Improving Time Series Generation and Self-Supervised Learning via Instance-Conditioned Contrasting

Philipp Engler[1,2], Nico Müller[2], Ludger van Elst[1], Sheraz Ahmed[1], Andreas Dengel[1,2]
[1]*German Research Center for Artificial Intelligence (DFKI), Kaiserslautern, Germany*
[2]*Department of Computer Science, RPTU Kaiserslautern-Landau, Kaiserslautern, Germany*
{first.last}@dfki.de

***Abstract*—Data sparsity is a major limiting factor for machine learning applications in industry. In order to make such applications feasible and also cost-effective, data synthesis strategies and self-supervised approaches can be employed, addressing the issue in different ways. We aim to advance both directions by combining generative approaches for time series synthesis and self-supervised learning techniques, yielding benefits for both. In this paper, we propose a self-supervised contrastive module for instance-conditioned generative adversarial networks (IC-GAN) and apply it to multiple time series datasets. This module improves the quality and diversity of generated time series without the need for data annotations. Simultaneously, the contrastive module can act as a self-supervised pre-training method of its own. Generated samples can also be used as augmentation in self-supervised techniques by injecting noise into the condition. We achieve significant improvements over the standalone IC-GAN, increasing the classification accuracy of a classifier trained on synthetic data by 5.5 percentage points on average across the 22 shortest time series datasets from the UCR classification archive. Even being trained without data annotations, our GAN is competitive with recent class-conditional time series generation models and generates samples orders of magnitudes faster than a diffusion model at comparable sample quality.**

***Index Terms*—Time Series Analysis, Data augmentation, Generative adversarial networks.**

I. Introduction

Machine learning has made big leaps in recent years, reaching and surpassing human performance in various tasks, increasing their areas of application throughout all industries. However one limitation remains. Training a powerful machine learning model from scratch requires large amounts of data. As industrial applications often are highly task-specific and data is generally not shared with the public or competitors, data mostly must be gathered independently. Collecting data can be rather expensive, especially if the data also needs to be annotated. Thus, the amount of available data is often scarce. This hinders many applications of machine learning approaches in industry. Two general directions to mitigate this issue are data augmentations and self-supervised learning (SSL). Data augmentation artificially expands the existing data, e.g. using random transformations or by synthesizing data. Self-supervised learning allows to leverage unlabeled data, which may be more available or easier to acquire at larger scales. We aim to advance both of these directions with an emphasis on time series data, which is one of the most common data modalities in industrial applications. While diffusion models become more and more prevalent in generative tasks, generative adversarial networks (GANs) still find application in specialized tasks [1], [2] and data augmentation [3], [4]. GANs can yield high quality synthetic data and can generate samples fast at inference. However, they suffer from stability issues during training and are prone to mode collapse [5].

In this paper, we adapt instance-conditioned GANs [6] to time series data and propose a contrasting module as an extension in order to improve training stability and increase diversity of generated samples. We evaluate the contrasting module on four datasets against the standalone IC-GAN, which we adapt to the time series domain. The results show that the contrasting module improves the quality of the generated data on all tested metrics in comparison to IC-GAN. We further show benefits over other GAN, variational autoencoder and diffusion approaches. Besides data synthesis, we then also apply our model for self-supervised learning. The contrasting module can fine-tune the self-supervised encoder of the GAN, refining its representations. We further apply the GAN as an augmentation method in a self-supervised learning scenario by injecting noise to the representations of the encoder, thus showing its capability to replace hand-crafted transformations. We compare our method to various self-supervised learning techniques on multiple time series datasets and find significant advantages. The main contributions can be summarized as follows:

- We adapt IC-GAN to time series data and propose a contrasting module, which improves the quality and diversity of generated data
- We compare the improved GAN to class-conditional approaches showing that it can compete with recent methods, even without being trained on annotated data
- Our generator can be used as an augmentation method, showing advantages over naive transformations or the standalone IC-GAN generator in augmenting time series for self-supervised learning

The datasets used in this paper all are publicly available. Our code is available on GitHub [1].

[1]https://github.com/Philipp-Engler/Contrastive-ICGAN/

II. Background and Related Work

A. Contrastive Learning

The contrastive learning approach in this paper is based on the contrastive loss as used in SimCLR [7]. SimCLR is a well-known self-supervised contrastive learning approach. Its core idea for learning a representation is creating two augmented views of each data sample from a mini-batch, then maximizing the cosine similarity between their feature representations while minimizing the similarity to representations from the other data samples. This is done with a normalized temperature-scaled cross entropy loss (NT-Xent):

$$L = \sum_{(i,j)\in C^+} -\log \frac{\exp(\text{sim}(e_i, e_j)/\tau)}{\sum_{(i,k)\in C} \exp(\text{sim}(e_i, e_k)/\tau)} \quad (1)$$

with feature representations e_i, the set of positive pairs C^+, which contains the pairs of augmented views from the same original data sample and the set of all pairs $C = C^+ \cup C^-$, which also include the negative pairs C^-, i.e. pairs of views from different original data samples within a given mini-batch.

