ORIGINAL ARTICLE

Generating and Refining Particle Detector Simulations Using the Wasserstein Distance in Adversarial Networks

Martin Erdmann[1] · Lukas Geiger[1] · Jonas Glombitza[1] · David Schmidt[1]

Abstract

We use adversarial network architectures together with the Wasserstein distance to generate or refine simulated detector data. The data reflect two-dimensional projections of spatially distributed signal patterns with a broad spectrum of applications. As an example, we use an observatory to detect cosmic ray-induced air showers with a ground-based array of particle detectors. First we investigate a method of generating detector patterns with variable signal strengths while constraining the primary particle energy. We then present a technique to refine simulated time traces of detectors to match corresponding data distributions. With this method we demonstrate that training a deep network with refined data-like signal traces leads to a more precise energy reconstruction of data events compared to training with the originally simulated traces.

Introduction

Modern deep learning methods have been shown to be highly successful, e.g., in applications of handwriting, speech, and image recognition [1–6].

A new training concept is realized in so-called generative adversarial networks (GANs) which produce artificial images from random input while guided by real images [7]. They are based on two networks, an image generator and a discriminator separating artificial from real images, trained in opposition to one another. Similarly, adversarial training methods have been used to modify artificial images to better reproduce patterns found in natural images [8].

Two improvements of GAN methods which influence this work have recently been reported. So-called auxiliary classifier generative adversarial networks (AC-GANs) generate artificial images bound to given image class labels using label conditioning [9]. In addition to quantifying differences between real and artificial images, the Wasserstein distance has been introduced in generative adversarial networks (WGANs) to improve the stability of the learning process and to avoid mode-collapsing problems which are widely

known for other GAN setups [10, 11]. For a review of deep learning methods see [12].

In both particle and astroparticle physics research, deep learning concepts have already been applied successfully for data analyses, see e.g. [13–20]. Applications of the GAN concept have demonstrated generation of jet kinematics and calorimeter showers with unprecedented speed [21–24]. In addition, adversarial training methods have been shown to protect classifier networks from an error-prone variable [25].

In this paper we investigate adversarial training with the Wasserstein distance for a number of particle physics applications. First we present a method for generating two-dimensional signal patterns in spatially distributed sensors for a given physics label. This is a general task with broad applications in both particle and astroparticle detector simulations.

As an example we use cosmic ray-induced air showers in the Earth's atmosphere which produce signals in ground-based detector stations. This setup corresponds to a calorimeter experiment with a single readout layer. We will train a WGAN to generate signal patterns corresponding to a given primary particle energy.

In a further step, we tackle a pressing matter arising in training deep networks with simulated data that differ from measured distributions. We refine simulated signal traces to approximate real data (which, for simplicity, are simulated in this paper as well) using an adversarial training concept

✉ Martin Erdmann
erdmann@physik.rwth-aachen.de

[1] Physics Institute 3A, RWTH Aachen University,
52056 Aachen, Germany

guided by WGANs. We then compare the quality of reconstructing particle energies using a deep neural network after training with either the original or the refined simulated data.

Our paper is structured as follows: We introduce the Wasserstein distance and explain its application in adversarial training before presenting our network architectures for generating data or refining simulated data. After that, we specify the simulated data sets used for training and evaluating the networks. We then generate data-like signal patterns constrained by energy labels. We also refine simulated time traces and evaluate their impact on network training before presenting our conclusions.

Adversarial Network Architectures

In the adversarial training method, a generator network is required to learn probability distributions underlying observed event data distributions. A discriminator network is used to support this learning process by quantifying the differences between a set of event data distributions and the generated event distributions.

Using supervised machine learning, models are usually trained to predict labels associated with data, and further show a small generalization error. In contrast, this adversarial approach targets on training a generator to approximate a probability distribution. This can be considered as an unsupervised learning task.

The feedback of the discriminator network to the generator network about the quality of the generated events is encoded in the loss function. When using cross entropy type loss functions, training GANs has been observed to be delicate, hard to monitor and sometimes produce incoherent results. A frequently observed issue is a phenomenon known as mode collapsing, where the generator produces results in a restricted phase space only.

In the following sections, we first introduce the Wasserstein distance as an alternative loss function in generative adversarial training which leads to improved training stability. Additionally, mode collapsing has not been observed when using the Wasserstein distance [11]. We then expand the WGAN concept to generating events according to a given label, and to refining simulated event distributions.

Adversarial Training with the Wasserstein Distance

An alternative loss function for adversarial networks has been formulated based on the Wasserstein-1 distance, also referred to as Earth mover's distance [10]. As an intuitive interpretation, this distance gives the cost expectation for moving a probability distribution onto a target distribution along optimal transport paths.

The Wasserstein distance exhibits wanted properties concerning convergence of sequences of probability distributions [10]. This distance measure can thus serve well for quantifying the similarity between data x and generated events $\tilde{x}$. Here, x represents a set of event observables in data, while $\tilde{x}$ represents the corresponding observables for generated events. A common approach for generating $\tilde{x}$ is to implement a neural network g_θ with weights θ:

$$\tilde{x} = g_\theta(z) \tag{1}$$

Here, z is a random input which can be sampled from an arbitrary distribution.

