Mathematical Modelling and Numerical Simulation with Applications, 2023, 3(3), 234–255

https://dergipark.org.tr/en/pub/mmnsa
ISSN Online: 2791-8564 / Open Access
https://doi.org/10.53391/mmnsa.1327485

RESEARCH PAPER

Understanding the mathematical background of Generative Adversarial Networks (GANs)

Bilgi Yilmaz [1,2,*,‡] and **Ralf Korn** [1,3,‡]

[1]Department of Mathematics, Kaiserslautern Technical University, Kaiserslautern, Germany,
[2]Department of Mathematics, Faculty of Science, Kahramanmaraş Sütçü Imam University, Türkiye,
[3]Department of Financial Mathematics, Fraunhofer ITWM, Germany

*Corresponding Author
‡ yilmaz@mathematik.uni-kl.de (Bilgi Yilmaz); korn@mathematik.uni-kl.de (Ralf Korn)

Abstract

Generative Adversarial Networks (GANs) have gained widespread attention since their introduction, leading to numerous extensions and applications of the original GAN idea. A thorough understanding of GANs' mathematical foundations is necessary to use and build upon these techniques. However, most studies on GANs are presented from a computer science or engineering perspective, which can be challenging for beginners to understand fully. Therefore, this paper aims to provide an overview of the mathematical background of GANs, including detailed proofs of optimal solutions for vanilla GANs and boundaries for f-GANs that minimize a variational approximation of the f-divergence between two distributions. These contributions will enhance the understanding of GANs for those with a mathematical background and pave the way for future research.

Keywords: Generative adversarial networks; unsupervised learning; qualitative analysis

AMS 2020 Classification: 91G15; 91G20; 91B02; 62E99

1 Introduction

Generative Adversarial Networks (GANs) introduced by [1] consist of generative and discriminative neural network models that are usually denoted by letters G and D, respectively. To visualize GANs environment better in our mind, the generative model may be regarded as a counterfeiter who is attempting to produce a fraud Van Gogh's Starry Night painting and sell it without being noticed, whereas the discriminative model is equivalent to an expert who specializes in Van Gogh, trying to detect the counterfeit fraud painting. However, the counterfeiter does not care about producing images that are a variation of the original Starry Night painting. In the applications of GANs, the aim is not to present a new image identical to the original painting. Instead, it

➤ **Received:** 14.07.2023 ➤ **Revised:** 06.08.2023 ➤ **Accepted:** 08.08.2023 ➤ **Published:** 30.09.2023

aims to create a unique illustration of Starry Night that the Van Gogh expert recognizes as an unknown Van Gogh painting that is unprecedented anywhere before. As a result, a computation starts between the generator and discriminator over the fraud painting detection. The competition continues until the counterfeiter becomes intelligent enough to deceive the expert successfully. More precisely, the discriminator's role is to distinguish the real and fraud paintings, while the generator's role is to generate fraud paintings in such a way that it can mislead the discriminator, and the discriminator is unable to cope with rejecting the fraud paintings any longer (see Figure 1).