B. Instance-Conditioned GAN

To condition our generative adversarial network on instances, we follow IC-GAN by Casanova et al. [6]. The condition is obtained as the feature representation of a reference instance using an encoder model trained in a self-supervised way. The obtained condition is given to the generator and the discriminator replacing the label in a class-conditional GAN architecture. The discriminator receives either generated samples of the reference instance or nearest neighbors of the reference from the training data. IC-GAN has been developed for image data. To the best of our knowledge no adaptations to time series generation have been developed.

C. Generative Models for Data Augmentation

As generative models become better and better at generating realistic data, there is much work on data augmentation based on generative models, especially in computer vision [8]–[13]. In this paper, we focus on unlabeled data augmentation for self-supervised learning tasks.

Astolfi et al. propose $\text{DA}_{\text{IC-GAN}}$ [8], which is an augmentation strategy based on IC-GAN, but for image data. Similarly to our approach, they generate augmented data samples from a pre-trained instance-conditioned GAN. To control the diversity of generated samples, the variance of the latent noise is modified. The main differences to our approach are that our method is tailored towards time series data and we use a contrastive loss to refine the generator and increase the diversity of the generated samples. Additionally the contrastive loss improves training stability of the GAN, which is important especially on the time-series datasets, while $\text{DA}_{\text{IC-GAN}}$ uses a pre-trained IC-GAN model and keeps the weights frozen.

Li et al. propose GenView [12], a diffusion approach for augmenting image data. A pre-trained CLIP image encoder extracts latent features for a given image. A pre-trained stable diffusion image generator then creates an augmented view of the reference image. To increase diversity of generated samples, Gaussian noise is added to the representations. The strength of the noise is adjusted depending on a prediction of the amount of foreground content in the image. This loss is then used for training a self-supervised model. The overall goal is similar to ours. However, the method is based on image data. Our approach is targeting time series. Additionally, our feedback is used to improve the generative model itself, while GenView only targets the self-supervised task, keeping the weights of the generator model frozen.

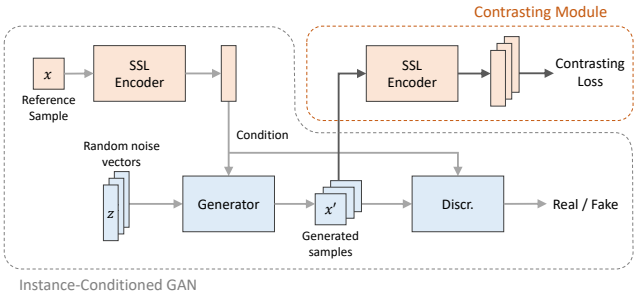


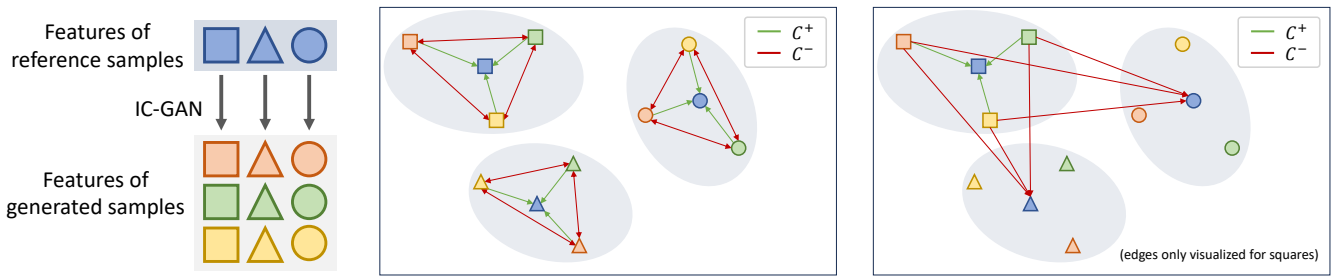
Fig. 1: Overview of IC-GAN with the proposed contrasting module. The contrasting loss may either be used as generator feedback or for fine-tuning the SSL encoder model.

III. Method

In this section we outline our method, consisting of an (instance-conditioned) generative adversarial network and our proposed contrasting module. Two contrasting strategies with different goals are introduced aiming to improve the GAN and train a self-supervised encoder model. The overall architecture is shown in Fig. 1.

A. Instance-Conditioned GAN

We transfer the instance-conditioned GAN concept [6] to time series. As GANs often suffer from stability issues, especially on the tested datasets [14], [15], we probe two different backbone architectures for the generator and discriminator model—a fully connected architecture and a convolutional approach. In either case, a self-supervised encoder model (*SSL Encoder*) extracts feature representations from reference samples, which then act as the condition for generator and discriminator model. Casanova et al. [6] train the encoder model on ImageNet using SwAV [16] in the self-supervised case. For our time-series data, we opt for SimCLR [7] using suitable transformations. Specifically, we use random scaling, masking, noise and permutations in order to obtain augmented views of time series samples. As the time series datasets differ vastly, we pretrain an individual encoder model on the training data of each respective dataset. Obtaining a universal encoder like a foundation model from a large, merged dataset from multiple different domains remains an interesting prospect for future work. Our results do not suggest any benefit on the time series datasets from using nearest neighbors instead of the reference sample. Thus, we use the reference sample directly.