As computing the formal definition of the Wasserstein distance (D_W) is intractable, an equivalent representation via the Kantorovich–Rubinstein duality is used [26]:

$$D_W = \sup_{f \in \text{Lip}_1} (\mathbb{E}[f(x)] - \mathbb{E}[f(\tilde{x})]) \tag{2}$$

Here, Lip_1 is the set of 1-Lipschitz functions f, and $\mathbb{E}$ represents expectation values for f operating on the data events x and the generated events $\tilde{x}$, respectively.

The core idea of the WGAN concept is to approximate the set of 1-Lipschitz functions f using a neural network $f \approx f_w$ parameterized by the weights w. The difference of the expectation values in Eq. (2)

$$C_1 = \mathbb{E}[f_w(x)] - \mathbb{E}[f_w(\tilde{x})] \tag{3}$$

constitutes the central term in the loss function of the network. The network is labeled 'critic' following [10, 11] as the converged value of C_1 gives a measure of the similarity of generated and data events.

Approximating the supremum in (2) is done by minimizing the loss of the critic network with the negative argument $-C_1$. In order to include the Lipschitz condition, the loss function has been extended by a gradient penalty [11]:

$$C_2 = \lambda \, \mathbb{E}\left[(||\nabla_{\hat{u}} f_w(\hat{u})||_2 - 1)^2\right]. \tag{4}$$

Here, the event admixture

$$\hat{u} = \varepsilon x + (1 - \varepsilon)\tilde{x} \tag{5}$$

of real data x and generated data $\tilde{x}$ is used to calculate the gradients of the critic network which are forced by the loss to remain close to one. The randomly and uniformly drawn value $0 \le \varepsilon \le 1$ samples the gradients along connecting lines between x and $\tilde{x}$. λ represents a hyperparameter of the training.

The generator uses the gradient of the distance measure C_1 (3) with respect to the parameters θ for training. In order to provide this measure we first update the critic by subjecting m data events and m generated events to the network represented by f_w.

In this initial training step, the weights w of the critic network are optimized to minimize the loss $-C_1 + C_2$ from Eqs. (3, 4). During this step, the parameters θ of the generator

are frozen. In the adjacent training step, the critic weights w are frozen temporarily, and the parameters θ of the generator network g_θ are adjusted. By reducing the Wasserstein distance measure, which is based on the output of the critic network, the generator $g_\theta(z)$ is trained to generate more realistic data samples. The critic is then trained again and the algorithm starts from the beginning.

This iterative procedure is repeated until overall convergence is achieved, leaving C_1 (3) as our measure of similarity between generated and data events. To provide an accurate gradient for the generator, the critic is usually trained for $n_{\text{critic}} > 1$ iterations before updating the generator once.

Physics Conditioning of the Generator

To enforce generated events to reflect certain properties of data events, the input to the generator can be extended by physics labels y_{label} in addition to the random numbers z in Eq. (1):

$$\tilde{x} = g_\theta(z, y_{\text{label}}) \qquad (6)$$

The required labels can, for example, be particle kinematics where the labels are obtained from corresponding energy or angular distributions. To push the generator network training towards a label condition, an additional term is introduced in the loss function. It compares the value of an input label with the result of an additional network $a_{\theta'}$ that reconstructs the corresponding observable from the generated data:

$$C_3 = \kappa \left[y_{\text{label}} - a_{\theta'}(\tilde{x}) \right]^2. \qquad (7)$$

We will call this network $a_{\theta'}$ 'constrainer' network parameterized by the weights θ'. The weight of the physics label in the total loss function $-C_1 + C_2 + C_3$ of the generator network is controlled by the hyperparameter κ. When using several physics labels in a conditioning process, the loss function can be extended accordingly.

The constrainer network $a_{\theta'}$ is trained supervised using data and their associated physics label y_{data}. Correspondingly, the loss function denotes:

$$C_4 = \left[y_{\text{data}} - a_{\theta'}(x) \right]^2. \qquad (8)$$

In the adversarial training explained above, the constrainer network $a_{\theta'}$ is trained supervised after each critic update. As the loss C_4 will influence only the constrainer (critic and constrainer are separated networks), both networks could also be trained simultaneously.

A similar term as Eq. (7) has been used in the so-called AC-GAN where images were generated using label conditioning [9]. Instead of the discrete classifier we use a continuous label here, along with the WGAN concept.

Generating Signal Patterns Using an Energy Label

Signals observed in particle detectors originate from physics-driven processes which lead to patterns dissimilar from random patterns. For example, low-energy events in a calorimeter typically exhibit signal patterns with small signals and a small spatial extent, while high-energy events cause signal patterns that can be widely distributed.

