Mechanical Systems and Signal Processing 204 (2023) 110725

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

journal homepage: www.elsevier.com/locate/ymssp

Multi-modal generative adversarial networks for synthesizing time-series structural impact responses

Zhymir Thompson [a,b], Austin R.J. Downey [a,c,*], Jason D. Bakos [b], Jie Wei [d], Jacob Dodson [e]

[a] Department of Mechanical Engineering, United States
[b] Department of Computer Science and Engineering, United States
[c] Department of Civil and Environmental Engineering University of South Carolina, Columbia, SC 29208, United States
[d] Department of Computer Science The City College of New York, 160 Convent Ave, New York, NY 10031, United States
[e] Air Force Research Laboratory, United States

ARTICLE INFO

Communicated by S. Laflamme

Dataset link: https://github.com/High-Rate-SHM-Working-Group/Dataset-1-High-Rate-Drop-Tower-Data-set, https://github.com/High-Rate-SHM-Working-Group/Dataset-1a-Shock-Test-GAN-model

MSC:
00-01
99-00

Keywords:
Time-series
Machine learning
Adversarial network
Impact
High-rate dynamics

ABSTRACT

The process of validating newly-defined state observers can potentially require a significant amount of data gathered from instrumentation. However, collecting data for high-rate dynamic events (short time-scale impact and shock) can be very expensive. Additionally, experiments performed for collecting data can provide highly variable results while the high-energy impacts introduce damage into the structure being tested, consequently resulting in different results for subsequent tests. This paper proposes the use of Generative Adversarial Networks (GANs) to generate data that supplements the experimental datasets required for the validation of state observers. GANs are a class of deep learning models used for generating data statistically comparable to that on which it was trained. They are an ideal candidate for validation due to their consistency, speed, and portability during inference. In this work, data collected from an electronics package under shock is used to examine the generative ability of GANs. This paper proposes a conditional Wasserstein GAN (CWGAN) implementation for the production of synthetic high-rate dynamic vibrations, and introduces the conditional input to the critic at the layer towards the end as opposed to the first layers. Results suggest the generative model proposed is capable of producing data statistically similar to the provided training data. The generated data is compared to the training data, and the advantages and limitations of the model are explored. The model and its artifacts are provided as supplemental material to this article and shared through a public repository.

1. Introduction

Hypersonic structures, vehicular accidents, active blast mitigation; are a few examples of applications that undergo high-rate dynamic events. High-rate dynamic events are defined by a time scale of less than 100 ms and an observed amplitude exceeding 100 g_n [1]. High-rate structural health monitoring (HRSHM) is a method to assess and mitigate rapid changes to structures [2]. The development of data-driven state observers that can react at such high speeds while capturing the complex dynamics of the system is difficult but attainable given that a sufficiently large dataset is available for training. The introduction of state observers for tracking the state (i.e. damage) of the structure [3,4] is one approach. The major consequence of a state-based approach is that

* Corresponding author at: Department of Mechanical Engineering, United States.
E-mail address: austindowney@sc.edu (A.R.J. Downey).

https://doi.org/10.1016/j.ymssp.2023.110725
Received 1 December 2022; Received in revised form 9 August 2023; Accepted 24 August 2023
Available online 9 September 2023

dynamic events lead to inconsistent responses from the structure as continuous damage accumulates over the life of the system. Repeated experiments result in significant variation among test cases, so a sufficiently large data set is required to account for the response of the structure due to its chaotic nature. However, collecting such a large data set can be impractical. Suppose a model is needed to make predictions for a hypersonic structure, and that, due to the nature of the problem, the task is nonlinear. It would be reasonable to assume that such a model would require a large dataset to account for all of the variables associated with the structure. That dataset would likely need to be on the scale of tens of thousands of examples at minimum for decent performance. Additionally, testing would require the development of multiple test structures due to the destructive nature of impact testing. In this scenario, data collection is prohibitively expensive and time consuming [5].

Time and labor limit dataset development, but these limitations could be lessened by synthesizing the dynamic response of structures under impact, sometimes termed *high-speed penetration mechanics* [6]. In structural and penetrator design, simplified approaches are commonly use to model ballistic impact dynamics because the lack of fundamental knowledge about impact phenomena precludes the use of more exact models [7]. While signals can be generated using first-order principles through the use of finite element analysis (FEA) models, high-fidelity FEA models that consider impact and penetration dynamics along with material separation are complex to produce [8]. Furthermore, developing variability in the FEA-derived dataset requires the seeding of select faults or test parameters into the models which is challenging to perform correctly. Purely data-driven approaches that build on experimental data offer the potential to expand datasets for structures under impact without the need to develop and manage complex FEA models nor simplify the approach which removes the higher fidelity component.

There are various examples of using GANs to generate temporal training data [9]. The survey from Brophy et al. provides a myriad of examples of GANs used to generate discrete and continuous data as well as detailing common challenges for GANs, common datasets used for GAN training, and some of the different fields GANs have been used [9]. Unlike classification or regression-based machine learning methodologies, standardized metrics are of limited use when comparing synthesized data across different classes of datasets. Therefore in what follows, we provide a context on relevant prior work and discussions on their chosen metrics.

Hartmann et al. created a GAN to generate EEG signals by using a GAN with a modified training loop. The generator with the best signals visually used strided convolutions and cubic interpolation. The metrics observed in the authors' results were inception score (a model-based evaluation of generator quality), Frechet-inception distance (a model-based evaluation of generator quality and output diversity), euclidean distance, and sliced wasserstein distance (a approximate measurement of the wasserstein distance between samples of the real and generated distributions converted to scalar values). Interestingly, the proposed model only performed best in the sliced Wasserstein distance metric with a score of 0.078, and performed worst in inception score, Frechet-inception distance, and minimum euclidean distance with scores of 1.292, 33.765, and −0.375 respectively. The scores mostly contradict intuition about the most visually realistic synthetic EEG signals, but as these metrics are all accepted as good indicators of model quality, diverse metrics are necessary to better illustrate model quality. The authors simultaneously demonstrated the ability to generate synthetic EEG signals and the importance of using multiple metrics to improve model evaluation [10].

Park et al. recreated frog sounds via GAN with a modified architecture. The generator from the GAN was then used for a number of experiments, one of which used a combination of generated and real data to train a classifier. For the experiment, classifiers were trained with augmented datasets composed of varied ratios of real to generated data. After training the classifiers, the classifier trained on the augmented dataset with the highest proportion of generated data had a "best average improvement of about 1.1%" compared to training solely on real data, and the classifier converged in fewer epochs [11].

Cai et al. generated two dimensional frictional signals using a generator that took preprocessed fabric images, along with noise, as input to generate spectrograms that could then be converted into electrovibrations [12]. Kuo et al. combined a GAN with an autoencoder to create denoised acoustic signals [13].

Jaylakshmy and Sudha made a conditional GAN to generate respiratory signals. Their models, using their proposed method with scalograms as input, achieved an accuracy of 92.68% and 92.5% for the Rale & Think Labs Lung Sounds Library and ICBHI datasets. The model with the next highest accuracy had an accuracy of 86.89% [14]. The numerous applications of time-series GANs in their review further supports the viability of data generation for sequential data like those found in high-rate dynamic events.

This paper proposes a conditional generative model capable of synthesizing data reminiscent of the data used to train it. Further, the model can mix features in the training data to simulate combinations of input unseen in the training data. Rare dynamics can cause an imbalance in the dataset used to train data-driven approaches. The model proposed in this work can be used to bolster the library used to train data-driven approaches leading to better balance through an increased diversity of events. Experiments can then be focused on collecting a wide assortment of examples rather than an exhaustive number of the most rare, which are often the most tedious to obtain. Further still, comparatively few multivariate experiments are required with the added benefit of multi-modal selection. Multi-modal selection allows for the synthesizing of dynamic responses that potentially represent unmonitored locations within the structure. For this work, the Conditional Wasserstein Generative Adversarial Network (CWGAN), a deep learning algorithm for selectively producing synthetic data, was chosen as the generic framework of the generative model [15–17]. When provided a library of prerecorded events, the CWGAN generates data that is statistically indiscernible from the original library. The contributions of this paper include: (1) a CWGAN trained on high-rate dynamic vibration data, and (2) development of a multi-modal conditional WGAN for selective multi-class interpolation. The code developed by this project and its pre-trained models are available as an open-source library within this article's supplemental documents and via a public repository [18].

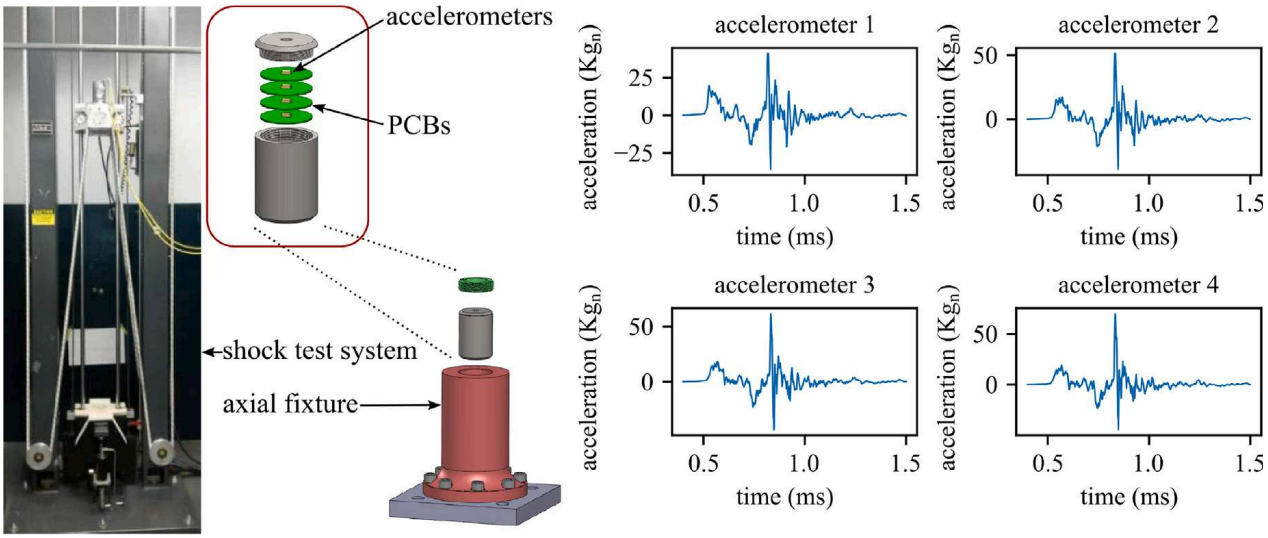

Fig. 1. An electronics package developed by Dodson et al. [19] where accelerometers were mounted, each unpopulated PCB board housed in axial fixture, and held in place with a single lock ring torqued at 25 ft-lbs; the right-hand-side of the figure shows examples of the measured accelerations for the four PCBs.