(a) Generating samples (b) Local Contrasting (c) Global Contrasting

Fig. 2: Schematic overview of the local and global contrasting concepts.

B. Contrasting Module

The contrasting module maps the generated samples to representation space again using the SSL encoder. A contrastive objective is used in addition to the GAN objective during training. Different contrastive objectives are used for fine-tuning the encoder or refining the generator model.

1) Local Contrasting (LC): Local contrasting is performed across generated samples of a single reference sample only, thus on a local scale in feature space. The embeddings of generated samples for the same reference aim to be close to the reference embedding, but at the same time as different as possible from each other (cf. Fig. 2b). This is realized by a contrastive learning objective, similar to SimCLR [7]. The loss follows the same formulation as in Equation 1 and only differs in the choice of the sets C^+ of positive pairs and C^- of negative pairs. A pair (i, j) is in C^+, if j is the corresponding reference sample that the generated sample i was conditioned on. A pair (i, j) is in C^-, if i and j are both generated samples from the same condition. The local contrasting loss is used for training the generator model – supplementary to the GAN loss. It is aimed to increase diversity among the generated samples as high similarity is penalized across generated samples with the same condition. That shall make the generator more utilitarian for data augmentation purposes and counteract mode collapse.

2) Global Contrasting (GC): Global contrasting is performed across all reference samples of a mini-batch. The embeddings of generated samples for the same reference should again be close to the reference embedding, but now they are targeted to be far away from the other reference samples (cf. Fig. 2c). Again, the loss is based on the formulation from Equation 1. In this case, a pair (i, j) is again in C^+, if j is the corresponding reference sample that the generated sample i was conditioned on. A pair (i, j) is in C^-, if j is a different reference sample that i was not conditioned on, but from the same mini-batch. The global contrasting loss may be used as a signal for training the generator, where it may improve fitness of generated samples to the condition.

Further, the global contrasting loss can be used to fine-tune the encoder model. A fine-tuned encoder model may yield higher quality conditions for the generator and it can also be used directly as feature extractor in a downstream task, making global-contrasting a self-supervised learning method on its own. When fine-tuning the SSL encoder, only the encoder module from the contrasting module - the online encoder - is updated directly via the global contrasting loss. The conditioning encoder from the IC-GAN is updated as an exponential moving average of the online encoder, allowing the GAN to slowly adapt to the changing condition space.

3) Feature Augmentation: In order to leverage the instance-conditioned GAN, we propose to augment the encoded conditions by randomly scaling each dimension. For scaling we sample a factor using a Gaussian distribution with mean 1 and a standard deviation σ^2, which can be optimized as a hyperparameter. This can be written as:

$$\boldsymbol{e}_i^{aug} = \boldsymbol{e}_i \odot \mathcal{N}(1, \sigma^2 \boldsymbol{I}) \tag{2}$$

with the feature representation $\boldsymbol{e}_i$ and augmented representation $\boldsymbol{e}_i^{aug}$.

IV. Experimental Setup

In this section, we give details of our implementation, the evaluation strategy and datasets used.

A. Implementation Details

1) Model Architecture: We apply our method on two different architectures, one is based on a multilayer perceptron (MLP) and the other one on dilated 1D-convolutional layers.

a) MLP Model: In the MLP architecture the generator model is a feed-forward network consisting of six fully connected layers with ReLU activations in between. As input a random 64-dimensional noise vector and the 320-dimensional reference representation are concatenated and linearly mapped to 64 dimensions. The discriminator model consists of three fully connected layers with ReLU activation. In the discriminator model the condition is concatenated to the input time series.

b) Convolutional Model: The convolutional architecture is based on dilated 1D-convolutions. Similarly to the MLP architecture, the generator samples a random noise vector and concatenates the condition vector. Then it is linearly mapped down to 64 dimensions again. Two fully connected layers follow and the output is reshaped to a time series of one eighth

of the target length with 128 channels. A learned positional encoding is concatenated on top as for some datasets the. Then the intermediate representation is fed into five 1D transposed convolution layers with leaky ReLU activation, similarly to the architecture of SNS-GAN [17]. Leaky ReLU is used to mitigate issues with dying neurons in early, unstable phases of the training. The discriminator model consists of five layers of 1D convolutions with spectral normalization [18] and leaky ReLU activation. For conditioning we follow the projection discriminator by Miyato and Koyama [19].

c) SSL Encoder: For the SSL encoder, we use the temporal convolutional architecture from TS2Vec [20]. However, we reduce the number of residual blocks from ten down to six and the initial projection is omitted.

