech. rep. LA-13976-MS. Los Alamos National Laboratory, 2004 (cit. on pp. 2, 7).

[5] G. A. Ferro, L. Restuccia, D. Falliano, A. Devitofranceschi, and A. Gemelli. «Collapse of Existing Bridges: From the Lesson of la Reale Viaduct to the Definition of a Partial Safety Coefficient of Variable Traffic Loads». In: *Journal of Structural Engineering* (2022) (cit. on pp. 4, 14).

[6] M. Gul and F. N. Catbas. «Statistical pattern recognition for structural health monitoring using time series modeling: Theory and experimental verifications». In: *Mechanical Systems and Signal Processing* 23.7 (2009), pp. 2192–2204 (cit. on pp. 6, 15).

[7] X. Zhu, J. Li, J. Zhang, and Z. Su. «A review of deep learning-based methods for acoustic emission signal processing in SHM». In: *Journal of Sensors* 2018 (2018), pp. 1–12 (cit. on p. 7).

[8] O. Abdeljaber, O. Avci, S. Kiranyaz, B. Boashash, H. A. Sodano, and D. J. Inman. «Real-time vibration-based damage detection using one-dimensional convolutional neural networks». In: *Journal of Sound and Vibration* 388 (2017), pp. 154–170 (cit. on pp. 7, 14).

[9] Y.Q. Ni, Y. Xia, W.Y. Liao, and J.M. Ko. «Operational modal analysis of a cable-stayed bridge for damage identification». In: *Journal of Structural Engineering* 135.3 (2009), pp. 341–350 (cit. on p. 7).

[10] K. Worden, C. R. Farrar, G. Manson, and G. Park. «Machine learning for structural health monitoring: Opportunities and challenges». In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 365.1851 (2007), pp. 515–537 (cit. on pp. 7, 13).

[11] Chao-Yu Wang et al. «Large scale environmental sound classification based on efficient feature extraction». In: *IEEE 45th International Conference on Parallel Processing Workshops*. 2014, pp. 421–425 (cit. on p. 7).

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. «Generative Adversarial Nets». In: *Advances in neural information processing systems*. 2014, pp. 2672–2680 (cit. on pp. 8, 20).

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollar, and Ross Girshick. «Masked Autoencoders are scalable vision learners». In: *arXiv preprint arXiv:2111.06377* (2021) (cit. on p. 9).

[14] Hangbo Bao, Li Dong, and Furu Wei. «BEiT: Bert Pre-Training of Image Transformers». In: *arXiv preprint arXiv:2106.08254* (2021) (cit. on p. 9).

[15] Diederik P Kingma and Max Welling. «Auto-encoding variational bayes». In: *arXiv preprint arXiv:1312.6114* (2013) (cit. on p. 9).

[16] Haohang Xu, Xiaopeng Zhang, Hao Li, Wenrui Dai, Lingxi Xie, and Qi Tian. «Corrupted Image Modeling for Self-Supervised Visual Pre-Training». In: *arXiv preprint arXiv:2202.03382* (2022) (cit. on p. 9).

[17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. «Unpaired image-to-image translation using cycle-consistent adversarial networks». In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2242–2251 (cit. on pp. 9, 24).

[18] Raffaele Zinno, Sina Shaffiee Haghshenas, Giuseppe Guido, and Alessandro Vitale. «Artificial Intelligence and Structural Health Monitoring of Bridges: A Review of the State-of-the-Art». In: *IEEE Access* 10 (2022), pp. 88058–88073 (cit. on pp. 11, 15).

[19] Yuqing Gao, Boyuan Kong, and Khalid M Mosalam. «Deep leaf-bootstrapping generative adversarial network for structural image data augmentation». In: *Computer-Aided Civil and Infrastructure Engineering* 34.9 (2019), pp. 755–773 (cit. on pp. 12, 23).

[20] Honggan Yu, Hao Sun, Jianfeng Tao, Chengjin Qin, Dengyu Xiao, Yanrui Jin, and Chengliang Liu. «A multi-stage data augmentation and AD-ResNet-based method for EPB utilization factor prediction». In: *Automation in Construction* 137 (2022), p. 104734. DOI: 10.1016/j.autcon.2022.104734. URL: https://doi.org/10.1016/j.autcon.2022.104734 (cit. on p. 13).

[21] Kaleb Smith and Anthony O. Smith. «Time Series GAN (TSGAN): Conditional GAN for Time Series Generation». In: *Journal Name* Volume Number (2023), Pages (cit. on pp. 13, 15, 17).

[22] K. M. Mosalam and Y. Gao. «Artificial Intelligence in Vision-Based Structural Health Monitoring». In: *Generative Adversarial Network for Structural Image Data Augmentation*. Springer Nature Switzerland AG, 2024, pp. 247–273 (cit. on pp. 14, 15, 26).

[23] F. Bazzucchi and G. A. Ferro. «The anatomy of a collapse: Forensic analyses, monitoring and restoration attempts of the Fossano Bridge». In: *Advances in Engineering Materials, Structures and Systems: Innovations, Mechanics and Applications*. Ed. by A. Zingoni. London, UK: Taylor & Francis Group, London, 2019. ISBN: 978-1-138-38696-9 (cit. on p. 14).