2. Background

This work uses measured structural response data of electronic packages under shock as the considered high-rate dynamic event [19]. This data set has been well studied for developing high-rate state observers [20]. Fig. 1 shows the shock test system used in the experiments and examples of the data collected. The system consists of a canister that houses four printed circuit boards (PCB) each with a high-g accelerometer (Meggitt 72) mounted in their center. The PCBs are potted in the canister to secure components in place. For this test, the shock test system's table was dropped from a height of 800 mm and a 1.58 mm F1 felt pad was used as the programmer to cushion the impact. This data is available through a public repository [19].

Wasserstein Generative Adversarial Networks [16] are a variation of a type of deep learning technique proposed by Goodfellow et al. [15]. Generative Adversarial Networks are composed of a discriminator and generator. When optimally trained, the generator is capable of producing data indiscernible from the training distribution. To accomplish this, the discriminator and generator both attempt to optimize the function shown in Eq. (1).

$$\mathbb{E}_{x\sim\mathbb{P}_r}[\log f(x)] + \mathbb{E}_{x\sim\mathbb{P}_\theta}[\log(1 - f(x))] \tag{1}$$

The goal of the discriminator is to maximize the value of the function while the generator works to minimize the value. Theoretically, this competition leads the generator to produce more realistic data over repeated iterations until the real and generated distributions are indistinguishable. In practice, this end goal can be more difficult to achieve. WGANs remedy two issues of interest: balancing the learning of the discriminator and generator, preventing mode collapse. Like GANs, WGANs are capable of producing realistic data similar to, but not the same as, the given set of input data. They accomplish this by approximating the Earth Mover Distance. Throughout training, the critic, analogous to the discriminator for GANs [16], tries to maximize the function and the generator attempts to minimize it until they reach a balance. WGAN transforms the objective function for the critic and generator as shown in Eq. (2).

$$\max_{\|f\|_L\leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)] \tag{2}$$

where $\mathbb{E}$ represents a summation over samples from the real $\mathbb{P}_r$ and the generated $\mathbb{P}_\theta$ distributions, and $f(x)$ is the function representing the critic. The goals of the critic and generator are still to maximize and minimize the value of the function respectively. The change to the objective function converts the gradient slope for backpropagation from sigmoidal to a linear shape. This change helps prevent convergence failure by vanishing gradients and simultaneously reduces the likelihood of the generative model undergoing mode collapse. The conditional WGAN (CWGAN) is a variant of the WGAN that uses auxiliary data as in the conditional GAN (CGAN) represented with Eq. (3).

$$\max_{\|f\|_L\leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x|y)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x|y)] \tag{3}$$

where $f(x|y)$ is the function representing a critic trained on x and conditional input y, and y is any additional input used in both the critic and generator. A common example for this supplementary information would be the use of class labels; these class labels could then be uni-modal or multi-modal. For the experiment, the model received prior noise as well as a fixed size array representing the presence or absence of a class label (i.e. [1, 0, 0] would represent class 1 with no other classes present) where each class represented the corresponding accelerometer of the four shown in Fig. 1. The CWGAN should still converge to its optimum if the new function meets the constraints of the WGAN. Since the weight constraints still apply to the additional input and respective weights, it is assumed that the constraints remain satisfied and the CWGAN will still converge.

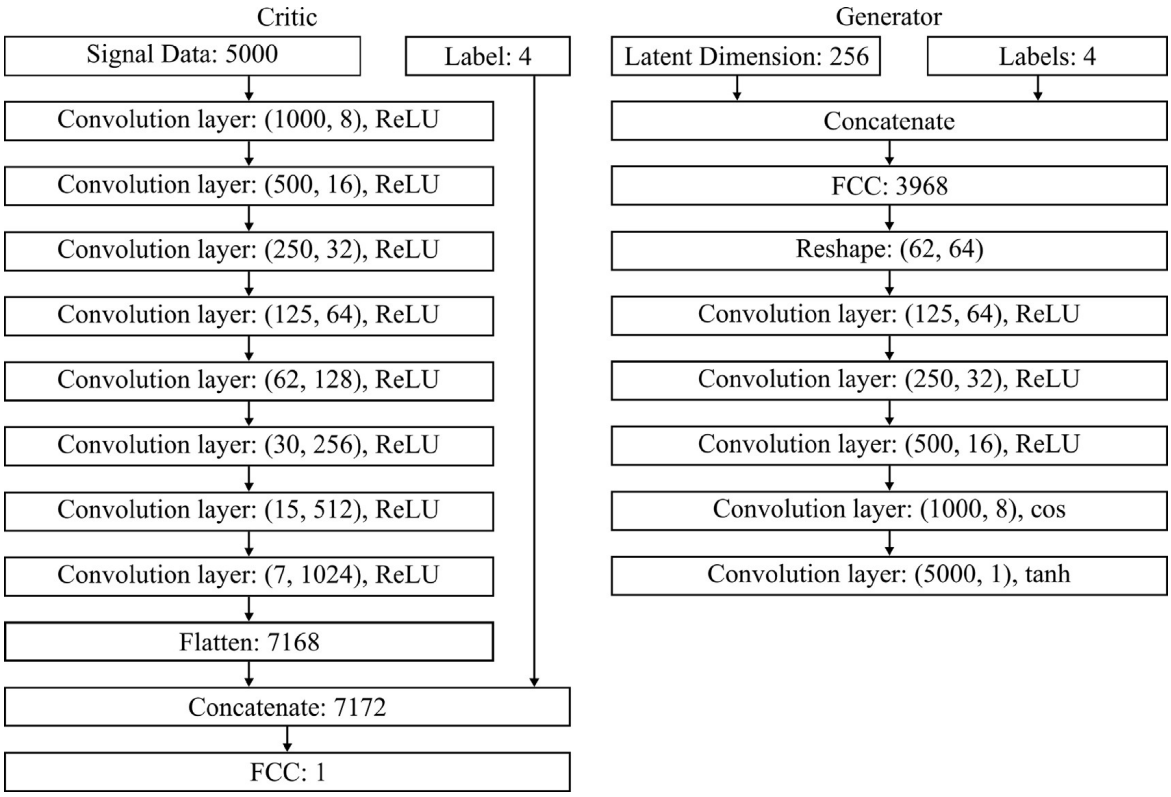

Fig. 2. Diagram of the architectures used for critic and generator models. The values in the convolution layers represent output vector size and number of filters respectively.

Table 1
Scoring for different models compared to input dataset. The input dataset is shown in the first row with columns representing different time-domain statistics applied. Following rows correspond with generative models examined.

	Abs mean	Root mean square	Skewness	Kurtosis	Shape factor	Impulse factor	Crest factor
real	2.53e−02	2.46e+01	2.69e+00	5.57e+01	1.05e+03	4.13e+01	3.98e−02
CWGAN	2.94e−02	2.45e+01	2.71e+00	4.36e+01	8.46e+02	2.88e+01	3.41e−02
CEBGAN	1.65e−02	1.58e+01	4.65e+00	7.86e+01	9.97e+02	5.56e+01	5.46e−02
CVAE	9.95e−01	3.87e+01	1.35e+01	2.13e+02	3.89e+01	9.98e−01	2.56e−02

3. Materials and methods

The CWGAN model architecture follows the DCGAN model architecture. The critic is a CNN with nine hidden layers, as diagrammed on the left-hand-side of Fig. 2. The signal data is passed through eight convolution layers to collect informative features. The label is not passed through the convolution layers. Instead, the label is treated as a collection of features, and the label is concatenated with the flattened output of the convolution layers before passing to the final layer. The final output combines the collection of features into a final singular output. The activation functions for the layers are leaky ReLU. The generator is a CNN with five hidden layers. The last two layers of the generator use the cosine and tanh activation functions respectively [11]. The CWGAN model was chosen over a conditional Variational Autoencoder (VAE) or Energy-based GAN (EBGAN) for its apparent advantage in this experiment. Prior to creating the conditional model, the WGAN, VAE, and EBGAN were all trained to produce the desired signals. Visually, the WGAN appeared to have more diverse outputs with a similar level of closeness to the expected outputs. Regarding metrics, Table 1 provides a comparison of time-based metrics between the training dataset and generated data for each model. Overall, the CWGAN had the closest metrics to the training data with shape factor as the sole exception where CEBGAN was closer. Table 2 provides a comparison over the same data but with frequency-based metrics. This table appears to contradict Table 1 with CEBGAN performing the best in every metric except MMFD. Taken together, these tables imply that CWGAN has higher frequency noise shifting the frequency metrics. It is worth noting, but not likely a major issue since the noise does not shift the signals far from the target distribution. There is no other obvious advantage this model may have to other similar generative models. However, the CWGAN is simpler to implement than an equally sufficient physics-based model, and once trained, the model can produce signals with equal or greater speed to other alternatives.