2) Training Details:

a) GAN Training: The GANs are trained using a WGAN loss with gradient penalty (WGAN-GP) [21]. A batch-size of 1000 is used, but at most the size of the training set. The GAN is trained for 60k iterations using an Adam optimizer with a step-wise learning rate decay and a linear warmup. The learning rate is optimized in values from 0.0003 to 0.003 and the gradient penalty weight for the WGAN-GP loss [21] is optimized between 0.1 and 1 for the different datasets. These hyperparameters are optimized based on metrics evaluated on training data, averaging over multiple runs. Due to stability improvements, these parameter ranges have been reduced in the convolutional model.

b) Contrasting Module: For the contrasting module, there are also some hyperparameters. We use $\tau = 0.5$ for the temperature parameter in either contrastive loss. The loss from local and global contrasting is added to the WGAN-GP loss with a scaling factor each. These are also optimized per dataset in ranges from 1 to 100. To reduce the computational overhead from the contrasting module, a subset of samples is randomly chosen from each mini-batch for calculating and back-propagating the contrastive losses. The fraction of samples is adjustable. We find a fraction of 20% as a fair trade-off between efficiency and effectiveness. For fine-tuning the SSL encoder with global contrasting, the online-encoder in the contrasting module is trained with a learning rate of 10^{-4} and the condition encoder from is updated with an exponential moving average using a decay of 0.999 for the MLP architecture. In the convolutional model learning rates from 10^{-5} to 10^{-4} are tested and the decay is increased up to 0.99999, further slowing down changes on the condition encoding. We have found that it is advantageous to update the encoder only towards the end of training, when the generator already produces higher quality samples, as updating the encoder directly with poor generated samples makes it more difficult for the generator to learn and disrupts the representation space with low quality feedback. Thus, we only perform fine-tuning during the last 25% of training.

c) SSL Encoder: The self-supervised encoder model is pre-trained using the objective from SimCLR [7] (cf. Fig. 3). An Adam optimizer is used with a constant learning rate of 0.0001 and a batch-size of 12. Instead of the image transformations of the original implementation, we apply suitable transformations for time series, meaning random scaling, Gaussian noise, permutations and masking. The magnitude or probability of these transformations is optimized based on the accuracy in a classification downstream task on a per-dataset basis. Each dataset required different strengths of transformations. This also motivates the use of GAN-based transformations as these can adapt to the natural variations in the dataset.

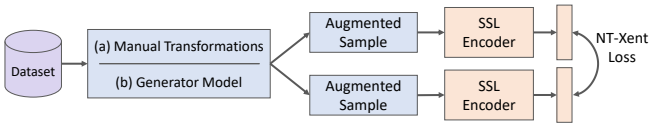


Fig. 3: Overview of SSL Encoder pre-training. Hand-crafted transformations are used to augment data samples. Manual transformations can be replaced with the generator model, generating augmented samples based on the instance-condition.

B. Evaluation Strategy

Our evaluation covers different aspects targeting different use cases for the instance-conditioned GAN. We evaluate different GAN metrics based on generated samples to assess the quality of the synthetic data and possible benefits as an augmentation method. As the instance-conditioned GAN is trained without labels, applications in self-supervised learning scenarios are possible. We either fine-tune the encoder model using the contrasting module and assess the the accuracy in a downstream classification task or we use the generator as an augmentation method in another self-supervised learning task replacing hand-crafted transformations.

1) GAN Metrics: The performance of the GAN is evaluated using the Fréchet Inception Distance (FID) [22], Inception Score (InS) [23] and TSTR classification accuracy scores. TSTR refers to training a classifier on synthetic data and then evaluating on the real test set. In contrast to computer vision, these metrics are not evaluated on Inception v3 using ImageNet, but instead on InceptionTime using the respective datasets [5]. The evaluation aims to analyze benefits of the contrasting module towards the quality of the generated data by the GAN. All results are averaged over three evaluation runs on the test set, using the hyperparameters found by optimizing on training set metrics.

2) Fine-tuned SSL Encoder: The performance of the SSL encoder is assessed after fine-tuning with global contrasting, which is fully self-supervised. The evaluation precisely follows TS2Vec [20]. A SVM with RBF kernel is trained with training data representations and the classification performance is evaluated using the test data (cf. Fig. 4). We analyze improvements over the pre-trained SSL encoder based on SimCLR [7] and compare to other self-supervised learning methods, namely T-Loss [24], TNC [25] and TS2Vec [20].

3) GAN as Augmentation Method: Lastly, the GAN is evaluated as an augmentation method. We train a new SSL encoder

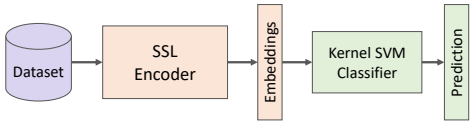


Fig. 4: Evaluation of a self-supervised encoder model. A kernel SVM is trained based on embeddings from the encoder.

using SimCLR [7], but instead of the naive transformations from before, we augment the dataset by generating new samples with the generator (cf. Fig. 3). The condition encoder obtains a feature representation for the given reference sample. We then scale the different dimensions of the representation using our feature augmentation with a standard deviation $\sigma^2 \in [0.03, 0.3]$. With these noisy representations multiple augmented views are generated. The evaluation aims to show that the GAN can be used as an augmentation method for self-supervised learning, overcoming the need for individual selection of hand-crafted transformations. Again we compare to our pre-trained SSL encoder based on SimCLR and the other self-supervised learning methods.