[24] Guillermo Iglesias and Edgar Talavera. «Data Augmentation techniques in time series domain: a survey and taxonomy». In: *Journal Name* Volume Number (2023), Pages (cit. on pp. 16, 20).

[25] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. «Unpaired image-to-image translation using cycle-consistent adversarial networks». In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232 (cit. on p. 18).

[26] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. «Perceptual losses for real-time style transfer and super-resolution». In: *European Conference on Computer Vision*. Springer. 2016, pp. 694–711 (cit. on p. 18).

[27] Ian McLoughlin et al. «Deep neural networks for environmental sound classification». In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.3 (2015), pp. 540–552 (cit. on p. 18).

[28] Yinggang Liu, Hong Fu, Ying Wei, and Hanbing Zhang. «Sound Event Classification Based on Frequency-Energy Feature Representation and Two-Stage Data Dimension Reduction». In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), pp. 1290–1304 (cit. on pp. 19, 28).

[29] David Foster. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*. O'Reilly Media, 2019. ISBN: 978-1-492-04194-8 (cit. on pp. 20, 29).

[30] Kushal Virupakshappa and Erdal Oruklu. «Using Generative Adversarial Networks to Generate Ultrasonic Signals». In: *2020 IEEE International Ultrasonics Symposium (IUS)*. 2020, pp. 1–4 (cit. on pp. 21, 24).

[31] Martin Arjovsky, Soumith Chintala, and Léon Bottou. «Wasserstein gan». In: *arXiv preprint arXiv:1701.07875* (2017) (cit. on p. 22).

[32] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. «Spectral normalization for generative adversarial networks». In: *International Conference on Learning Representations (ICLR)*. 2018 (cit. on p. 22).

[33] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. «Improved Techniques for Training GANs». In: *arXiv preprint arXiv:1606.03498* (2016) (cit. on p. 22).

[34] Dougal J Sutherland, Hieu Tung, Heiko Strathmann, Soham De, Aaditya Ramdas, Alex Smola, and Arthur Gretton. «Generative models and model criticism via optimized maximum mean discrepancy». In: *International Conference on Learning Representations*. 2017 (cit. on p. 22).

[35] Kai Chen, Zhili Liu, Lanqing Hong, Hang Xu, Zhenguo Li, and Dit-Yan Yeung. «Mixed Autoencoder for Self-Supervised Visual Representation Learning». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023 (cit. on p. 22).

[36] Mehdi Mirza and Simon Osindero. «Conditional generative adversarial nets». In: *arXiv preprint arXiv:1411.1784* (2014) (cit. on p. 23).

[37] Yue Yu, Bin Wu, and Changlong Wang. «Conditional GAN for time series forecasting». In: *Proceedings of the International Conference on Neural Information Processing*. 2017, pp. 504–515 (cit. on p. 23).

[38] X. Yu and F. Porikli. «Ultra-resolving face images by discriminative generative networks». In: *European Conference on Computer Vision*. Springer. 2017, pp. 103–118 (cit. on pp. 23, 24).

[39] Alec Radford, Luke Metz, and Soumith Chintala. «Unsupervised representation learning with deep convolutional generative adversarial networks». In: *International conference on learning representations*. 2016 (cit. on p. 23).

[40] Han Zhang, Tao Xu, Hongsheng Li, et al. «Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks». In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 5907–5915 (cit. on p. 24).

[41] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. «Image-to-Image Translation with Conditional Adversarial Networks». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134 (cit. on p. 24).

[42] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. «Unpaired image-to-image translation using cycle-consistent adversarial networks». In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232 (cit. on p. 24).

[43] Jiahao Lin, Zhengping Wang, et al. «TSGAN: Time series generative adversarial networks for generating and forecasting time series data». In: *Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–8 (cit. on p. 24).

[44] Dong Nie, Roger Trullo, Jie Lian, Li Wang, Caroline Petitjean, Su Ruan, and Dinggang Shen. «Medical image synthesis with context-aware generative adversarial networks». In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2017, pp. 417–425 (cit. on p. 24).

[45] Jinsung Yoon, Daniel Jarrett, and Mihaela van der Schaar. «Time-series Generative Adversarial Networks». In: *Advances in Neural Information Processing Systems*. 2019, pp. 5508–5518 (cit. on p. 24).

[46] Cristóbal Esteban, Sarah L Hyland, and Gunnar Rätsch. «Real-valued (medical) time series generation with recurrent conditional gans». In: *arXiv preprint arXiv:1706.02633* (2017) (cit. on p. 24).

[47] Haoran Zhang et al. «Robust sound event classification using deep neural networks». In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.3 (2015), pp. 540–552 (cit. on p. 25).

[48] Anup K Srivastava et al. «Short-term load forecasting methods: A review». In: *2016 International Conference on Emerging Trends in Electrical Electronics Sustainable Energy Systems (ICETEESES)*. IEEE. 2016, pp. 130–138 (cit. on p. 25).

[49] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. «MirrorGAN: Learning Text-to-Image Generation by Redescription». In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2019, pp. 8898–8907 (cit. on p. 25).

[50] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. «GANs trained by a two time-scale update rule converge to a local Nash equilibrium». In: *Advances in neural information processing systems*. 2017 (cit. on p. 32).