To enforce the generator to respect this dependency, we input an energy label E_{label} in addition to the random noise z to generate a signal pattern for the detectors of our cosmic ray observatory. Therefore, the generator is modified to $g = g_\theta(z, E_{\text{label}})$. The distribution of the input E_{label} follows the energy distribution of the air shower simulation. In this way, generated patterns are conditioned to follow the primary particle energy as reconstructed by the constrainer network. The resulting energy distribution of the generated events will cover a similar phase space as the simulated data.

Network architecture and training In order to generate signal patterns to a given energy label, our training architecture consists of a generator, a constrainer and a critic. The complete training architecture is shown in Fig. 1a. The generator and the critic networks are used for the adversarial training procedure explained above in "Adversarial Training with the Wasserstein Distance".

Our generator architecture is motivated by the class of DCGANs which is proposed in [27] and is based on Transposed Convolutions. Its specification can be found in the Appendix Table 1.

For the critic network we used an architecture inspired by [9] with LeakyReLU non-linearity and without batch normalization layers as we used the gradient penalty loss C_2 (4). For details of the critic model see Table 2.

Also shown in Fig. 1a is the constrainer network which is constructed similarly to the architecture presented in [20]. In the following we will refer to this architecture as AixNet. It is used to reconstruct the energy contained in the signal patterns. In our setup we used $l = 80$ noise variables which
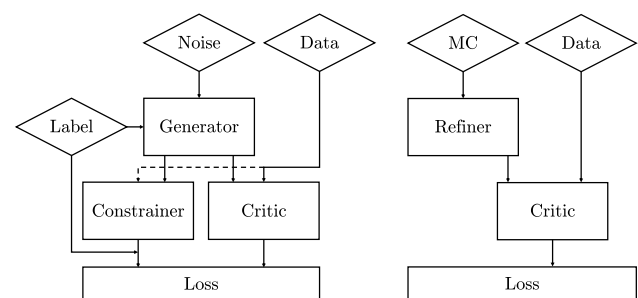

Fig. 1 Architectures of **a** the generator step of a conditioned generative adversarial network (solid line), and supervised training step of the constrainer network (dashed line), **b** the refining adversarial network

are sampled from a Gaussian distribution. The loss weight $\kappa = 0.01$ was used in Eq. (7), and the gradient penalty was scaled with $\lambda = 10$ (Eq. 4). Furthermore, we used a batch size of 128, and the training was run for 500 epochs with $n_{\text{critic}} = 10$ critic and constrainer updates before 1 generator update was applied. We used the Adam optimizer with learning rate $lr = 5 \times 10^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.9$ [28]. Furthermore, a decay of 10^{-4} for the generator and 10^{-5} for the critic was used. To enhance the training, the constrainer was pre-trained for 3 epochs using a learning rate of $lr = 10^{-3}$ and the default settings of Adam. As deep learning framework we use Keras [29] and TensorFlow [30]. For training we used a single NVIDIA GeForce GTX 1080 card provided by the VISPA project [31].

Refining Simulated Signal Traces to Match Data

Our second application of WGANs aims at refining simulated detector signal distributions to match data distributions. This is an attempt to solve a long-standing issue in machine learning, namely training of deep networks with simulations that differ from data distributions. Such refined simulations potentially increase the robustness of deep neural networks for data applications.

To refine signal traces we require the energy label of the simulation to follow a similar profile as the data distribution and make use of a generator network architecture which allows only for small modifications of the simulated traces.

Network architecture and training Figure 1b shows the network architecture for refining simulated signal distributions. Here, the generator network of adversarial training is called 'refiner' [8] to emphasize its aim to improve the quality of existing simulations rather than generating new samples. On input the refiner receives simulated signal distributions instead of random numbers. It returns modified distributions with the same dimensionality as the original input, thereby aiming to preserve the physics annotation of the original simulation. Thus, the refiner acts as a deterministic transformation from a single simulated event to a more 'data-like' event. The refiner network and the critic network are subjected to adversarial training as explained in "Adversarial Training with the Wasserstein Distance", where the refiner replaces the generator part.

In our example application of the cosmic ray observatory, for every event we simulate time traces for d detectors placed on a hexagonal grid, each of which has k time bins with amplitude A_k. In total $d \times k$ amplitude values are given to the refiner network. On output the refiner network again delivers $d \times k$ values as the modified time traces for the d detectors.

Correspondingly, the data pool contains time traces of data events. These traces are unlabeled, i.e., the data traces have no direct relation with the generated time traces.

The refiner employs a residual architecture [5] inspired by SimGAN [8] using four residual blocks, each consisting of two 3-dimensional convolutions with kernels operating on the time traces of the detector array. The architecture of the critic closely resembles the structure of AixNet [20] later used to reconstruct the energy. The detailed network architecture for the refiner is listed in Table 3.

The networks are trained for 10,000 refiner iterations with a batch size of 100 following the algorithm outlined in "Adversarial Training with the Wasserstein Distance" using the distance measure as presented in (3, 4). For each refiner step we update the critic $n_{\text{critic}} = 10$ times with a gradient penalty scaled by $\lambda = 5$ using the Adam optimizer [28] with learning rate $lr = 10^{-4}$, $\beta_1 = 0.5$ and $\beta_2 = 0.999$. We evaluate the final performance of the refining network by training AixNet [20] on the refined traces to reconstruct the primary particle energy (see "Network Training with Refined Signal Traces" below).