C. Datasets

Our approach is evaluated on multiple datasets from the UCR time series classification archive [26], which contains dataset of different domains. The datasets used contain between 2 and 50 classes and between 20 and almost 10.000 training samples. The fully connected architecture is evaluated on a selection of four datasets from various domains that exhibit different properties. These datasets are ChlorineConcentration (abbreviated as Chlorine), Crop, OSULeaf and WordSynonyms (abbreviated as WordSyn). Expanding the evaluation, the convolutional architecture is evaluated on the 22 datasets from the UCR time series classification archive that have less than 100 time steps, excluding datasets with missing values.

V. Results and Discussion

In this section, we present our results for both architectures and the different evaluation strategies.

A. GAN Metrics

For evaluation of the GAN metrics, the different contrasting strategies are used for generator feedback.

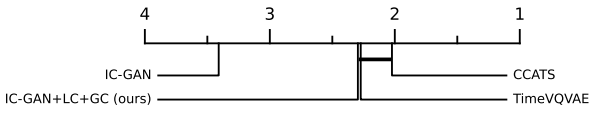
1) MLP Model: The average results across the four datasets for the MLP architecture are presented in Table I. The proposed contrasting strategies show benefits in different aspects. Local contrasting shows better performance on the TSTR classification accuracy score, while global contrasting shows large improvements on the Fréchet Inception Distance, for which lower values are better, and slight improvements on the Inception Score. An explanation could be that the local contrasting increases the diversity of the generated samples, acting as a stronger augmentation compared to the IC-GAN. This may come at the cost of sample quality and semantic similarity to the reference. Joining local contrasting and global contrasting appears to combine and even amplify the advantages of both strategies. The results improve further beyond local or global contrasting alone.

Local contrasting measurable increases the diversity of generated samples. We compute the average cosine similarity in feature space between data samples generated from the same reference sample. We find that local contrasting decreases the sample-to-sample similarity for ChlorineConcentration from 99.9% to 99.7%, for Crop from 99.6% to 84.5%, for OSULeaf from 99.8% to 94.9% in comparison to the standalone IC-GAN. This indicates an increase in generated sample diversity, which may explain the classification accuracy improvements on these datasets. Only on WordSynonyms, the similarity is slightly increased from 99.5% to 99.8%. However, on this specific dataset, the TSTR classification accuracy also is not increased with local contrasting.

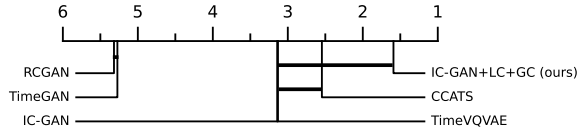
2) Convolutional Model: Based on the previous results, we jointly use local and global contrasting as generator feedback in our convolutional architecture. We use the instance-conditioned GAN without contrasting module as baseline and additionally compare the results to RCGAN [27], TimeGAN [28], TimeVQVAE [14] and CCATS [15], covering GAN, VAE and diffusion approaches. The average ranks are computed on the different metrics and the results are summarized as critical difference diagrams in Fig. 5.

The local and global contrasting shows significant improvements over the baseline and competes with class-conditional generative models. After finding significant differences across methods using the Friedmann test on all metrics, pairwise Wilcoxon signed-rank tests with Holm correction for multiple comparisons at $\alpha = 0.05$ are conducted to test for statistical significance. On the TSTR classification accuracy score, IC-GAN with contrasting significantly outperforms standalone IC-GAN. The average classification accuracy is 78.4% compared to 72.9%, which is a notable improvement of 5.5 percentage points. In addition to quality improvements of the generated samples, we again find that the average cosine similarity between generated samples is lowered from 0.990 to 0.978 due to the contrasting, while there is no decrease in similarity to the reference sample on average. This suggests increased diversity of the generated samples while remaining consistent with the condition, which then explains the gains in accuracy on the downstream classification task. Compared to CCATS and TimeVQVAE, there is no significant difference in TSTR classification accuracy, but CCATS on average ranks higher.

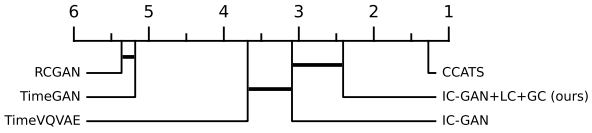
On the Fréchet inception distance, the the use of local and global contrasting consistently improves the performance, such that our model significantly outperforms the baseline IC-GAN. The FID is decreased on 21 out of 22 datasets. Although the uncorrected Wilcoxon test yields $p < 0.05$ between our GAN and TimeVQVAE, the difference does not remain significant after Holm correction for multiple comparisons. While there is no significant gap between our model and CCATS, the average rank of our GAN is 1.6 compared to 2.5, competing well with the diffusion model.



