GENERATION OF SYNTHETIC EXAMPLES USING GENERATIVE ADVERSARIAL NETWORKS (GAN) TO EXTEND A DATABASE OF FAULT SIGNALS ON POWER DISTRIBUTION LINES

Javier GRANADO
Circe – Spain
jgranado@fcirce.es

Elías HERRERO
University of Zaragoza – Spain
jelias@unizar.es

Andrés LLOMBART
Circe – Spain
allombart@fcirce.es

ABSTRACT

The detection and classification of the type of fault is an essential technique for the improvement of electricity grids due to its potential to improve the reliability of supply and, therefore, its quality. This paper reports a method to obtain an extended database of fault signals in order to use Neural Networks (NN) to process them. The need of a large database for the training process is an inherent need for the right working of a NN. In this type of chaotic nature signals, it is impossible to record enough real ones and, even simulating is near unfeasible task due to the variety of the causes that produces faults events. The proposed solution is to obtain a short database of simulated signals from a real modelled electrical grid and extend this database by means of GAN. This technique simplifies the process to obtain the database of fault signals.

Keywords: GAN, Fault Signals, Distribution Lines.

1 INTRODUCTION

The use of neural networks are giving good results in the field of detection, classification, and localization of faults in distribution lines [1]–[4].
During the corresponding training process, a series of examples of labelled signals are shown to the Neural Network (NN). Then, NN learns certain features in order to recognize a new example shown to it. This training process typically requires tens of thousands of examples to correctly train the NN and avoid the well-known phenomenon of overfitting. When this last happens, the NN learns the few examples shown and is unable to generalise.
In distribution lines, in fault classification and location problems, obtaining a large database of real signals is a very complex process. The fault is usually a catastrophic and chaotic event (tree fall, lightning, etc.), so it is difficult to reproduce, predict, collect, and label.
Model the real distribution line with appropriate software and simulate the faults could be an option. However, obtaining tens of thousands of examples is practically unfeasible because it takes a lot of time. In addition, selecting all the different situations by hand to reproduce all possible events is an almost impossible task.
These are the main reason because data augmentation is intended to increase the number of examples. There are two basic techniques among others, by adding slightly modified versions of existing examples, or by adding synthetic examples created from existing ones. Modifying existing examples is based on applying changes such as rotations, symmetries, scaling, etc. In case of example synthesis, new examples are obtained by using generative adversarial networks (GAN).
Unlike other problems related to image classification (animal detection in photos, etc.) here, the images (Fig. 1) are a transformation [5] of temporal signals[6](Fig. 2). Classical data augmentation techniques cannot be applied since the effects over the information contained in the temporal signal would be unpredictable.

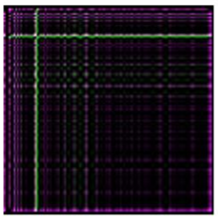
Figure 1. Original signal transformed into image.

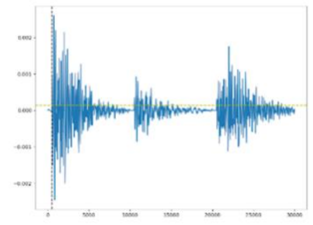
Figure 2. Original time signals

Generative Adversarial Network (GAN) is a branch of machine learning systems discovered in 2014 [7]. In GAN, two neural networks compete against each other.
For an existing database, the GAN will be trained to generate new data with the same characteristics as the training data. For example, a GAN trained with images of people can generate new images that look real to humans, with many lifelike attributes. GAN is composed of two neural networks (Fig.3), a generator and a discriminator.

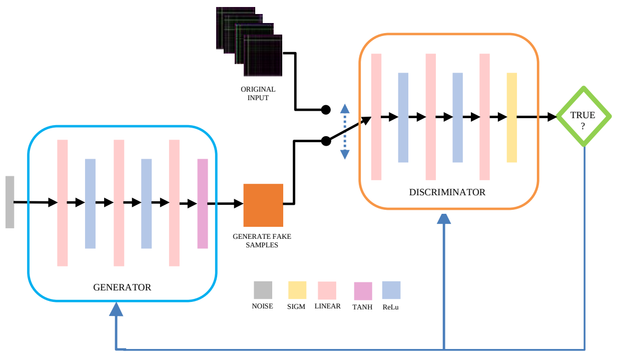

Figure 3. Architecture of the GAN

The generator produces examples while the discriminator evaluates them. Training continues until the discriminator is no longer able to determine whether the displayed example is real or generated. The training objective of the generator is to decrease the discriminator's error (i.e., to "fool" the discriminating system by creating new examples that the discriminator believes to be real).