Simulated Data for Training and Evaluation

To simulate cosmic ray-induced air showers we use the parameterized simulation program presented in [20]. This simulation directly produces signal traces in water-Cherenkov detectors placed on a hexagonal grid with a spacing of 1500 m. They are located at a height of 1400 m above sea level, motivated by the Pierre Auger Observatory [32]. For simplicity we restrict our simulations to vertical showers with a fixed depth of the shower maximum. Alternatively, the setup can be understood as a granular calorimeter with a single readout layer.

For each simulated event, the air shower consists of two components, one component reflecting muons from pion decays, the other being particles of the electromagnetic cascade which arrive with a time delay compared to the muons. The simulation has been tuned from measurements to deliver $\sim 30\%$ of the energy in the muon component, and $\sim 70\%$ through the electromagnetic cascade.

Data We simulated 10^5 cosmic proton events with energies between $E = (1, \ldots, 100)$ EeV following a flat distribution (1 EeV $= 10^{18}$ eV). The muon and electromagnetic energies follow the above-mentioned 30/70 subdivision. Each event consists of $d = 9 \times 9$ detectors with signal traces containing $k = 80$ amplitude values in the time bins.

For each event, the time integrated signal strengths in the detectors can be visualized as a two-dimensional signal pattern. Examples of signal patterns as well as of signal time traces will be presented in "Energy Constrained Spatial Signal Pattern" and "Network Training with Refined Signal Traces" respectively. We will refer this simulated data set to as our 'data' events.

Simulation In order to produce a simulation which deviates from the data, we produce another set of 10^5 simulated cosmic proton events with the same conditions as for the above-mentioned data set, except for the division of the energy. For the energy fractions of the muonic and electromagnetic energies we use the inverted 70/30 subdivision instead. Furthermore, the amount of absolute noise in the time traces and event-by-event fluctuations are reduced by a factor of two in order to reflect underestimation of noise in detector simulations.

As the time of arrival and the transverse shower distributions are different between muons and particles of the electromagnetic cascade, the shapes of the time traces are different compared to the traces of the data set, as shown in "Network Training with Refined Signal Traces". We will refer to this set of simulated events as our 'simulated' events.

Energy Constrained Spatial Signal Pattern

To generate patterns of detector signals as the response to cosmic ray events we use the network setup presented in Fig. 1a. The events in the data pool originate from the data set described in "Simulated Data for Training and Evaluation", using only the $d = 9 \times 9$ values of the time integrated signal traces and the original energy of the primary particle.

In Fig. 2a we show example patterns of detector signals generated after 10 training epochs with test labels of $E_{label} = 10, 30, 70$ and 100 EeV. All patterns appear to be rather different from the typical patterns with large signals in the shower center and smaller signals around that. Furthermore, the sizes of the generated signal patterns are not in agreement with the energy labels.

In Fig. 2b we show example patterns after 490 epochs, again with test labels of $E_{label} = 10, 30, 70$ and 100 EeV. Already here the signal patterns improve and are inline with our expectations. The hottest station is in the center of the shower and the signal decreases for outlying stations. The pattern structure also shows a highly local correlation of neighbor stations which coincide with expectations. Furthermore, the increasing pattern size for higher energies is clearly visible. In addition, the total signal distribution correlates significantly with higher energies and meets with expectations.

To reconstruct the primary particle energy from the generated signal patterns we use the simultaneously trained constrainer (see Fig. 1a).

In Fig. 3 we show the correlation of the input energy to the generator with the particle energy as reconstructed by the constrainer network AixNet for 5000 generated events. The distinct correlation implies that the generator has not only been trained to produce detector signal patterns as an image-like product, but has in addition learned to produce patterns related to a given particle energy.

In Fig. 4a we show the critic loss of the adversarial training. The approximate Wasserstein distance C_1 (3) (red curve) converges slowly to zero for increasing epochs. Furthermore, the gradient penalty C_2 (4) (green curve) is reduced during the iterations. Hence the loss C_1 gives an estimation of the Wasserstein distance and therefore a similarity measure of the generated and the data events. The convergence to zero is in accordance with the generated events to reproduce expected properties (Fig. 2).