(a) TSTR Classification Accuracy



(b) Fréchet Inception Distance



(c) Inception Score

Fig. 5: Critical difference diagrams comparing ranks between IC-GAN with and without contrasting to various other methods across different evaluation metrics.

TABLE I: Metrics for (fully connected) GAN evaluation averaged over four datasets. Contrasting is used as generator feedback.

Method	TSTR [%]	FID	InS
IC-GAN [6]	71.4 ± 0.8	4.12 ± 0.30	8.45 ± 0.09
LC (ours)	74.0 ± 1.6	5.01 ± 0.42	8.17 ± 0.11
GC (ours)	72.2 ± 2.0	2.90 ± 0.28	8.67 ± 0.09
LC + GC (ours)	$\mathbf{75.4 \pm 0.7}$	$\mathbf{2.31 \pm 0.16}$	$\mathbf{8.84 \pm 0.07}$

The instance-conditioned GAN shows a disadvantage on the inception score. While our model with contrasting module significantly outperforms RCGAN, TimeGAN and TimeVQVAE, CCATS does have a significant advantage over our model. While our IC-GAN model generates data of good quality—as indicated by the FID results—it is only conditioned on instances, not the ground truth labels. The competing methods, TimeVQVAE and CCATS are using labels directly for conditioning the GAN. This way, they can learn to generate class-specific details, which benefits a high inception score. Being based on the output of a classifier model, the inception score favors data that result in a highly specific output for each sample. IC-GAN instead is not dependent on class-labels. This has the advantage that it can be trained on unlabeled data and thus can be used for self-supervised learning and in weakly supervised scenarios where much more unlabeled data than labeled data is available. On the other hand, this has the disadvantage that generated samples do not have the same strict class-distinctions.

For the GAN metrics, we can conclude that the contrasting module offers significant benefits to instance-conditioned GANs. The quality of the generated data can compete with existing class-conditional time series generation techniques, but the generated data can be less class-specific. In addition, our GAN generates data magnitudes faster than the diffusion approach CCATS, while achieving competitive sample quality. On the example of the ElectricDevices dataset, our GAN takes 10 ms to generate 10k samples on an NVIDIA A100 80GB GPU. In contrast, CCATS takes about 14 minutes for the same amount of data. This is because CCATS requires many forward passes for the generation process and additionally does many more operations per forward pass.

B. Fine-tuned SSL Encoder

Global contrasting is used for fine-tuning the SSL encoder during GAN training. The representations of the SSL encoder are evaluated on a downstream classification task and results are compared to the baseline SimCLR model the encoders are based on as well as other self-supervised learning techniques.

1) MLP Model: We apply two different strategies, using the global contrasting objective only for the SSL encoder (GC-Enc) or additionally using it as generator feedback (GC-Full). The results are summarized in Table II.

In the MLP model, fine-tuning the encoder model using global contrasting increases downstream classification accuracy in almost all cases, as shown in Table II. The average accuracy also is significantly higher than for T-Loss [24] and TNC [25] in both fine-tuning variants. GC-Full additionally outperforms TS2Vec on two out of four datasets. This highlights the potential of our contrasting module for self-supervised learning based on generative models. However, it also shows limitations of the approach, as our encoder does not match TS2Vec on all datasets.

2) Convolutional Model: The convolutional model exhibits stability issues when being trained jointly with the encoder. The accuracy in the downstream task only increases on 3 out of 22 datasets. On multiple datasets, the quality of the encoding decreases noticeably. The model does not seem to adapt to the changing condition quickly enough. A possible fix may be to do multiple generator updates after each condition update, but this would largely increase the training effort.

C. GAN as Augmentation Method

Finally, we evaluate the GAN as an augmentation method for self-supervised learning. We train our encoder model using SimCLR, but instead of using naive transformations, we augment training samples by generating semantically similar samples using the generator conditioned on the training sample as reference. We compare our approach to SimCLR with naive transformations, other self-supervised methods and IC-GAN without contrasting module for augmentation.

1) MLP Model: In the MLP model we do not yet apply noise to the feature representations. As shown in Table III, the GAN with local contrasting shows major benefits as an augmentation technique. Comparing the SimCLR-based approaches in the bottom half of the table, we find that local contrasting yields a consistent advantage over the standalone IC-GAN as an augmentation method. This coincides with our observation on the GAN metrics and our hypothesis

TABLE II: Evaluation of SSL encoder using kernel SVM after fine-tuning with global contrasting. Global contrasting is either used as feedback for encoder only (GC-Enc) or for generator and encoder (GC-Full)). Comparison between other SSL methods (top) and fine-tuned SSL encoder (bottom).