A set of real data examples serves as a reference for the discriminator. Generally, the generator generates the examples from random information. The examples generated are then evaluated by the discriminator. Backpropagation is applied in both systems. While the discriminator becomes progressively more adept at detecting fake images, the generator system becomes progressively more adept at generating better images.

Some of the most useful applications of GANs are enhancing the resolution of images [8], modifying the appearance of an image [9][10], or generating images synthetically [11][12].

Here we find several types of GAN: Linear GAN, Convolutional GAN (CGAN) [13], Conditional GAN (cGAN) [13], even Conditional Deep Convolutional GAN (cDCGAN)[14].

In this paper, we intend to address data augmentation using GAN[15][16]. In this way, we want to introduce versatile examples. Here, it is the GAN itself during its training, which manages to extract the hidden information, and by generating each new example from noise, the GAN adds variability without losing the essence that characterizes each class (type of fault).

2 PROPOSED SOLUTION

This section explains the complete process (Fig. 4). It is divided into two parts: transformation of time signals to images and generation of examples using GAN.

In our case, we start from time signals. These signals come from the response of the electrical network to pulse injection (TDR technique). For each example, we have three signals (coming from each of the R-S-T phases of the electrical network). After digitization, we have 12,000 samples (4,000 x 3). Then, signals are transformed into images for better and easier data processing by convolutional neural networks.

The transformation used is *GAF* (Gramian Angular Field). In the GAF transformation, the time series is represented in a polar coordinate system. The amplitude of the signal is encoded as the angular cosine, and the time stamp as the radius. This information is collected in the form of a matrix in which the relationships between the different time instants can be identified. In this matrix, each element is either the cosine of the sum of the angles (GASF) or the sine of the difference of the angles (GADF). The GASF transform has the advantage over the GADF transform that the time signal can be reconstructed from the image. This property makes it possible to compare the original signals with the transformed ones.

In the process of transformation from time series to images, we can also reduce the dimensionality of the images. The algorithm used is called *PAA* (Piecewise Aggregate Approximation). With PAA, we try to reduce the dimensionality as much as possible to improve the subsequent training processes.

We have processed the images starting from their initial dimensionality (4000x4000x3) and we have progressively reduced it to a dimensionality of 128x128x3. With this reduction we have been able to reduce the training time of the GAN, as well as the subsequent processing by NN without reducing the accuracy in the classification of the different classes.

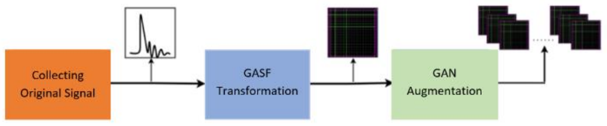

Figure 4. Process block diagram

At the end, GAN has been used to generate the extended database.

GANs are difficult neural networks to training and suffer from some inherent problems in their structure [15][16]. One of them is mode collapse. In this case, the generator learns to generate a few examples from the data distribution but fails to learn many others. In the worst case, the generator simply produces a single example.

Another typical problem with GANs is the vanishing gradient problem. This occurs when training these GANs and the gradient becomes infinitely small. This, in the worst case, can cause the neural network to stop training and evolving to the minimum.

On the other hand, and as already explained, GANs are successfully used in applications where it is required to generate examples of a very high quality, which are practically indistinguishable from the real ones. To achieve this, we would need to train GANs with complex structures (cGAN, cDCGAN, etc...).

Having said that, and having the chance to choose different alternatives, the GAN tested in this work are based on:

- Linear GAN,
- Conditional GAN (cGAN)
- Convolutional GAN (CGAN)
- Conditional Deep Convolutional GAN (cDCGAN)

In general, CGANs are more powerful for image training than linear GANs, although this power also has a trade-off in training time and parameterization difficulty.

On the other hand, Conditional GANs (cGAN and cDCGAN) have the advantage that all classes can be trained at the same time. This is done by introducing the class type as additional information to the generator.

This, a priori, is an advantage, but this type of GAN has a greater tendency to present mode collapse. In our case, we had to train a linear GAN by training each class separately.