Fig. 3 diagrams the steps taken for training the model. The training procedure for the model was split into training the critic and training the generator. First, the critic was trained on a batch of training data and associated labels combined with an equally

Table 2
Scoring for different models compared to input dataset. The input dataset is shown in the first row with columns representing different frequency-domain and time/frequency-domain statistics applied. Following rows correspond with generative models examined.

	Root mean square frequency	Root variance frequency	MMFD	TRAC
real	3.68e+04	2.93e+04	4.95e+01	9.91e−01
CWGAN	8.08e+04	7.30e+04	3.17e+00	9.90e−01
CEBGAN	3.68e+04	3.11e+04	1.30e+00	9.93e−01
CVAE	4.74e+04	3.85e+04	7.02e+01	9.92e−01

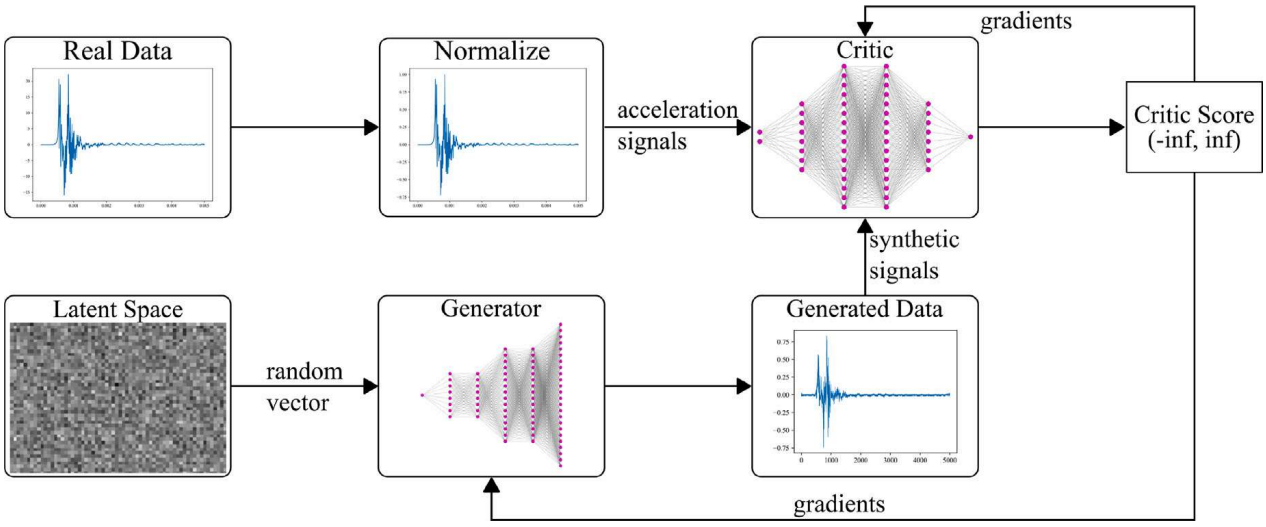

Fig. 3. Flowchart of training steps utilized for the proposed GAN methodology.

sized batch of generated data made with the same labels. The critic was also regularized using w2 regularization where the critic is regularized based on the training data. Regularization was included to improve critic stability during training. This critic training iterated for 4 epochs before beginning the generator training. The generator was trained on the critic's scores for the generated data. The generator then updated its weights and the loop began again for some number of epochs. The training loop iterated until the model was thought to have converged. For the training, convergence was considered a lack of improvement after five epochs.

Algorithm 1 Training procedure for the CWGAN architecture.

procedure TRAIN CWGAN(critic, generator, X, Y)
  X: Real data
  Y: Labels
  n: Number of epochs
  m: Number of critic training loops per epoch
  for $i = 0 \rightarrow n$ **do**
    $(x, y) \leftarrow$ sample(X, Y)
    $r \leftarrow$ Gaussian sample of latent space
    for $j = 0 \rightarrow m$ **do**
      $v \leftarrow$ generator$((r,\ y))$
      $L_c \leftarrow$ Loss(critic$([(x,\ y),\ (v,\ y)])$)
    end for
    $\theta \leftarrow$ ADAM$(L_c, \mathrm{W}_{\mathrm{critic}})$
    $L_g \leftarrow$ Loss(critic([generator$(r,\ y)$]))
    $\theta \leftarrow$ ADAM$(L_g, \mathrm{W}_{\mathrm{generator}})$
  end for
end procedure

Algorithm 1 provides pseudo-code for the training procedure of the CWGAN. This is a more detailed description of the steps summarized in Fig. 3. X is the training data, Y is the corresponding conditional labels, n is the number of training epochs for the

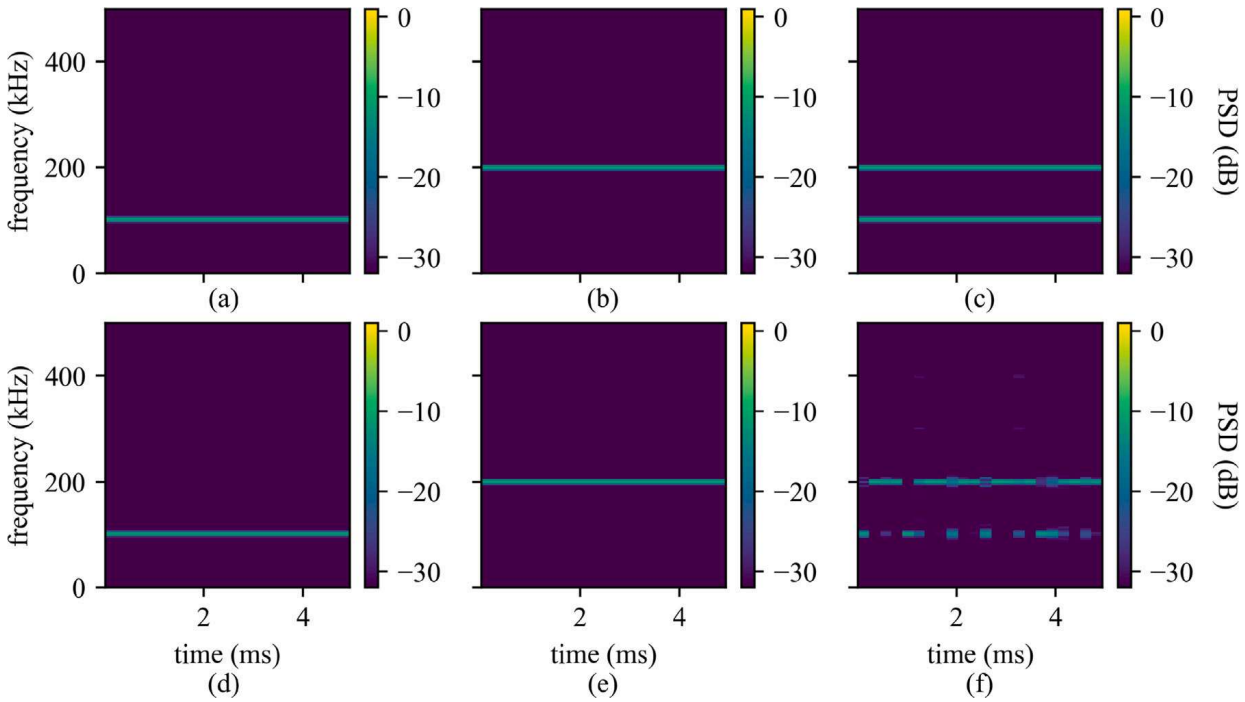

Fig. 4. Demonstration of model behavior when producing multimodal signal. Top row depicts, from left to right, a 100 kHz signal (a), a 200 kHz signal (b), and the sum of the 100 kHz and 200 kHz signals (c). The bottom row depicts a generated signal for the unimodal 100 kHz (d), the unimodal 200 kHz (e), and the multimodal 100 kHz and 200 kHz signal (f).

CWGAN, and m is the number of critic training loops per epoch. One important note is that the loss functions for the critic and generator differ slightly. While both are a Wasserstein loss function, the critic loss is further modified by w2 regularization. This regularization stabilizes the critic optimization process and improves overall training.

To demonstrate the model's behavior on a tractable dataset, the model was trained on sine waves. Sine waves were chosen as a good example signal since they are easier to analyze and verify. For the demonstration, 64,000 single-frequency sine waves were organized into four classes: 100 KHz, 200 KHz, 300 KHz, and 400 KHz. All of the signals were fed to the model for training and the model was used to generate signals according to some chosen label. The trained generator was set to output 16 each of 100 KHz, 200 KHz, and the multiclass combination of the 100 KHz and 200 KHz classes. Fig. 4 reports spectrograms for the reference signals and generated signals for comparison where the top row shows the spectrogram of the source signal and the bottom row the generated data. For the 100 kHz and 200 kHz signals, the model learned to generate accurate representations of the signal. The mixed model had varying results, but generally, the model produced a signal where the frequency components were a weighted average of the separate classes. The model produced better combination signals for some multimodal signals than others, but the unimodal signals were correct regardless. The model was also trained on 160,000 sine waves, but the multimodal signals produced were worse overall based on the spectrograms. This is likely a consequence of the lack of multimodal training examples. This could be mitigated by adding in multimodal examples, but choosing an optimal amount of data, in this case about 64,000, allows for the production of ideal multimodal signals.