Method	SVM classification accuracy [%]				
	Chlorine	Crop	OSULeaf	WordSyn	Avg.
SimCLR [7]	77.5	71.9	86.0	66.1	75.4
T-Loss [24]	74.9	72.2	76.0	69.1	73.1
TNC [25]	76.0	73.8	72.3	63.0	71.3
TS2Vec [20]	**83.2**	**75.6**	87.6	70.0	**79.1**
GC-Enc (ours)	77.0 ± 0.2	73.0 ± 0.1	86.8 ± 0.3	70.5 ± 1.4	76.8 ± 0.5
GC-Full (ours)	78.6 ± 0.1	73.3 ± 0.1	$\mathbf{87.9 \pm 0.5}$	$\mathbf{72.3 \pm 0.4}$	78.0 ± 0.3

TABLE III: Comparison between MLP GAN model as augmentation method in SimCLR and other self-supervised approaches based on hand-crafted transformations. We use the contrasting module for generator feedback.

Method	SVM classification accuracy [%]				
	Chlorine	Crop	OSULeaf	WordSyn	Avg.
T-Loss [24]	74.9	72.2	76.0	69.1	73.1
TNC [25]	76.0	73.8	72.3	63.0	71.3
TS2Vec [20]	**83.2**	**75.6**	87.6	70.0	**79.1**
SimCLR [7] (hand-crafted)	77.5	71.9	86.0	66.1	75.4
IC-GAN [6]	77.6 ± 0.1	72.4 ± 0.0	85.1 ± 0.0	69.8 ± 0.0	76.2 ± 0.0
LC (ours)	79.0 ± 0.1	73.2 ± 0.2	$\mathbf{88.7 \pm 0.2}$	$\mathbf{71.2 \pm 0.2}$	78.0 ± 0.5
GC (ours)	77.5 ± 0.0	71.0 ± 0.4	85.5 ± 0.0	71.1 ± 0.1	76.3 ± 0.1
LC + GC (ours)	79.4 ± 0.2	70.6 ± 0.1	85.5 ± 0.0	69.9 ± 0.0	76.4 ± 0.1

TABLE IV: Comparison between the convolutional GAN model as augmentation method in SimCLR and other self-supervised approaches based on hand-crafted transformations across 22 datasets. We use the contrasting module for generator feedback.

Method	Avg. SVM classification accuracy [%]
T-Loss [24]	82.4
TNC [25]	80.9
TS2Vec [20]	**83.1**
SimCLR [7] (hand-crafted)	80.9
Local Contrasting (ours)	79.5 ± 1.5

that the local contrasting facilitates the generation of more diverse data. The downstream classification accuracy is also increased compared to SimCLR with naive transformations. Global contrasting or combinations of global and local contrasting roughly perform on par with the standalone IC-GAN, indicating that only the local contrasting leads to sufficiently diverse augmentations.

With local contrasting, the GAN-augmented SimCLR variant can compete with TS2Vec on some of the datasets. It outperforms TS2Vec on two datasets, but on others it does not match the performance of TS2Vec. An explanation could be that the contrasting at multiple scales as performed in TS2Vec may have advantages over the contrasting that is only performed at a full sample scale in SimCLR.

2) Convolutional Model: In the convolutional model, we apply noise to the feature representations as described in section III-B3.

Without any hand-crafted transformations, the performance is close to the performance of SimCLR with optimized hand-crafted transformations, as the results in Table IV show. However, a rather large performance gap to TS2Vec [20] remains. The primary advantage of the GAN-augmentations is that no hand-crafted transformations as in TS2Vec are required. However, an interesting prospect for future work can be the application of GAN transformations in the TS2Vec framework, potentially combining advantages of multi-scale contrasting and generative transformations. Also, the GAN may also benefit from the higher quality embeddings for instance-conditions using a TS2Vec encoder instead of a SimCLR encoder.

VI. Conclusion

In this paper, we adapted instance-conditioned GANs [6] to time series data and introduced a contrasting module as a versatile extension. The proposed module provides feedback to the generator and the self-supervised encoder in the form of local and global contrasting. We have shown that local contrasting improves sample diversity. This makes the generator more useful for augmentation tasks, as indicated by improvements in the TSTR classification accuracy and analysis of the sample similarity. Global contrasting improves the quality and semantic fidelity of generated data, showing significant improvements in the Fréchet inception distance and inception score over the standalone IC-GAN. Thus, the two contrasting techniques provide complementary benefits.

Our improved instance-conditioned model outperforms previous class-conditional GAN and VAE approaches without access to class-labels during training, indicating great benefit in unlabeled or weakly labeled scenarios. The model can even compete with the diffusion model CCATS at a fraction of the computation time for generating samples.

Global contrasting can additionally fine-tune the self-supervised encoder model. While this improved the downstream classification accuracy in the MLP architecture over the pre-trained SimCLR model and on some datasets also over TS2Vec [20], we could not confirm similar benefits in the convolutional architecture on a larger set of datasets.

We applied our generator model for data augmentation in a self-supervised task, showing that the GAN augmentation can compete with hand-crafted transformations. On some datasets, SimCLR with GAN transformations is also able to compete with TS2Vec. However, in a more broad evaluation, it does not match the performance of TS2Vec.