4 RESULTS

First, we tried to implement a Conditional Deep Convolutional Generative Adversarial Networks (cDCGAN) to try to train all classes at once.
We tried first it with some known state-of-the-art database such as MNIST. With this database, the GAN converges correctly.
However, with our database, we had the problem of mode collapse. Mode collapse manifests itself because although the input of the generator is random, the generated images are always the same, i.e. the generator specializes in generating the same image of each class.
We tried to apply some of the existing techniques to avoid mode collapse[17], [18]. We try for example to adjust the learning rate of the GAN, since, as a general rule, lowering the learning rate can make the problem disappear.
Another method we tried was known as label smoothing, i.e., assigning a value of "0.9" to real labels instead of a "1". This usually makes the discriminator not too confident in its classification, which sometimes avoids mode collapse.
Another way to try to avoid the mode collapse is to implement the Wasserstein GAN, or WGAN[19], [20]. This is an extension of the GAN that seeks an alternative way to train the generator model. The modification over the GAN basically boils down to using a linear activation function in the output layer of the discriminator. Also, it uses the Wasserstein loss (RMSProp) to train the discriminator and generator instead of ADAM and SDG respectively and updating the discriminator more times than the generator.
But when applying these techniques, either the cDCGAN did not converge correctly (Fig. 6), or it converged but mode collapse occurred and generated the same image for each class.

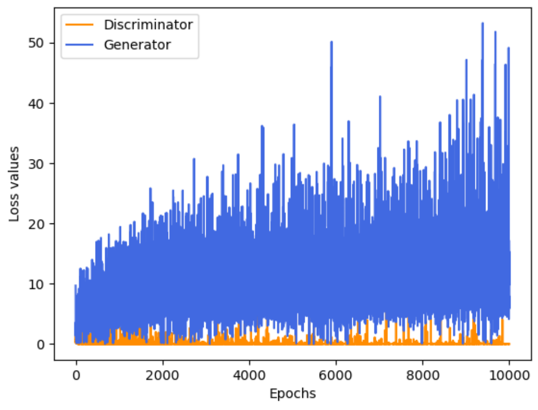

Figura.6 Training cDCGAN

We also tried to train a cGAN, i.e., a convolutional but not conditional GAN. The result was not good either. We still had mode collapse. We tried to train the cGAN without dimensionality reduction, i.e. with 4000x4000 images, to see if the problem was the loss of information in the dimensionality reduction. The problem was that GAN training with high dimensionalities is state of the art and they have convergence problems, so this didn't work either. The conclusion is that our images have little variability as they are transformed images of temporal signals very similar to each other.
In the end, we tried to train an independent linear GAN for each class with images of 128x128 dimensionality.
This finally made it possible to train all classes and generate a database of 2000 examples of each class different from each other.
For our data augmentation objective, we need to have sufficient variability to have distinct examples and we also need to be able to generate many examples in a robust way (without mode collapse). Also, it is desirable that does not require a large amount of computational cost. Thus, it has been decided to use a GAN formed by linear layers (Fig. 3) and apply a dimensionality reduction of the images that allows us to continue to successfully classify the different types of faults (128 x 128 x 3). Figure 7 shows the training of the linear GAN in which it can be seen that after the epoch 1000, the discriminator loss is around 0.5, while the generator is around 1.5. These values are typical of a correct GAN training.

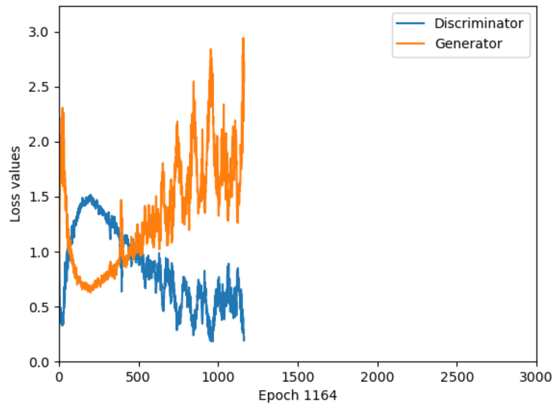

Figura.7 Training Linear GAN

The following figures show four examples of signals (images) from the original database (Fig. 8) and synthetic examples generated by the GAN (Fig. 9).