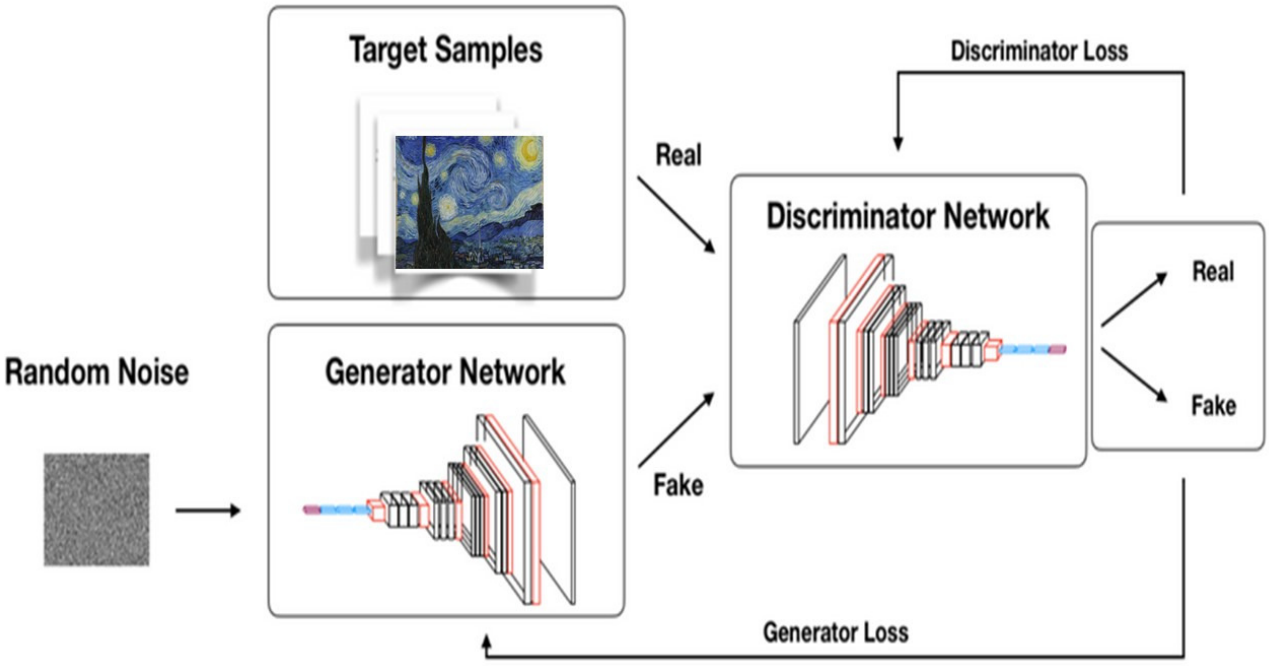

Figure 1. A visualization of the discriminator and generator networks as a counterfeiter and Van Gogh's painting expert

Figure 1 presents a visualization of the training process of Generative Adversarial Networks (GANs). The GAN training process involves two main components: the Generator and the Discriminator. The Generator takes random noise from the latent space as input and generates fake data, attempting to mimic the real data distribution. Initially, the generated data is random and typically of low quality. A batch of real data is sampled from the training dataset that serves as the ground truth for the Discriminator during training. The Discriminator is trained on both real and fake data. It is presented with the real data and the corresponding labels (1 for real) to learn to distinguish real data from fake data. It is then presented with the fake data generated by the Generator and the corresponding labels (0 for fake) to learn to identify the fake data. The Discriminator's performance is evaluated using a loss function, such as binary cross-entropy, which measures how well the Discriminator is differentiating the real data from the fake data. The Generator is trained to deceive the Discriminator by generating fake data that appears as realistic as possible. The Generator takes random noise as input and aims to generate data that the Discriminator labels as real. The Generator's performance is evaluated using the Discriminator's response to the fake data it generates. The Generator's loss function encourages it to generate data that deceives the Discriminator (i.e., the Discriminator's prediction closer to

1). The model parameters of both the Generator and Discriminator are updated using gradient descent or some variant, optimizing considering loss functions. The process continues iteratively, with the Generator getting better at generating realistic data, and the Discriminator becoming more skilled at differentiating real from fake data. The ideal state is reached when the Generator can create data that is indistinguishable from real data, and the Discriminator cannot confidently classify between the real and generated data. It is important to note that the Generator never uses the real data as input and trains solely with random noises.

To put this in a positive framework, we can say that the discriminator serves as a kind of quality control of the generated data. The better the discriminator performs, the better the benchmark for the generator. Then, the generator can finally beat the benchmark in a form in which the optimal strategy of the discriminator is essentially only guessing whether the generated data are fraud or real. Finally, the generator is ready to be used in synthetic data generation.