The contrasting module can provide various benefits for generative adversarial networks and self-supervised learning tasks. While our method can improve the stability of the GAN training, the method still depends on the success of the GAN at learning to generate the data distribution. Thus, the method requires careful hyperparameter tuning and in cases where the GAN does not learn well, alternative approaches based on diffusion models may be advantageous. This may specifically apply for more complex datasets. For future work, the self-supervised application of the instance-conditioned GAN may be further developed. TS2Vec or other self-supervised techniques may either be extended with GAN augmentations or may be used as an encoder basis for the instance conditions to improve the GAN. Additionally, the possibility to replace the discriminator with the contrasting module entirely remains to be assessed.

REFERENCES

[1] M. Arunachalam, R. Babu, S. Ganesh, and T. Sanjay, "Generative adversarial networks based adaptive modulation and coding for next-generation 5g communication systems," *Discover Applied Sciences*, vol. 7, 01 2025.

[2] D. Ma, D. Yuan, M. Huang, and L. Dong, "Vgc-gan: A multi-graph convolution adversarial network for stock price prediction," *Expert Syst. Appl.*, vol. 236, no. C, Feb. 2024.

[3] L. Alhoraibi, D. Alghazzawi, and R. Alhebshi, "Generative adversarial network-based data augmentation for enhancing wireless physical layer authentication," *Sensors*, vol. 24, no. 2, 2024.

[4] D. Gao, X. Wu, Z. Wen, Y. Xu, and Z. Chen, "Few-shot sar vehicle target augmentation based on generative adversarial networks," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. X-1-2024, pp. 83–90, 2024.

[5] A. Koochali, M. Walch, S. Thota, P. Schichtel, A. Dengel, and S. Ahmed, "Quantifying quality of class-conditional generative models in time series domain," *Applied Intelligence*, vol. 53, 07 2023.

[6] A. Casanova, M. Careil, J. Verbeek, M. Drozdzal, and A. Romero Soriano, "Instance-conditioned gan," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 27 517–27 529.

[7] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," ser. ICML'20. JMLR.org, 2020.

[8] P. Astolfi, A. Casanova, J. Verbeek, P. Vincent, A. Romero-Soriano, and M. Drozdzal, "Instance-conditioned gan data augmentation for representation learning," *ArXiv*, 2023.

[9] H. Bansal and A. Grover, "Leaving reality to imagination: Robust classification via generated datasets," *ArXiv*, 2023.

[10] L. Dunlap, A. Umino, H. Zhang, J. Yang, J. Gonzalez, and T. Darrell, "Diversify your vision datasets with automatic diffusion-based augmentation," *Conference on Neural Information Processing Systems*, 2023.

[11] R. He, S. Sun, X. Yu, C. Xue, W. Zhang, P. H. S. Torr, S. Bai, and X. Qi, "Is synthetic data from generative models ready for image recognition?" *ArXiv*, 2022.

[12] X. Li, Y. Yang, X. Li, J. Wu, Y. Yu, B. Ghanem, and M. Zhang, "Genview: Enhancing view quality with pretrained generative model for self-supervised learning," in *European Conference on Computer Vision*, 2024.

[13] F. R. Rahat, M. S. Hossain, M. R. Ahmed, S. K. Jha, and R. Ewetz, "Data augmentation for image classification using generative ai," *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 4173–4182, 2024.

[14] D. Lee, S. Malacarne, and E. Aune, "Vector quantized time series generation with a bidirectional prior model," in *International Conference on Artificial Intelligence and Statistics*, 2023.

[15] P. Engler, A. Koochali, L. van Elst, A. Dengel, and S. Ahmed, "Ccats: Moving forward with class-conditional time series generation," in *International Conference on Neural Information Processing (ICONIP)*, 2024.

[16] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 9912–9924.

[17] P. Engler, L. van Elst, S. Ahmed, and A. Dengel, "Can generative adversarial networks compete against diffusion models at generating time series?" in *Proceedings of the 18th International Conference on Agents and Artificial Intelligence - Volume 4: ICAART*, 2026.

[18] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018.

[19] T. Miyato and M. Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=ByS1VpgRZ

[20] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, "Ts2vec: Towards universal representation of time series," in *AAAI*, 2022.

[21] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Neural Information Processing Systems*, 2017.

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6629–6640.

[23] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 2234–2242.

[24] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[25] S. Tonekaboni, D. Eytan, and A. Goldenberg, "Unsupervised representation learning for time series with temporal neighborhood coding," in *International Conference on Learning Representations (ICLR)*, 2021.

[26] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, and Hexagon-ML, "The ucr time series classification archive," October 2018, https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

[27] C. Esteban, S. L. Hyland, and G. Rätsch, "Real-valued (medical) time series generation with recurrent conditional gans," *arXiv preprint*, 2017.

[28] J. Yoon, D. Jarrett, and M. van der Schaar, "Time-series generative adversarial networks," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.