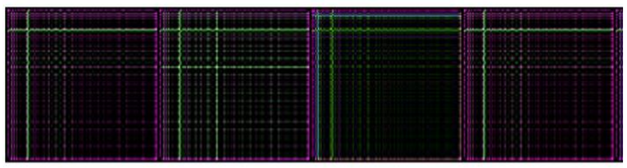

Figure 8. Original examples

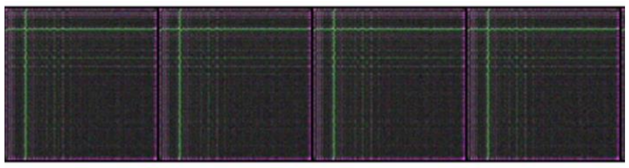

Figure 9. Synthesized examples generated by the GAN

Thus, with the generation of data augmentation by GAN, we are able to introduce versatility of examples, while maintaining the essence that characterizes each class (type of fault). In this case, it is the GAN itself during its training, which manages to extract the hidden information. We have also demonstrated that the dimensionality of the original signals can be reduced and that a very simple version of GAN (linear GAN) can be trained, avoiding the inherent problems of mode collapse in this type of neural networks.
In the literature, there are several metrics used to measure the quality with which GANs generate synthetic images. The parameters that are usually measured are:

- Creativity: non-duplication of the actual images.
- Consistency: the generated images must have the same style.
- Diversity: the generated images are different from each other.

The main parameter is called *Likeness Score* (*LS*), wich is (based on Euclidean distance) to analyse how two classes of data are mixed together.
In our case, the LS metric yields values very low, which is to be expected since, as we have said, the generated images have a very high variability respect to the original ones, which results in a low "quality" from the point of view of the LS metric.
In our experiment, we have used as metric the distance based on the *Contrastive Learning*. This metric is based on the percentage of success obtained in the classification of the different types of faults between the original images and those synthetically generated by the GAN.
We have a database of 200 original examples classified in 5 classes (0, 1, 2, 3, 4), with 40 examples of each class. In addition, a database of synthetic examples generated by our linear GAN is also available. These last are 10,000 examples also classified in 5 classes (0, 1, 2, 3, 4), with 2,000 examples for each class.
The tables below show the percentage of success in determining whether a class belongs to the class to which it is being compared. The first table (Table I) compares synthetic examples generated by GAN. Each class is compared with 384 different images from its own class and with 96 images from each of the other classes. The second table (Table II) compares original examples with synthetic examples generated by GAN. Each class is compared with 28 examples from its own class, and with 8 examples from each of the other classes.

CONCLUSIONS

Data augmentation is a widely used tool, with excellent results, in the field of machine learning. Being able to have a strongly extended training set of examples leads to more robust and better performing learning processes.

	0	1	2	3	4
0	99,73%	100%	100%	100%	100%
1	100%	99,73%	100%	100%	100%
2	100%	100%	99,73%	100%	100%
3	100%	100%	100%	100%	100%
4	100%	100%	100%	100%	100%

Table I. Comparison of synthetic examples vs synthetic examples

	0	1	2	3	4
0	96,43%	100%	100%	100%	100%
1	100%	100%	100%	100%	100%
2	100%	100%	100%	100%	100%
3	100%	100%	100%	100%	100%
4	100%	100%	100%	100%	100%

Table II. Comparison of original examples vs synthetic examples

Normally standard techniques such as rotations, scaling, etc. can be used, which unfortunately in some other cases is not possible. Therefore, GANs are becoming a very feasible option to generate these new examples, and to extend the number of training examples.
In conclusion, we can say that the proposed method manages to increase the signal database in an efficient way and solves the problem of this type of signals due to the difficulty to obtain them in field, or even simulating them by means of a model of the electrical grid.
In this article we start from a very small database of 200 examples, and finally get a set of 10,000 new examples with sufficient variability. Finally, all of this is backed up by a classification process with excellent results.

REFERENCES

[1] M. M. A. K. E. M. and H. H. M. M. Irzaei, “(PDF) Review of fault location methods for distribution power system.” https://www.researchgate.net/publication/234682955_Review_of_fault_location_methods_for_distribution_power_system (accessed Jan. 05, 2023).

[2] R. J. Hamidi and H. Livani, “Traveling-Wave-Based Fault-Location Algorithm for Hybrid Multiterminal Circuits,” *IEEE Transactions on Power Delivery*, vol. 32, no. 1, pp. 135–144, Feb. 2017, doi: 10.1109/TPWRD.2016.2589265.