The training data used in the experiment was multiplied 4000 times and combined with Gaussian noise to bolster the data for training resulting in 64,000 signals split across four classes. The data was duplicated 4000 times since it struck a decent balance between accuracy and training time. Greater amounts of data would improve both generated signal resolution and diversity, but increasing the amount of data to train on also increases the overall training time. Fig. 5 shows how the generated signals compare to real signals on select metrics. Specifically, Fig. 5 shows the following metrics: (left to right, top to bottom) shape factor, impulse factor, crest factor, frequency center, root mean square frequency, and root variance frequency. In theory, the increased training time could be reduced with batching, but presently, increased batch size decreases overall performance of the given model. The training data for the model was scaled via min–max normalization along each sample. The normalized data was used for training and evaluation. The scaling factors were saved to scale the output data back up for production. Overall, this decreases the convergence time for the model without damaging performance.

An interpolation over batch size was also performed since batch size was noted to have a notable effect on model performance. Fig. 6 shows the collected results. For this interpolation, sine waves were again used as the experiment training data, and the data size was fixed at 64,000 for varying batch sizes from 1–512.

The hyperparameters for the model were chosen using Keras-tuner with the hyperband optimization algorithm [21]. The vector size was insignificant when compared solely by latent dimension vector size. Fig. 7. shows the average change in score as latent dimension size increased. For the experiment, conditional models with different sized latent input dimensions were trained on the

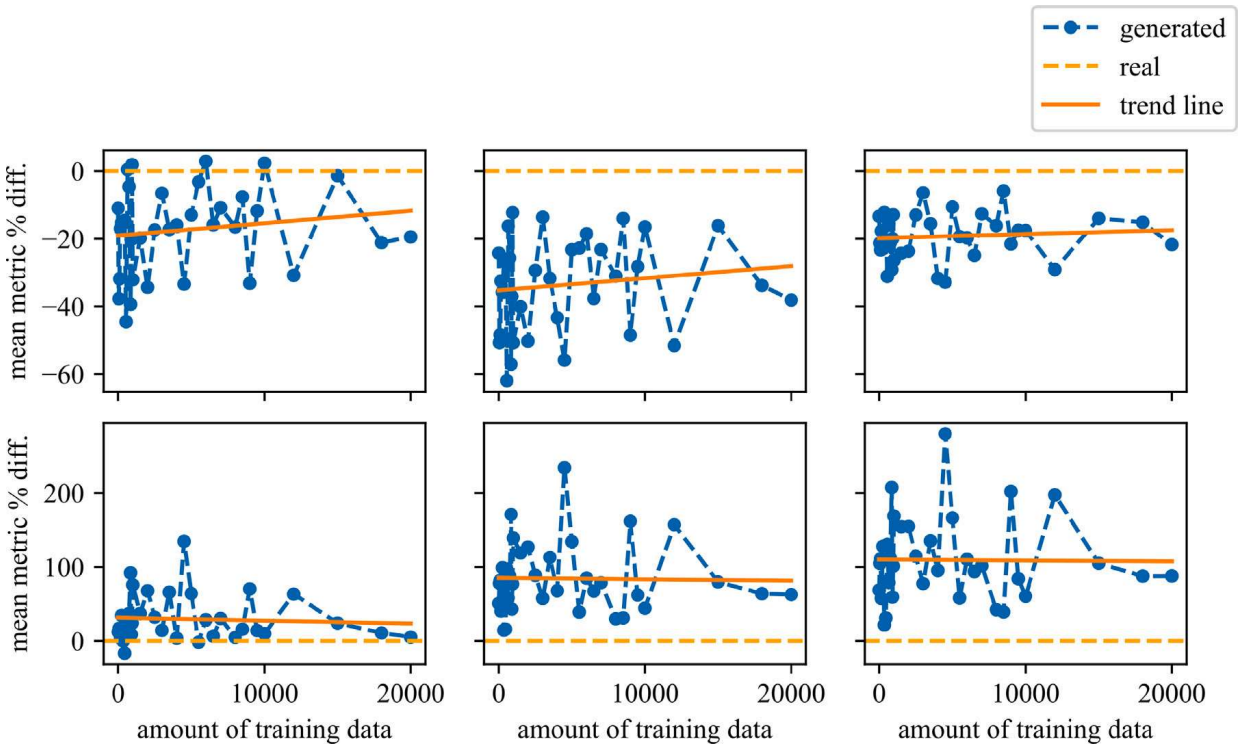

Fig. 5. Metric scores plotted over increasing data amounts. Solid line shows average metric score for model as percent difference from reference value. Dotted line shows metric value for training data (centered at 0). The metrics (left to right, top to bottom) were shape factor, impulse factor, crest factor, frequency center, root mean square frequency, and root variance frequency.

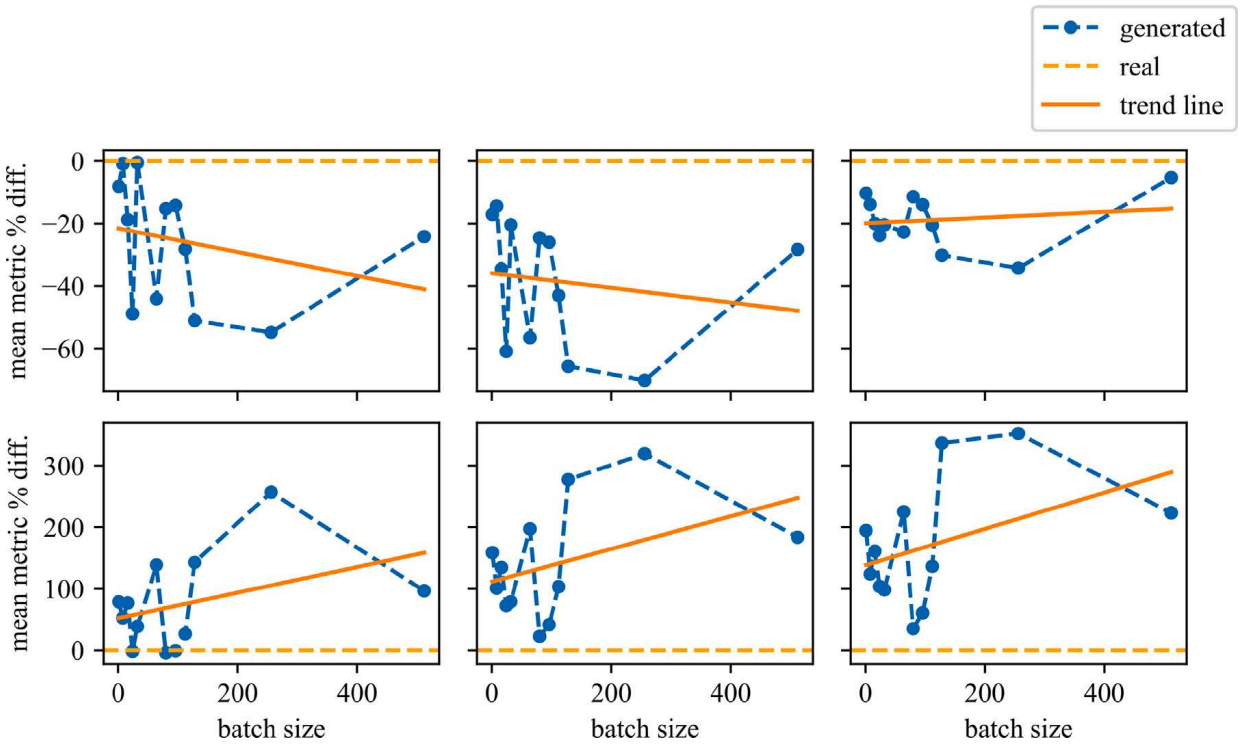

Fig. 6. Metric scores plotted over increasing batch sizes. Solid line shows average metric score for model as percent difference from reference value. Dotted line shows metric value for training data (centered at 0). The metrics (left to right, top to bottom) were shape factor, impulse factor, crest factor, frequency center, root mean square frequency, and root variance frequency.

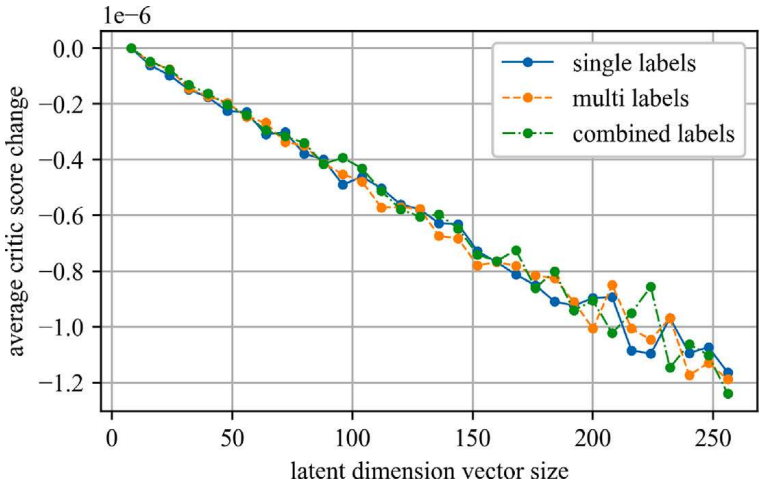

Fig. 7. The overall change in critic scores over latent dimension vector size.

training data to be used for the final model. As shown in Fig. 7, model score improved, but the scale of the increase was minuscule. The learning rates for the critic and generator were 1e-3 and 1e-4. The best-performing critic model consisted of a convolution model with a single starting convolutional layer of kernel size 5 followed by 8 convolutional layers with kernel size 3. The critic was not tested without the first convolutional layer, so it is unknown if the model would perform better without it. The layer in question was included with the expectation that it would reduce training time by downsampling, but the larger filter size and lack of overlap were believed to be able to capture better representations of the larger patterns. The training data is composed of the acceleration data collected in the structure response data set [19].