In Fig. 4b the supervised training loss C_4 (8) of the constrainer network AixNet reflects the improving reconstruction performance. We checked the validation loss
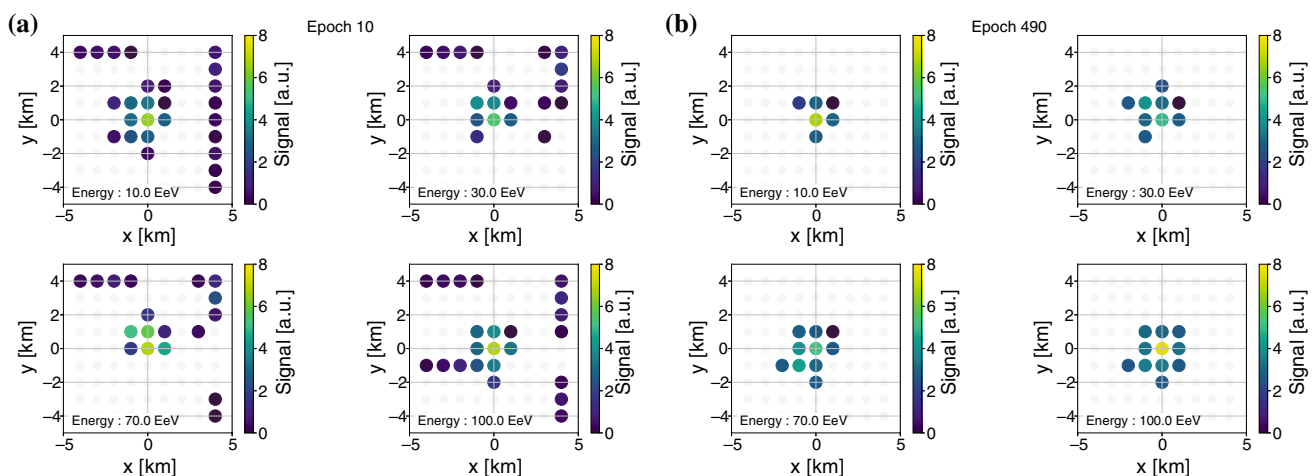

Fig. 2 Detector signal patterns of WGAN generated events including an energy condition. Color coded is the signal strength for four events with different cosmic ray energies after **a** 10 training epochs, **b** 490 training epochs
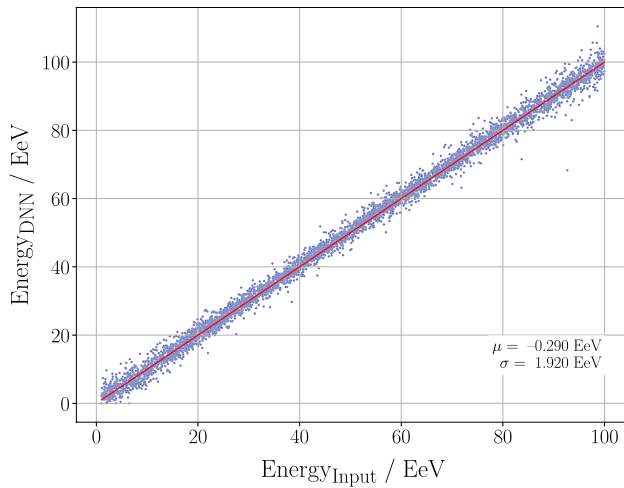
Fig. 3 Energy reconstructed by the constrainer network from analyzing the WGAN generated signal patterns compared to the energy input to the WGAN
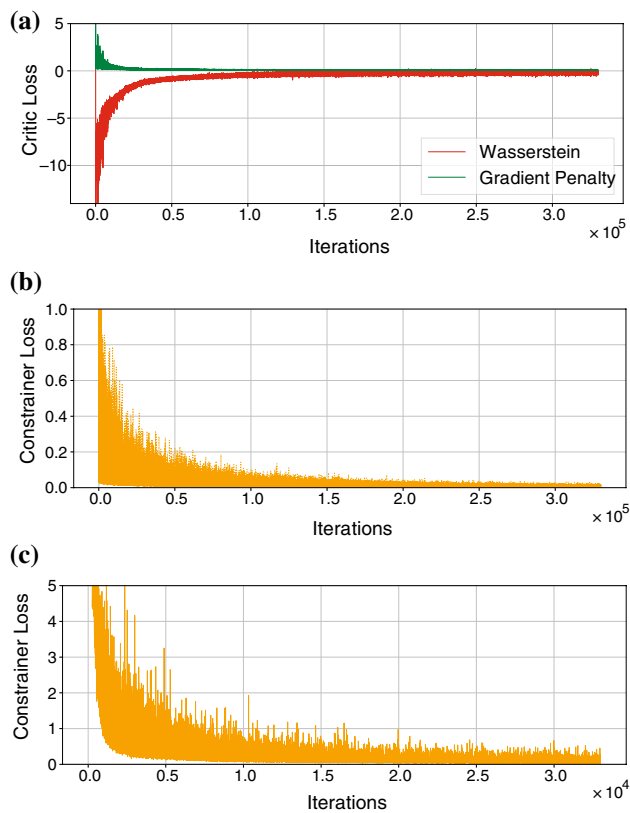

Fig. 4 Loss function **a** $-C_1$ (3) during the critic training showing the approximated Wasserstein distance (red) between generated and simulated signal patterns, and the gradient penalty C_2 (4) (green) scaled by λ, **b** C_4 (8) of constrainer scaled by the hyperparameter κ during the supervised training, and **c** C_3 (7) of the constrainer scaled by κ during the generator training visualizing the increasing conditioning performance. Note that the constrainer is only updated every $n_{\text{critic}} = 10$ iterations

as well (not shown in the figure) which shows no sign of overtraining.