[3] S. Wei, G. Yanfeng, and L. Yan, “Traveling-wave-based fault location algorithm for star-connected hybrid multi-terminal HVDC system,” *2017 IEEE Conference on Energy Internet and Energy System Integration, EI2 2017 - Proceedings*, vol. 2018-January, pp. 1–5, Jun. 2017, doi: 10.1109/EI2.2017.8245645.

[4] L. Xie *et al.*, “A novel fault location method for

hybrid lines based on traveling wave," *International Journal of Electrical Power & Energy Systems*, vol. 141, p. 108102, Oct. 2022, doi: 10.1016/J.IJEPES.2022.108102.

[5] Z. Wang and T. Oates, "Imaging Time-Series to Improve Classification and Imputation," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2015-January, pp. 3939–3945, Jun. 2015, doi: 10.48550/arxiv.1506.00327.

[6] J. G. Fornas, E. H. Jaraba, A. L. Estopinan, and J. Saldana, "Detection and Classification of Fault Types in Distribution Lines by Applying Contrastive Learning to GAN Encoded Time-series of Pulse Reflectometry Signals.," *IEEE Access*, 2022, doi: 10.1109/ACCESS.2022.3214994.

[7] D. M. de Silva and G. Poravi, "A Review on Generative Adversarial Networks," *2021 6th International Conference for Convergence in Technology, I2CT 2021*, Apr. 2021, doi: 10.1109/I2CT51068.2021.9417942.

[8] C. Ledig *et al.*, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 105–114, Sep. 2016, doi: 10.48550/arxiv.1609.04802.

[9] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2242–2251, Mar. 2017, doi: 10.48550/arxiv.1703.10593.

[10] W. Chen and J. Hays, "SketchyGAN: Towards Diverse and Realistic Sketch to Image Synthesis," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 9416–9425, Jan. 2018, doi: 10.48550/arxiv.1801.02753.

[11] F. H. Kiyoiti, S. Tanaka, C. Aranha, W. S. Lee, and T. Suzuki, "Data Augmentation Using GANs," *Proc Mach Learn Res*, vol. XXX, pp. 1–16, Apr. 2019, doi: 10.48550/arxiv.1904.09135.

[12] E. Mansfield, J. Wang, J. Russell, J. Li, and C. Adams, "Data Augmentation Generative Adversarial Networks," Nov. 2017, doi: 10.48550/arxiv.1711.04340.

[13] M. Li, J. Lin, Y. Ding, Z. Liu, J.-Y. Zhu, and S. Han, "GAN Compression: Efficient Architectures for Interactive Conditional GANs," *IEEE Trans Pattern Anal Mach Intell*, vol. 44, no. 12, pp. 9331–9346, Nov. 2022, doi: 10.1109/TPAMI.2021.3126742.

[14] F. Zhu, M. He, and Z. Zheng, "Data augmentation using improved cDCGAN for plant vigor rating," *Comput Electron Agric*, vol. 175, p. 105603, Aug. 2020, doi: 10.1016/J.COMPAG.2020.105603.

[15] T. Salimans *et al.*, "Improved Techniques for Training GANs," *Adv Neural Inf Process Syst*, vol. 29, 2016, Accessed: Jan. 06, 2023. [Online]. Available: https://github.com/openai/improved-gan.

[16] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton, "VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning," *Adv Neural Inf Process Syst*, vol. 2017-December, pp. 3309–3319, May 2017, doi: 10.48550/arxiv.1705.07761.

[17] 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," *Adv Neural Inf Process Syst*, vol. 2017-December, pp. 6627–6638, Jun. 2017, doi: 10.48550/arxiv.1706.08500.

[18] Y. Qin, N. Mitra, and P. Wonka, "How does Lipschitz Regularization Influence GAN Training?," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12361 LNCS, pp. 310–326, Nov. 2018, doi: 10.48550/arxiv.1811.09567.

[19] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral Normalization for Generative Adversarial Networks," *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, Feb. 2018, doi: 10.48550/arxiv.1802.05957.

[20] A. Karnewar and O. Wang, "MSG-GAN: Multi-Scale Gradients for Generative Adversarial Networks," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7796–7805, Mar. 2019, doi: 10.48550/arxiv.1903.06048.