4. Results/discussion

Qualitative measures are a popular form of GAN evaluation. Two of the most common evaluations are Inception Score [22] and FID [23]. These common evaluation methods are focused around ImageNet and thus not easily applicable to the current model. In this work, qualitative measures were used, but increased focus was placed on quantitative measures [24]. These measures are more objective in nature, and allow for both greater sample sets to inspect as well as more detailed inspection of individual generated signals. Qualitative measures, such as visual inspection of acceleration over time and recurrence plots, appeared visually similar. The acceleration over time plots in Fig. 8 suggest that the generated examples model the real data well. The generated plots are noticeably diverse, and when compared to the training examples the similarity is clear. The generated examples also contain higher noise and lower overall amplitude than their experimental counterparts.

The recurrence plots in Fig. 9 further support the idea that the generated unimodal examples capture the same peaks and troughs as the real signals (as shown by the vertical and horizontal lines at different times in the plots), but those peaks and troughs are less pronounced (shown by the dimmer color representing the short distance between points).

Fig. 10 reports synthesized multi-modal signals that were generated from a combination of two measured signals. As discussed above, these generated multi-modal signals could be thought of as being analogous to the signal obtained at an unmonitored location. The examples in Fig. 10 appear to take some features from both of its components while also being visually distinct from either. These measures suggest the generative model effectively produced the general shape for its outputs. Further, there is sufficient visual variety to suggest modal collapse did not occur within the model.

While visual analysis in the time domain is informative, a more complete picture requires observation in the frequency domain as well. Fig. 11 demonstrates the uni-modal and multi-modal performance of the generated signals in the frequency domain. Following the structure of Fig. 4, Fig. 11 compares the expected signals to their respective generated signals. The figure shows that the uni-modal examples match the reference samples well up to about 200 kHz. Beyond that range, the generated signals appear to add noise not measured in the observed data. The generated multi-modal signal matches the structure of the expected multi-modal signal, but not as well as the uni-modal examples. The noise terms are also still present in the multi-modal example. The noise is likely a consequence of the training procedure, where the critic does not perceive the noise in the higher frequency range as important to signal analysis.

The time-based and frequency-based metrics provide quantitative measures of the distributions for the training data and generated data. Table 3 lists the 11 temporal and frequency metrics used as quantitative metrics in this work. For these metrics: P is power spectrum of a signal, F is frequency components of a signal, X and Y are batches of real and generated samples, x and y are individual signals from their respective batches, n is length of signal in x, M is a number of generated signals in batch y. Generally, these metrics are accepted statistical metrics, except for mean min frequency difference which was developed by the authors in a prior work as a metric to observe the similarity between samples of the generator and training data with respect to

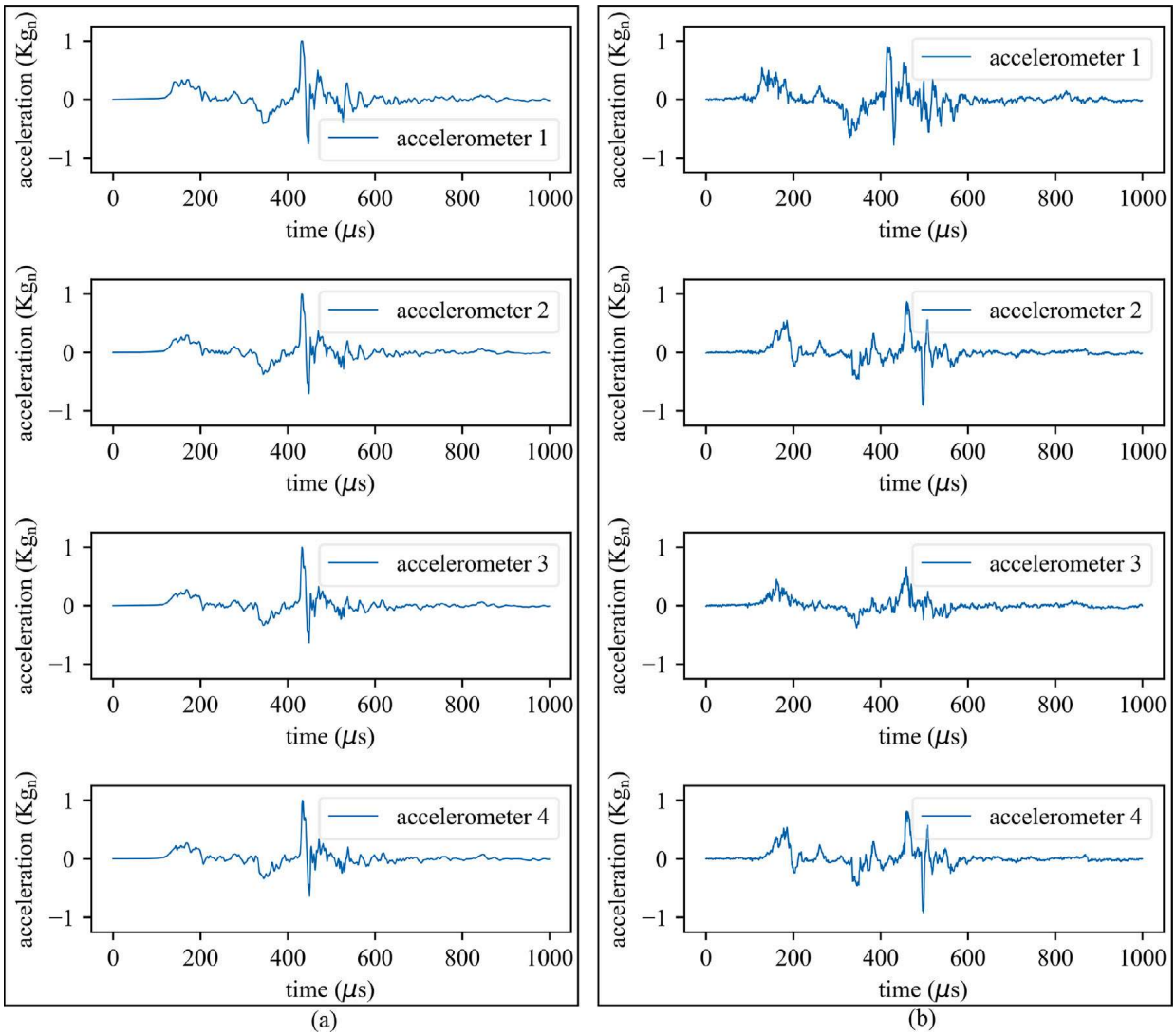

Fig. 8. Unimodal generated signals. Each of the four accelerometers has an arbitrary example from the dataset in column (a) and a generated example in column (b).

their minimum difference in the frequency domain [25]. The data in Tables 4 and 5 show similar values for time-based metrics, low values for TRAC, and significant variation in the frequency-based metrics. However, the metrics are spread across different scales, so the radar charts in Fig. 12 provide an alternative view of the data. In (b) of Fig. 12, the frequency center is shown to be fairly similar, and the RMSF and RVF have greater differences. The frequency center and RMSF differences confirm that the energy of the signals is skewed more towards the higher frequencies of the spectrum. The high RVF suggests the is spread across more of the frequency spectrum compared to the reference signals. The fuzzy property observed in Figs. 8 and 10 is likely a result of the higher frequency energy. The mean min frequency difference for the generated signals cannot be easily compared to the reference data since at baseline the value would be zero. Instead, the MMFD metric for the generated signals was compared to randomly generated signals. This allows a visual comparison where the optimal case would show a separation of the two values. As expected, the generated signals are significantly closer to the reference signals than randomly generated signals.

To further investigate and visualize the statistical separation of the real and synthetic data, Gaussian mixture models of the first 10 features listed in Table 3 were developed for the datasets. The Gaussian mixture model is plotted from the first ten metrics in Figs. 13 and 14 using Principle Component Analysis (PCA) to reduce the dimensionality of the data for plotting. Only the first 10 features were used as MMFD and TRAC are relative metrics, best used to compare between real and synthetic data.

Fig. 13 shows the composition of each of the accelerometer labels. Specifically, the Gaussian mixture model provides a visual aid to dissect how the distinct accelerometer data groups differ from those represented by the generator data. The Gaussian mixture model has initial difficulties separating the expected groups as shown by the different color groupings seen between the actual accelerometer categories in Fig. 13(a) and the predicted groupings of Fig. 13(b). The separation between the reference and generated data suggest the two are easily separable. Fig. 14 shows the separability between reference and generated. Along with the datasets being distinct, the generated data has greater variability in the displayed metrics.