Figure 4c shows the loss function C_3 (7) during the generator training. The constrainer loss decreases considerably with increasing iterations. This development is also visible in Fig. 2 where a correlation between signal pattern, signal size, and energy is apparent only for later epochs.

Network Training with Refined Signal Traces

To reconstruct the primary particle energy from detector signals we will make direct use of the amplitude distributions of the time traces. We again perform the energy reconstruction with AixNet [20].

Usually the training of a network is based on simulated data. However, when reconstructing particle energies from measured traces, differences between data and simulated traces may cause substantial uncertainties in the reconstructed energy. In order to reduce these uncertainties we will refine simulated traces to match unlabeled data-like traces using the adversarial network architecture presented in Fig. 1b. The refined traces will then be used to train AixNet.

In Fig. 5a we show the time trace of the detector with the largest signal in a data event with particle energy $E = 69$ EeV by the black solid curve. The black dotted curve represents a corresponding simulated time trace with matching primary energy. Due to the overestimated muon component in the simulated shower, the amplitude of the simulated time trace rises faster than in the corresponding data event (for a detailed definition of the data sets refer to "Simulated Data for Training and Evaluation"). As shown by the blue circular symbols in Fig. 5a, the refiner network modified the simulated trace to more closely resemble the data trace.

Note that this direct comparison between data traces and simulated traces results from a test data set simulated with identical parameters and identical random seeds. During the adversarial training of the networks, this matched information is not available as the data traces are simulated with different random seeds and are passed unlabeled to the critic network.

In Fig. 5b we show the signal traces for a neighbor detector. Also here, the originally simulated trace is adapted by the refiner network to match the data trace.

To evaluate the ability to preserve the properties of the simulation we investigate the impact of refined traces on the energy reconstruction. We trained AixNet to reconstruct the primary particle energy on the originally simulated traces, or alternatively the refined simulated traces.

In Fig. 6a, b we trained AixNet on the originally simulated traces. In Fig. 6a we benchmark AixNet by reconstructing the particle energy on a test set of simulated traces following the same distribution as the simulated training data.
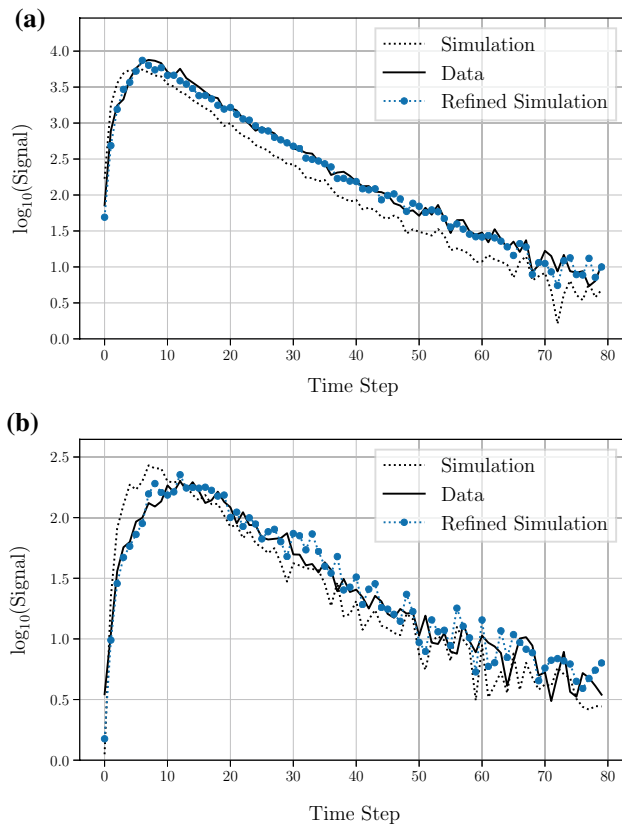
(a)

(b)

Fig. 5 Signal traces of an event with energy $E = 69$ EeV measured by **a** the detector with the largest signal, **b** a neighbor detector. The black solid curve shows the data trace, the black dotted curve the trace from simulation, and the blue circular symbols represent the refined simulated traces

This demonstrates a good energy reconstruction quality of the network.

In Fig. 6b we reconstruct particle energies of data events with the previous network trained on simulated traces. The network generalizes poorly on data due to the dissimilarities of the training set (simulated) and test set (data) which leads to a non-linear reconstruction bias and increased reconstruction uncertainties. This is a common problem when training neural networks on simulations that do not perfectly mirror real data.

In Fig. 6c we trained AixNet on the refined simulated traces instead of the original simulated traces and again evaluated the network performance on data traces. The network performs remarkably better compared to the training with the originally simulated data. The reconstruction quality is found to be worse compared to the benchmark shown in Fig. 6a. However, compared to the training with the original simulation (Fig. 6b), training with refined traces leads to a lower energy bias and improved energy resolution This shows that the refiner network is able to modify simulations to more accurately resemble the data distribution.

This quantitative test can also be conducted on measured data for a detector with hybrid techniques where measurements from two or more sub-detectors are available. Examples are cosmic ray experiments using particle detectors and fluorescence telescopes, or collider detectors with calorimeters and tracking detectors.