Some of the key issues that are critical in the applications of GANs are as follows:

- Quantifying "similar objects" is trickier than it sounds, and it carries the core to GANs. In the field of mathematics research, we have many alternative methods to quantify the similarity between any two objects, which can also lead us to different objectives in setting up GANs.
- In the application of GANs, we aim to generate original objects, which can be distant in which distance measure we consider to any objects at hand as a training dataset χ (i.e. we do not want to copy χ, but we feel the generated objects and χ belong to the same class).
- We do not care about generating a perturbation of the original painting. Instead, we want to produce a fake painting that the expert is going to consider like a unique painting that belongs to Van Gogh, which she has seen for the first time in her life.
- In this setting, the appropriate concept of similarity is distributional similarity. We call two objects similar if both are samplings from the same (or roughly same) probability distribution. This means that the two objects share similar characteristics and features that are determined by the underlying probability distribution. Therefore, we maintain a training dataset denoted with $\chi \subset \mathbb{R}^n$ that consists of samples gathered from μ. In this context, μ is a probability distribution, and its density is represented by $p(x)$. We want to arrive at a reasonable approximating probability distribution ν having a density $q(x)$ to μ. Then, we can obtain artificial or synthetic objects that are identical to objects in the training (real) dataset χ by sampling from ν.
- You may question, why we do not just consider the distributions as $\nu = \mu$ and obtain samples from the real data distribution μ.
  Unfortunately, such sampling is exactly the main problem of GANs since μ *is not known explicitly*. The only thing that we know is that we have a finite set of samples χ sampled from μ. Consequently, the actual issue is identifying the properties of μ by only using χ. In this sense, we must focus on specifying an appropriate probability distribution ν as an approximation process to μ.
- In addition to considering a distribution similar to μ in the sense of probability distances, one can also try to characterize μ by the empirical behavior of the data, their so-called *stylized facts*.
- Generally, the success of GANs depends on the sophistication of μ and the training dataset χ size.

The basic approach of GANs

The purpose of the study is to clarify the mathematical background of GANs. Therefore, it focuses on only theoretical aspects of GANs and contains any applications.

To approximate a given probability distribution μ, GANs require an initially defined probability distribution to start its training. Generally, the initial distribution, which we define as γ, is

introduced in space $\mathbb{R}^d$. Here, the space dimension d is not necessarily identical to the space dimension n (of $\mathbb{R}^n$). Now, suppose we have chosen the initial distribution γ to be the standard normal distribution, and we have denoted it with $N(0, \mathcal{I}_d)$. However, we are free to choose γ from other well-known probability distribution families (e.g., uniform). GANs utilize a technique to discover a mapping G, defined as $G : \mathbb{R}^d \longmapsto \mathbb{R}^n$. At this stage, consider a random variable $z \in \mathbb{R}^d$ sampled from initial distribution γ. Then, we can claim that the mapping $G(z)$ is from the same distribution family as μ. To emphasize, the probability distribution of $G(z)$ can be defined in the form of $\gamma \circ G^{-1}$. Here, G^{-1} denotes the inverse of G, and the inverse maps subsets of space $\mathbb{R}^n$ to subsets in space $\mathbb{R}^d$. Therefore, in the GANs modeling method, we desire to find a mapping $G(z)$ that satisfies $\gamma \circ G^{-1} = \mu$ or at least $\gamma \circ G^{-1}$ is a reasonable approximation of the real data distribution μ.

The vanilla GAN approach forms an adversarial system from which the generator receives updates on a continuous basis to increase output accuracy. More rigorously, the vanilla GAN presents a neural network called a discriminator, which attempts to label the observed samples as real, and generated samples as fake. From this perspective, the discriminator behaves like a classifier that attempts to distinguish real samples from fake samples. To this end, the discriminator assigns a probability $D(x) \in [0, 1]$ to each sample x for its probability of being a real sample. If samples $G(z_j)$ are outputs of the generator, the discriminator attempts to restrict them since they are fake samples.

In the early stage of training a GAN, restricting generated samples as fake should not be challenging since the generator is not elegant at generating realistic samples. However, after each attempt G fails to produce realistic samples to trick D, and G learns and adjusts itself with a refinement update. Thus, the improved G performs more reasonably compared to the one used at the early stages, and then it is the discriminator D's progression to revise for refinement. In an ideal case, through such an adversarial iterative process, we can eventually arrive at an equilibrium point; therefore, even the most reasonable D cannot perform more satisfactory labeling than a random guess. At this point, the samples generated by G become extremely identical to training samples χ in distribution. Consequently, the discriminator decision becomes completely random, and the probability of being real approximates 50%.