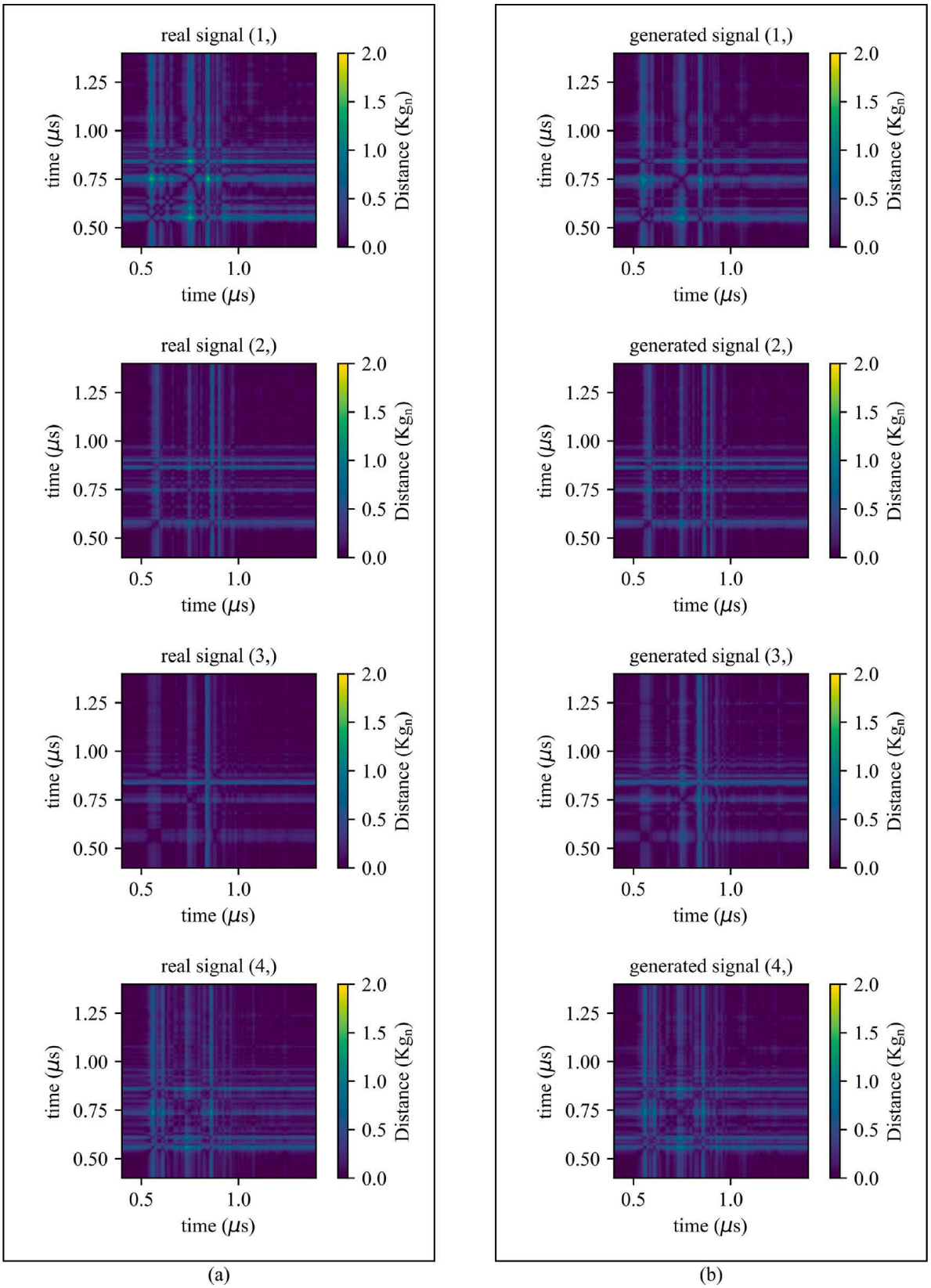

Fig. 9. Recurrence plots for generated signals. Column (a) contains closest matching examples from the classes used for generation. Column (b) contains the corresponding generated examples.

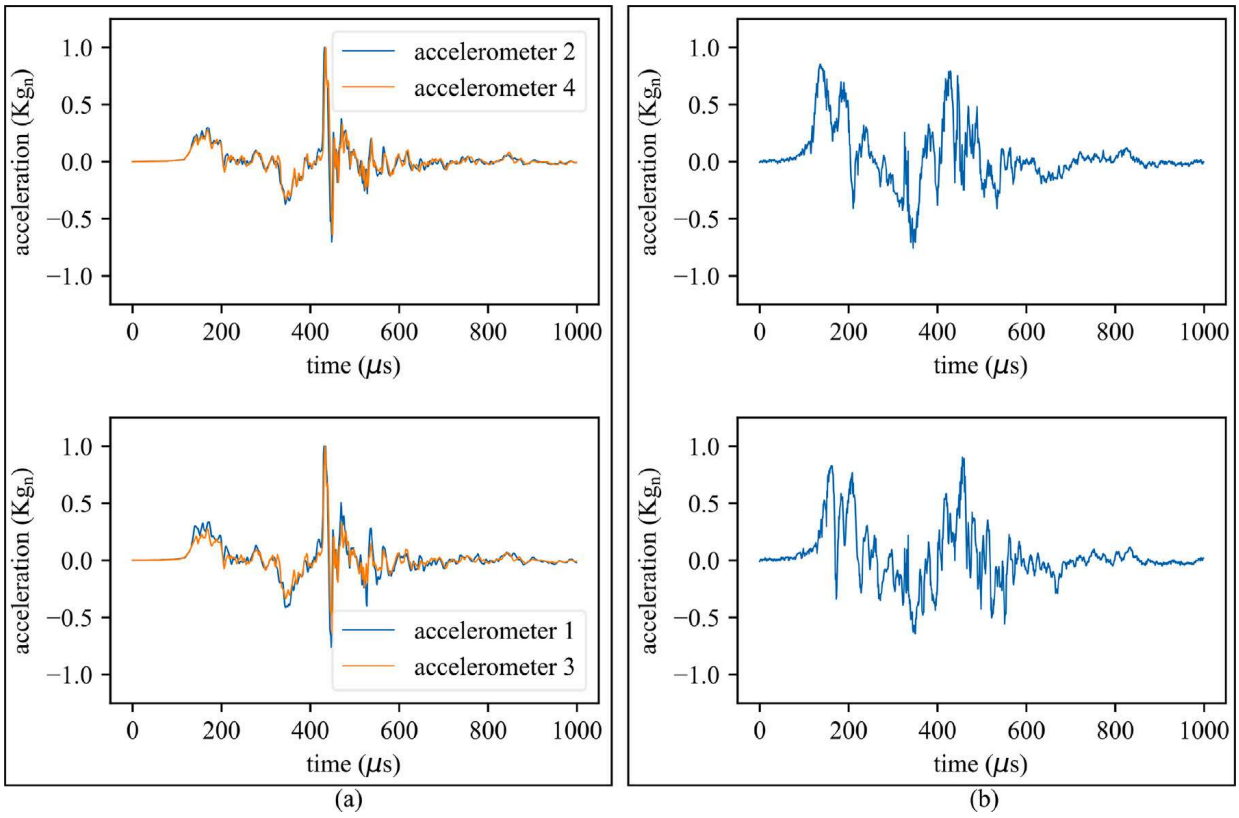

Fig. 10. Multi-modal generated signals. Column (a) contains examples from the classes used for generation overlaid. Column (b) contains the corresponding generated examples.

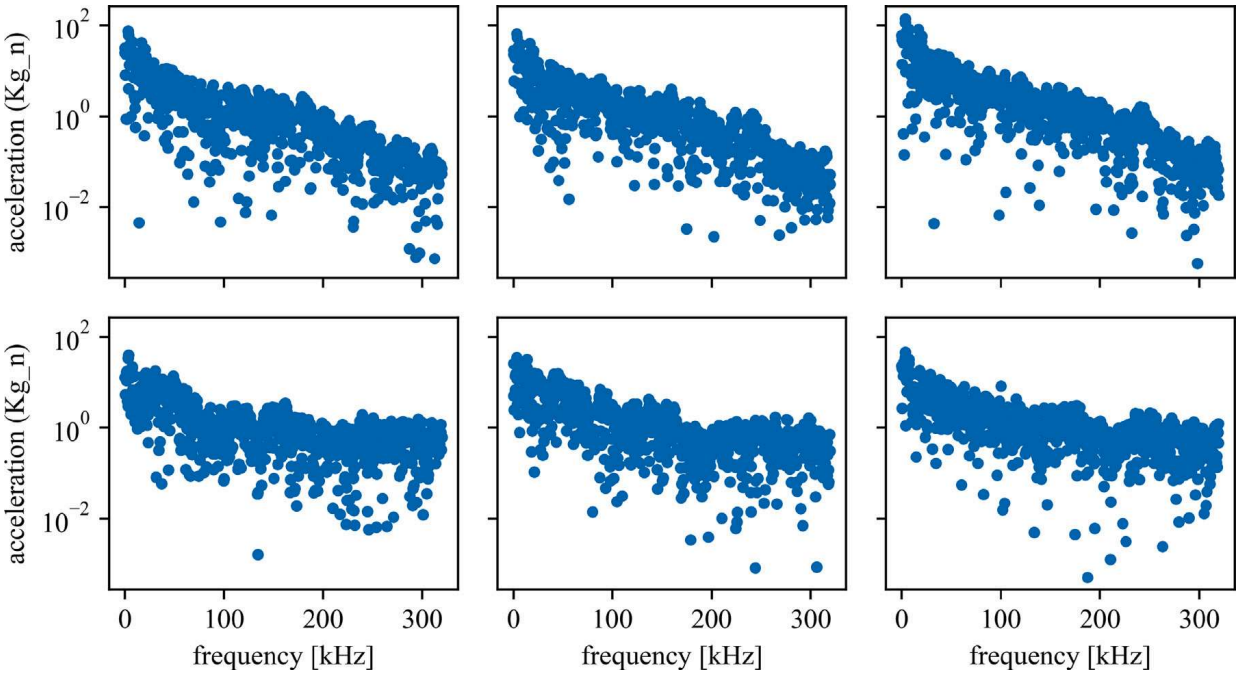

Fig. 11. The frequency components of selected reference and generated signals. The structure is identical to Fig. 4. The top row (from left to right: accelerometer 1, accelerometer 3, accelerometer 1 + accelerometer 3) are the reference signals, and the bottom row represents the respective generated signals.