In Fig. 7 we show the convergence of the critic loss $-C_1 + C_2$ (3, 4) as a similarity measure of the refined and the data traces. With an increasing number of iterations, the refiner network is able to adapt simulations to better resemble data. The converged distance measure indicates remaining differences between data and simulation. However, the impact of these differences appear to be
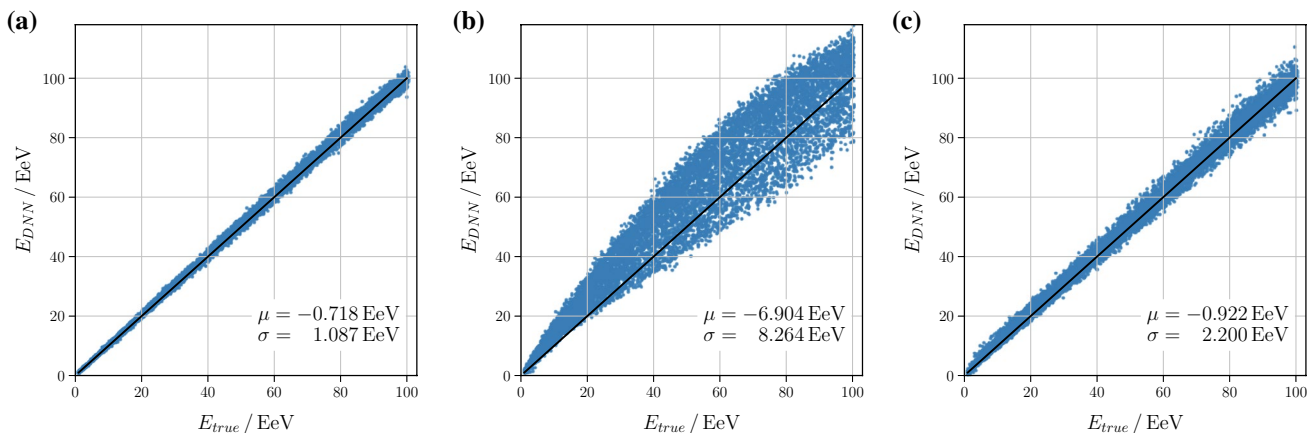

(a)   **(b)**   **(c)**

Fig. 6 Reconstruction of the primary particle energy using the signal traces of the detectors as input to AixNet. **a** Benchmark for training and evaluation, both based on simulated data. **b** Reconstructed data events using the network of **a** trained on simulated data. **c** Reconstructed data events with AixNet trained by simulated traces that were refined to match data events prior to the training (compare Fig. 5)
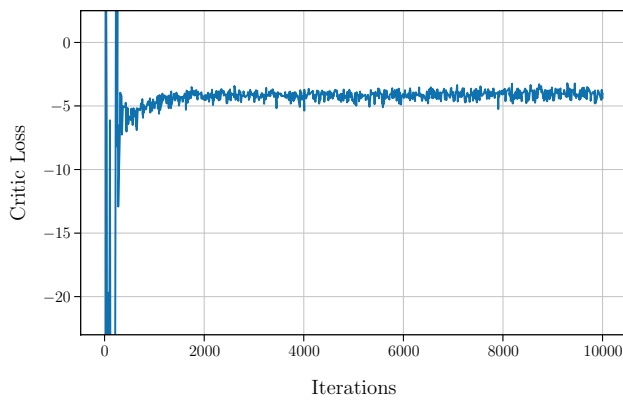
Fig. 7 Loss function of the critic network reflecting the similarity of the simulated traces and data traces

sufficiently small when evaluating the quality of the energy reconstruction.

Conclusion

In this paper we investigated two variants of adversarial network methods for detector simulations. In both cases, the transfer of probability distributions from one data set to another data set using adversarial network training is found to work well using the Wasserstein distance in the loss function. As a specific example we used air shower simulations with an array of ground-based water-Cherenkov detectors to represent single-layer calorimeter simulations.

We generated signal patterns of detector responses showing that the patterns can be constrained to follow properties expected from physics. In our example we constrained the energy contained in the shower and found that the generated events follow this given energy.

Refinement of simulated detector signal traces to match data traces appears to be a promising method for solving a long-standing issue in machine learning. Instead of training a deep network with simulations that differ in details from data, simulations can be adapted to match data prior to network training. For our example of the air shower simulations we showed that small refinements of the signal traces lead to improved reconstruction of the primary particle energy with respect to both energy bias and energy resolution.

Compliance with Ethical Standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Appendix

See Tables 1, 2 and 3.