In GANs modelling approach, we have to define both the discriminator and generator by utilizing neural networks to understand the distributional properties of given data. Each neural network has its corresponding parameters ω and θ. These parameters are used in the training of the discriminator and generator and include the weights (also known as synaptic weights) of the neural network layers, as well as the biases of these layers. They are learned during training to optimize the performance of the GAN in generating realistic samples. Hence, we should register $D_\omega(x)$ for the discriminator and $G_\theta(z)$ for the generator, and we should denote $\nu_\theta := \gamma \circ G_\theta^{-1}$. Thus, it is clear that our task is to identify the desired generator $G_\theta(z)$ by adequately adjusting its parameter θ.

Building a GAN framework

As we mentioned above, there are two parties in GANs modeling method: generator $G_\theta(z)$ and a discriminator $D_\omega(x)$ who are in competition, and both parties have their own roles during the modeling process. More specifically,

The generator:

- The generator operates with a random vector whose length is fixed and, then, produces a fake sample in the corresponding domain.
- The vector is sampled from the Gaussian distribution (generally) and utilized to seed the

generator. After the training, points in the multidimensional vector space conform with points in the real data domain, forming a compact replica of the training data distribution.
- The vector space is called the latent space or equally vector space. It consists of some latent variables or some hidden variables, which are critical for the domain but cannot be observed directly.

The discriminator:

- The discriminator uses a sample from the domain as input (it may be either real or fake) and assigns a real or fake (generated) binary class label.
- The real sample directly comes from the original data, while fake samples are only outputs of the generator.
- The discriminator is a classifier model. When the training is finished, the discriminatory model is junked as we are curious about in the generator. Occasionally, the generator can be reset as it has learned to effectively determine characteristic from examples sampled from the problem domain. Some or all of the characteristics extraction layers can be utilized in transfer learning applications by utilizing the same or similar input data.

Both players in the min-max game are expressed by a corresponding function. Each function is differentiable concerning its inputs and parameters. As it introduced above, the discriminator is a differentiable function denoted by D that uses x as input and is allowed to use only the discriminator network weights ω as parameters. On the other hand, the generator is specified by G and uses the random vector z as the initial input and is only allowed to use the weights of the generator network θ as parameters [2].
In this setting, both players have their own loss functions. The loss functions are described with regard to parameters specific to players. The discriminator desires to minimize the problem $L^{(D)}(\omega, \theta)$ and it must accomplish the minimization by controlling only its parameters ω. On the other hand, the generator desires to minimize $L^{(G)}(\omega, \theta)$ and must accomplish the minimization by controlling only its parameters θ. Here, the discriminator and generator losses rely on the other player's parameters. However, both players are limited to controlling only their own parameters. Since each player's loss relies on the opposite player's parameters, despite each player being allowed to regulate its parameters and cannot control the opposite player's parameters, such a scenario is generally expressed as a game rather than a classical optimization problem [2].
As we mentioned already, generator G is a differentiable function. After we produce its random vector z from a well-known initial distribution called γ, G generates a fake sample x, which is implicitly sampled from the model distribution ($\mathbb{P}_{model} = \nu$). Commonly, a deep neural network is utilized to characterize the generator. However, we have some constraints on the configuration of the corresponding neural network. If we want $\mathbb{P}_{model}$ to have complete support on $\mathcal{X}$, the dimension of the generator should be at least as large as the dimension of $\mathcal{X}$ [2].
In a similar fashion, discriminator D is also a differentiable function, whose objective is to categorize samples accurately as real and fake. The discriminator is also naturally characterized by a deep neural network. Again, it has some restrictions on the configuration of its corresponding network. It has to use only real and fake samples as entries and assigns a probability score $D(x) \in [0, 1]$ for each x [2]. Here, notice that the generator never sees the real data and only uses random vector z as input, while the discriminator uses both real, and the generator's output.

A simple derivation of the loss functions

Before starting the definition of the loss functions, note that in the classical GANs architectures, the design of the discriminator loss functions $L^{(D)}$ always remains the same. They differ only by the cost function for the generator, $L^{(G)}$ [2]. The loss function introduced in the original study [1]

is obtained from the binary cross-entropy formula as follows

$$L(\hat{y}, y) = \Big[y \cdot \log(\hat{y}) + (1-y) \cdot \log(1-\hat{y}) \Big]. \tag{1}$$

Here, y and $\hat{y}$ correspond to the original and fake data, respectively.