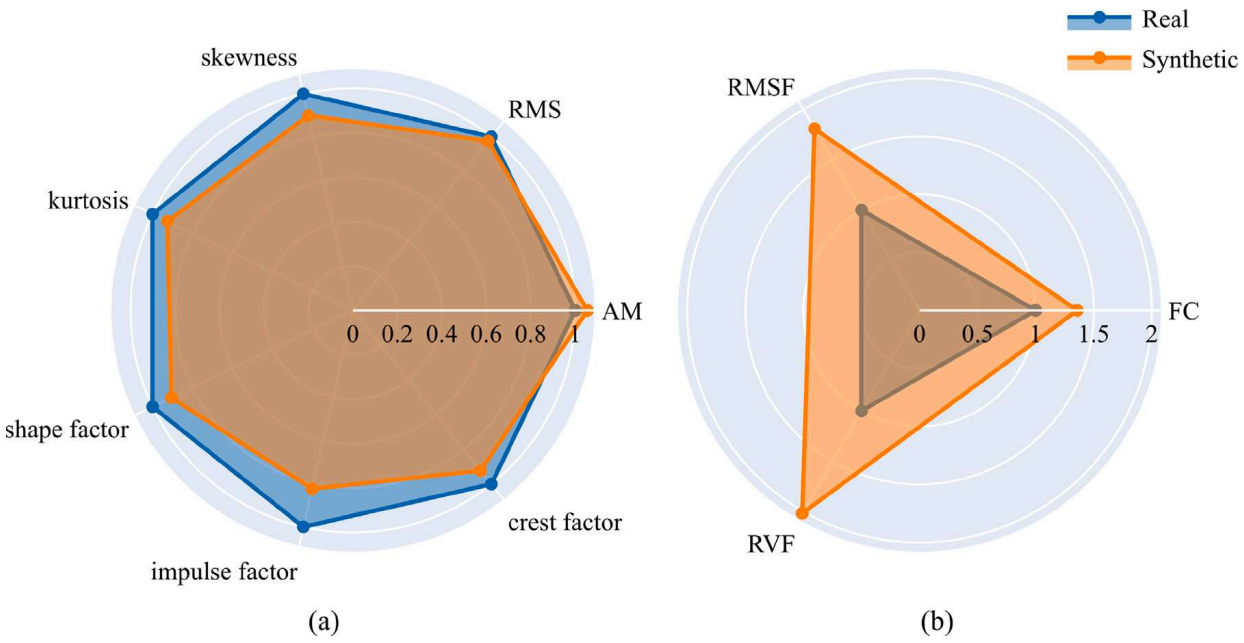

Fig. 12. Radar charts for the metrics listed in the first columns of Tables 4 and 5 for the: (a) time-based, and; (b) frequency-based domains.

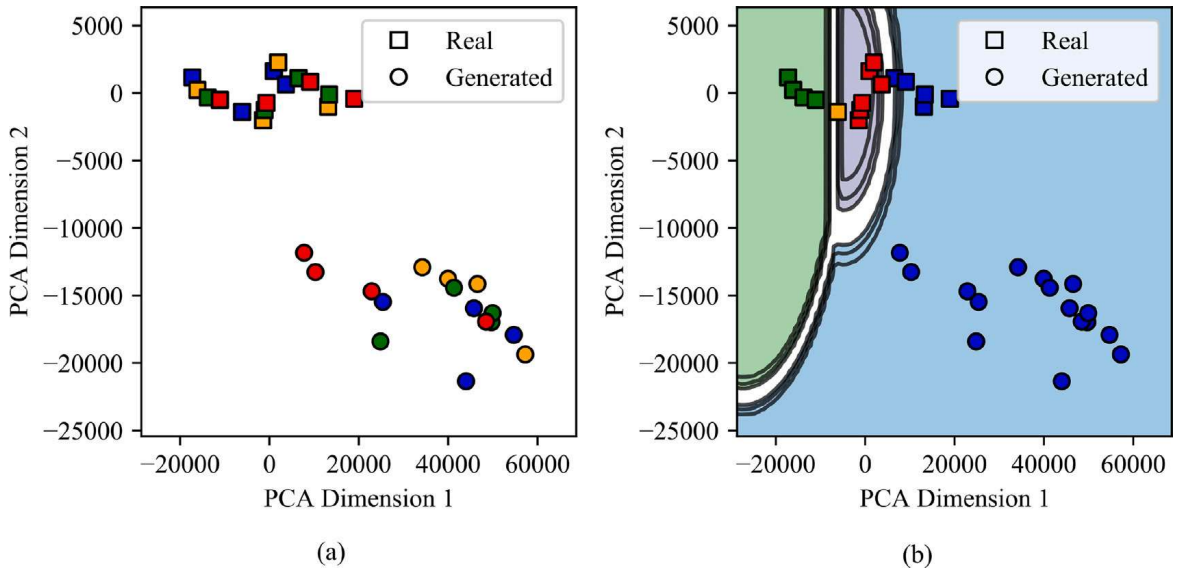

Fig. 13. Plot of Gaussian mixture model predictions over reference and generated data. x and y axes are PCA dimension reduced from the first ten metrics in Table 3 where the square marker represents real data and the circle marker represents generated data. Color represents grouping where: (a) displays colors for which accelerometer classification the data originated, and (b) displays colors for the groups chosen by the Gaussian mixture model.

Table 3
Metrics used for scoring the similarity between samples of the generator and training data, in both the temporal and frequency domain.

Metric	Formulation
Absolute mean	$\frac{1}{n}\sum\lvert x\rvert$
RMS	$\sqrt{\frac{1}{n}\sum x^2}$
skewness	$\frac{\sum x-\bar{x}^3}{(n-1)s(x)^3}$
kurtosis	$\frac{\sum x-\bar{x}^4}{(n-1)s(x)^4}$
shape factor	$\frac{\mathrm{RMS}(x)}{\text{absolute mean}(x)}$
impulse factor	$\frac{\max(x)}{\text{absolute mean}(x)}$
crest factor	$\frac{\max(x)}{\mathrm{RMS}(x)}$
frequency center	$\frac{\sum P\cdot F}{\sum P}$
Root mean square frequency	$\sqrt{\frac{\sum F^2\cdot P}{\sum P}}$
Root variance frequency	$\sqrt{\frac{\sum(F-\text{frequency center}(x))^2}{\sum P}}$
mean min frequency difference	$\frac{1}{M}\sum\min\left(\mathrm{MSE}[\mathrm{FFT}(X)-\mathrm{FFT}(Y)]\right)$
TRAC	$\frac{[x^Ty]^2}{[x^Tx][y^Ty]}$

Table 4
Scoring for input dataset; the total is FFT score over all accelerometer data and each subsequent column is an FFT score for the respective accelerometer data.

Metric	Total	[1 0 0 0]	[0 1 0 0]	[0 0 1 0]	[0 0 0 1]
Abs mean	2.53e−02	3.03e−02	2.56e−02	2.31e−02	2.22e−02
RMS	8.56e−02	9.92e−02	8.67e−02	7.96e−02	7.67e−02
skewness	2.69e+00	2.37e+00	2.63e+00	2.88e+00	2.88e+00
kurtosis	5.57e+01	4.28e+01	5.26e+01	6.08e+01	6.66e+01
shape factor	3.45e+00	3.32e+00	3.44e+00	3.51e+00	3.55e+00
impulse factor	4.13e+01	3.42e+01	4.07e+01	4.41e+01	4.63e+01
crest factor	1.18e+01	1.02e+01	1.18e+01	1.24e+01	1.29e+01
frequency center	2.22e+04	2.00e+04	2.17e+04	2.27e+04	2.42e+04
Root mean square frequency	3.68e+04	3.35e+04	3.64e+04	3.77e+04	3.97e+04
Root variance frequency	2.93e+04	2.68e+04	2.91e+04	3.00e+04	3.14e+04
MMFD	4.90e+01	4.90e+01	4.99e+01	4.96e+01	5.00e+01
TRAC	9.91e−01	9.86e−01	9.90e−01	9.89e−01	9.86e−01

Table 5
Scoring for generative model; Each row is a statistical measure for the data, the total column is the statistical score over all accelerometer data and each subsequent column is a score for the respective accelerometer data.