Table 1 Generator network as used in the WGAN to generate signal patterns

Operation	Kernel	Feature maps	Padding	BN	Activation
Generator input 80 + 1					
Linear	N/A	80		×	ReLU
Transposed convolution	3×3	64	Valid	√	ReLU
Transposed convolution	3×3	128	Valid	√	ReLU
Transposed convolution	3×3	128	Valid	√	ReLU
Transposed convolution	3×3	256	Valid	√	ReLU
Convolution	3×3	1	Same	×	ReLU
Generator output $9 \times 9 \times 1$					

Table 2 Critic network as used in the WGAN to generate signal patterns

Operation	Kernel	Feature maps	Padding	BN	Activation
Critic input $9 \times 9 \times 1$					
Convolution	3×3	64	Same	×	LeakyReLU
Convolution	3×3	128	Same	×	LeakyReLU
Convolution	3×3	128	Same	×	LeakyReLU
Convolution	3×3	256	Same	×	LeakyReLU
GlobalMaxPooling				×	
Dropout					
Linear	N/A	100		×	LeakyReLU
Dropout					
Linear	N/A	1		×	
Critic output 1					

Table 3 Refiner network as used in the WGAN to refine signal traces. Residual shortcut connections are added after every pair of convolutions

Operation	Kernel	Feature maps	Padding	Activation
Input $9 \times 9 \times 80 \times 1$				
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 7$	64	Same	ReLU
Convolution	$1 \times 1 \times 1$	1	Same	ReLU
Output $9 \times 9 \times 80 \times 1$				

References

1. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554
2. Ciresan DC, Meier U, Schmidhuber J (2012) Multi-column deep neural networks for image classification. arXiv:1202.2745
3. Yu D, Deng L (2014) Automatic speech recognition: a deep learning approach. Springer, London
4. Russakovsky O et al (2015) Imagenet large scale visual recognition challenge. Int J Comput Vis 115(3):211–252
5. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. arXiv:1512.03385
6. Silver D et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529:7578
7. Goodfellow I et al (2014) Generative adversarial networks. arXiv:1406.2661 [stat.ML]
8. Shrivastava A et al (2016) Learning from simulated and unsupervised images through adversarial training. arXiv:1612.07828 [cs.CV]
9. Odena A, Olah C, Shlens J (2016) Conditional image synthesis with auxiliary classifier GANs. arXiv:1610.09585 [stat.ML]
10. Arjovsky M, Chintala S, Bottou L (2017) Wasserstein GAN. arXiv:1701.07875 [stat.ML]
11. Gulrajani I et al (2017) Improved training of Wasserstein GANs. arXiv:1704.00028 [cs.LG]
12. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT Press, Cambridge
13. Aurisano A et al (2016) A convolutional neural network neutrino event classifier. JINST 11(09):P09001
14. Baldi P et al (2014) Searching for exotic particles in high-energy physics with deep learning. Nat Commun 5:4308
15. Baldi P et al (2015) Enhanced Higgs to $\tau^+\tau^-$ searches with deep learning. Phys Rev Lett 114:111801
16. Adam-Bourdarios C et al (2015) The Higgs boson machine learning challenge. In: Cowan G et al (eds) Proceedings of the NIPS 2014 workshop on high-energy physics and machine learning, Proceedings of machine learning research, vol 42. PMLR, Montreal, pp 19–55
17. Guest D et al (2016) Jet flavor classification in high-energy physics with deep neural networks. Phys Rev D 94(11):112002
18. Baldi P et al (2016) Jet substructure classification in high-energy physics with deep neural networks. Phys Rev D 93(9):094034
19. Erdmann M, Fischer B, Rieger M (2017) Jet-parton assignment in $t\bar{t}$H events using deep learning. JINST 12(08):P08020
20. Erdmann M, Glombitza J, Walz D (2018) A deep learning-based reconstruction of cosmic ray-induced air showers. Astropart Phys 97:46–53
21. de Oliveira L, Paganini M, Nachman B (2017) Learning particle physics by example: location-aware generative adversarial networks for physics synthesis. Comput Softw Big Sci 1(1):4
22. Paganini M, de Oliveira L, Nachman B (2018) Accelerating science with generative adversarial networks: an application to 3D particle showers in multilayer calorimeters. Phys Rev Lett 120(4):042003
23. Paganini M, de Oliveira L, Nachman B (2018) CaloGAN. Phys Rev D 97:014021
24. Carminati F et al (2017) Calorimetry with deep learning: particle classification, energy regression, and simulation for high-energy physics. In: Workshop on deep learning for physical sciences (DLPS 2017), NIPS 2017, Long Beach
25. Shimmin C et al (2017) Decorrelated jet substructure tagging using adversarial neural networks. Phys Rev D 96(7):074034
26. Villani C (2008) Optimal transport: old and new. Grundlehren der mathematischen Wissenschaften. Springer, Berlin
27. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv:1511.06434
28. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
29. Chollet F et al (2015) Keras. https://github.com/keras-team/keras
30. Abadi M et al (2015) TensorFlow: large-scale machine learning on heterogeneous systems. https://www.tensorflow.org
31. Erdmann M et al (2017) The VISPA internet-platform in deep learning applications. In: Proc. 18th int. workshop on advanced computing and analysis techniques in physics research (ACAT), Washington
32. Aab A et al (2015) The Pierre Auger cosmic ray observatory. Nucl Instrum Methods A 798:172–213