Metric	Total	[1 0 0 0]	[0 1 0 0]	[0 0 1 0]	[0 0 0 1]
Abs mean	2.62e−02	2.83e−02	2.36e−02	2.60e−02	2.58e−02
RMS	7.97e−02	8.54e−02	7.02e−02	7.74e−02	7.77e−02
skewness	2.88e+00	2.42e+00	2.86e+00	2.99e+00	2.92e+00
kurtosis	5.09e+01	4.15e+01	5.92e+01	5.41e+01	5.28e+01
shape factor	3.03e+00	3.02e+00	2.98e+00	2.97e+00	3.00e+00
impulse factor	3.47e+01	3.09e+01	3.61e+01	3.52e+01	3.47e+01
crest factor	1.15e+01	1.02e+01	1.21e+01	1.19e+01	1.16e+01
frequency center	2.99e+04	2.94e+04	3.64e+04	3.05e+04	2.71e+04
Root mean square frequency	6.66e+04	6.83e+04	7.61e+04	6.94e+04	6.07e+04
Root variance frequency	5.94e+04	6.15e+04	6.68e+04	6.22e+04	5.42e+04
MMFD	1.40e+00	3.71e+00	2.26e+00	1.95e+00	2.08e+00
TRAC	9.91e−01	9.88e−01	9.89e−01	9.87e−01	9.90e−01

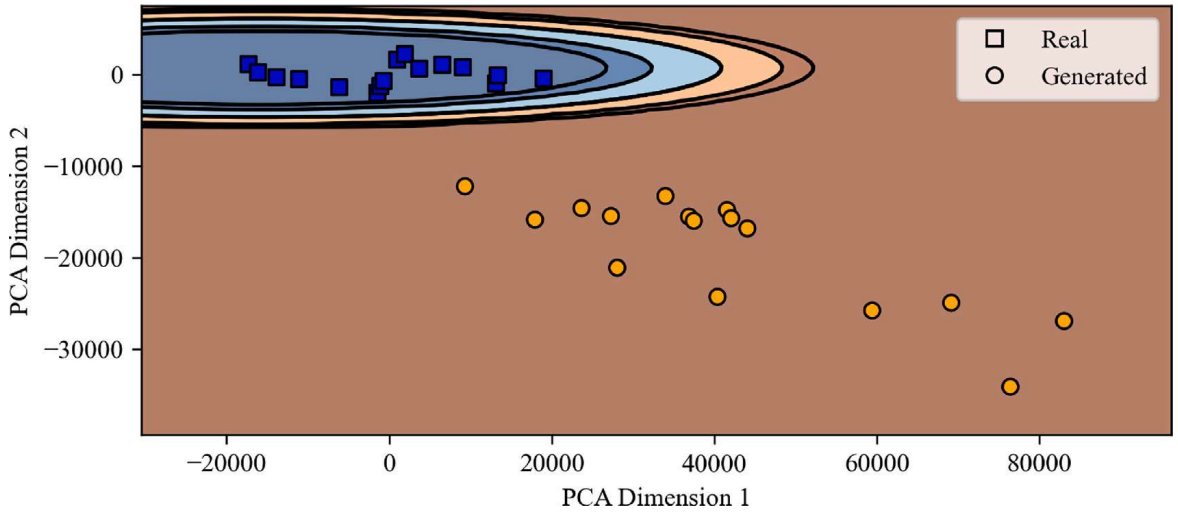

Fig. 14. Plot of Gaussian mixture model predictions over reference and generated data. x and y axes are PCA dimension reduced from the first ten metrics in Table 3 where the square marker represents real data and the circle marker represents generated data. Color represents group classification chosen by the Gaussian mixture model. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5. Conclusion

This paper proposes a solution for the challenges related to generating sufficient data for training state-observers on high-rate events via Conditional Wasserstein Generative Adversarial Networks (CWGAN). Training on a sample of experiments, the generator is able to use conditional labels to find a mapping across the latent space that produces data statistically similar to given signals. The use of conditional labels allows for the generation of multi-modal signals that are a fusion of the statistical properties in discrete signals in the data set. As the generated multi-modal signals were not in the training set, they can be thought of as being analogous to sensor data obtained at unmeasured locations. The developed CWGAN was used to generate signals based on a training library of time-series impact test from an electronics package subjected to shock tests. The statistical measures and FFT scores suggest that the model is able to produce data similar to the training data. The generator does not appear to have evidence of mode collapse since the generated signals are distinct and variable. This is true both between and within classes. The code developed by this project and its pre-trained models are available as an open-source library within this articles supplemental documents and via a public repository [18]. The results obtained in this work are encouraging. Different choices of network architecture, optimization method, and learning rate combinations between the critic and generator can lead to vastly different results. More investigations along these lines will be explored in the near future.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is publicly available. Link to Data: https://github.com/High-Rate-SHM-Working-Group/Dataset-1-High-Rate-Drop-Tower-Data-set, Link to Model: https://github.com/High-Rate-SHM-Working-Group/Dataset-1a-Shock-Test-GAN-model

Acknowledgments

This material is based upon work supported by the Air Force Office of Scientific Research (AFOSR), United States through award no. FA9550-21-1-0083 and no. FA9550-21-1-0082. This work is also partly supported by the National Science Foundation, United States Grant numbers 1850012, 1956071, and 1937535. The support of these agencies is gratefully acknowledged. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the United States Air Force (Distribution A. Approved for public release; distribution unlimited (AFRL-2022-5120)).

References

[1] J. Hong, S. Laflamme, J. Dodson, B. Joyce, Introduction to state estimation of high-rate system dynamics, Sensors 18 (2) (2018) 217, http://dx.doi.org/10.3390/s18010217.

[2] J. Dodson, A. Downey, S. Laflamme, M. Todd, A.G. Moura, Y. Wang, Z. Mao, P. Avitabile, E. Blasch, High-rate structural health monitoring and prognostics: An overview, in: IMAC 39, 2021.

[3] J. Hong, S. Laflamme, L. Cao, J. Dodson, B. Joyce, Variable input observer for nonstationary high-rate dynamic systems, Neural Comput. Appl. 32 (9) (2018) 5015–5026, http://dx.doi.org/10.1007/s00521-018-3927-x.

[4] A. Downey, J. Hong, J. Dodson, M. Carroll, J. Scheppegrell, Millisecond model updating for structures experiencing unmodeled high-rate dynamic events, Mech. Syst. Signal Process. 138 (2020) 106551, http://dx.doi.org/10.1016/j.ymssp.2019.106551.

[5] A. Schenker, I. Anteby, E. Gal, Y. Kivity, E. Nizri, O. Sadot, R. Michaelis, O. Levintant, G. Ben-Dor, Full-scale field tests of concrete slabs subjected to blast loads, Int. J. Impact Eng. 35 (3) (2008) 184–198, http://dx.doi.org/10.1016/j.ijimpeng.2006.12.008.

[6] G. Ben-Dor, A. Dubinsky, T. Elperin, Engineering Models in High-Speed Penetration Mechanics and their Applications, World Scientific, 2018, http://dx.doi.org/10.1142/11077.

[7] G. Ben-Dor, A. Dubinsky, T. Elperin, Ballistic impact: Recent advances in analytical modeling of plate penetration dynamics–A review, Appl. Mech. Rev. 58 (6) (2005) 355–371, http://dx.doi.org/10.1115/1.2048626.

[8] G. Ben-Dor, T. Elperin, A. Dubinsky, High-Speed Penetration Dynamics: Engineering Models and Methods, World Scientific, 2013.

[9] E. Brophy, Z. Wang, Q. She, T. Ward, Generative adversarial networks in time series: A survey and taxonomy, 2021, arXiv:2107.11098.

[10] K.G. Hartmann, R.T. Schirrmeister, T. Ball, EEG-GAN: Generative adversarial networks for electroencephalograhic (EEG) brain signals, 2018, arXiv:1806.01875.

[11] S. Park, D.K. Han, H. Ko, Sinusoidal wave generating network based on adversarial learning and its application: Synthesizing frog sounds for data augmentation, 2019, arXiv:1901.02050.

[12] S. Cai, L. Zhao, Y. Ban, T. Narumi, Y. Liu, K. Zhu, GAN-based image-to-friction generation for tactile simulation of fabric material, Comput. Graph. 102 (2022) 460–473, http://dx.doi.org/10.1016/j.cag.2021.09.007, URL https://www.sciencedirect.com/science/article/pii/S009784932100193X.

[13] P.-H. Kuo, S.-T. Lin, J. Hu, DNAE-GAN: Noise-free acoustic signal generator by integrating autoencoder and generative adversarial network, Int. J. Distrib. Sens. Netw. 16 (5) (2020) 1550147720923529, http://dx.doi.org/10.1177/1550147720923529.

[14] S. Jayalakshmy, G.F. Sudha, Conditional GAN based augmentation for predictive modeling of respiratory signals, Comput. Biol. Med. 138 (2021) 104930, http://dx.doi.org/10.1016/j.compbiomed.2021.104930, URL https://www.sciencedirect.com/science/article/pii/S0010482521007241.

[15] I.J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial networks, 2014, arXiv:1406.2661.

[16] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein GAN, 2017, arXiv:1701.07875.

[17] M. Mirza, S. Osindero, Conditional generative adversarial nets, 2014, arXiv:1411.1784.

[18] Z. Thompson, A. Downey, J. Bakos, J. Wei, J. Dodson, Dataset 1a shock test GAN model, 2022, URL https://github.com/High-Rate-SHM-Working-Group/Dataset-1a-Shock-Test-GAN-model.

[19] J. Dodson, J. Hong, A. Beliveau, Dataset 1 high rate drop tower data set, 2019, URL https://github.com/High-Rate-SHM-Working-Group/Dataset-1-High-Rate-Drop-Tower-Data-set.

[20] J. Hong, S. Laflamme, J. Dodson, Study of input space for state estimation of high-rate dynamics, Struct. Control Health Monit. 25 (6) (2018) http://dx.doi.org/10.1002/stc.2159.

[21] T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, et al., KerasTuner, 2019, https://github.com/keras-team/keras-tuner.

[22] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, 2016, arXiv:1606.03498.

[23] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, S. Hochreiter, GANs trained by a two time-scale update rule converge to a local Nash equilibrium, in: Advances in Neural Information Processing Systems. Vol. 30, NIPS 2017, 2017, arXiv:1706.08500.

[24] B.C. Jeon, J.H. Jung, B.D. Youn, Y.-W. Kim, Y.-C. Bae, Datum unit optimization for robustness of a journal bearing diagnosis system, Int. J. Precis. Eng. Manuf. 16 (11) (2015) 2411–2425, http://dx.doi.org/10.1007/s12541-015-0311-y.

[25] Z. Thompson, A.R. Downey, J.D. Bakos, J. Wei, Synthesizing dynamic time-series data for structures under shock using generative adversarial networks, in: Data Science in Engineering, Volume 9, Springer International Publishing, 2022, pp. 135–142, http://dx.doi.org/10.1007/978-3-031-04122-8_16.