In the training of the discriminator, the label of data assigned by the real data $\mu(x)$ is $y = 1$ (real/observed data) and $\hat{y} = D(x)$. Then, by substituting this into Eq. (1), we have

$$L(D(x), 1) = \log(D(x)), \tag{2}$$

and for the data sampled from the generator, the label is $y = 0$ (fake data) and $\hat{y} = D(G(z))$. Similarly, by substituting these into Eq. (1), we end up with

$$L(D(G(z)), 0) = \log(1 - D(G(z))).$$

In this setting, the goal of the discriminator is to accurately classify its input as fake or real. Therefore, the given loss functions for G and D have to be maximized. Then, the final loss function of D is denoted as

$$L^{(D)} = \max \Big[\log(D(x)) + \log(1 - D(G(z))) \Big]. \tag{3}$$

At this stage, it is important to remember that the generator is competing against the discriminator. Hence, the generator aims to minimize the optimization problem given in Eq. (3), and consequently, its loss function evolves to

$$L^{(G)} = \min \Big[\log(D(x)) + \log(1 - D(G(z)) \Big]. \tag{4}$$

Now, let us combine the loss functions (3) and (4). By combining these two equations, we obtain a min-max problem as

$$L = \min_G \max_D \Big[\log(D(x)) + \log(1 - D(G(z))) \Big]. \tag{5}$$

Here, it is worth emphasizing that the loss function in Eq. (5) is valid only for a single data point. Therefore, to consider the entire dataset, we need to consider the expectation of the combined loss function as

$$\min_G \max_D V(D, G) = \min_G \max_D \Big[\mathbb{E}_{x \sim \mu}[\log(D(x))] + \mathbb{E}_{z \sim \gamma}[\log(1 - D(G(z)))] \Big]. \tag{6}$$

The min-max formulation introduced in Eq. (6) is a concise one-liner function that intuitively captures the adversarial nature of the competition between the players G and D. However, in practice, individual loss functions are defined for both players since the gradient of $y = \log(x)$ is steeper around $x = 0$ than $y = \log(1 - x)$. This means that trying to maximize $\log(D(G(z)))$, or equivalently minimizing $-\log(D(G(z)))$ leads to quicker and more significant improvements in the generator performance than attempting to minimize $\log(1 - D(G(z)))$.

2 Mathematical description of vanilla GANs

The adversarial game introduced in the previous section can be expressed mathematically by a min-max task for a target function defined by the discriminator $D(x) : \mathbb{R} :\longmapsto [0,1]$ and generator $G : \mathbb{R}^d \longmapsto \mathbb{R}^n$. Here, it is clear that G transforms the random vector $z \in \mathbb{R}^d$ sampled from γ into generated (fake) samples $G(z)$. Then, D attempts to distinguish the generated samples from the training samples that are supposed to be sampled from μ while G attempts to generate new samples that are identical in distribution to the data that we use in the training of GANs [3].

In the original study [1], a target loss function is introduced as

$$V(D,G) := \mathbb{E}_{x\sim\mu}[\log(D(x))] + \mathbb{E}_{z\sim\gamma}[\log(1-D(G(z)))],$$

where $\mathbb{E}$ represents the expectation concerning the distribution appointed in the subscript. We can avoid the subscript if there is no confusion.

The vanilla GAN solves the min-max problem given in Eq. (6). Heuristically, for a given G, the optimization problem $\max_D V(D,G)$ reveals the optimal D to reject outputs $G(z)$ by assigning higher probabilities to samples from μ and low probabilities to outputs $G(z)$. In contrast, for a given D, $\min_G V(D,G)$, the optimization problem reveals the optimal G, and therefore, the outputs $G(z)$ attempt to deceive D by assigning high probabilities for $G(z)$ [3].

Then, let us define $y = G(z) \in \mathbb{R}^n$ having a distribution defined as $\nu := \gamma \circ G^{-1}$, and the random vector $z \in \mathbb{R}^d$ is from the γ distribution family. Thus, we may rearrange $V(D,G)$ in terms of D and ν as follows

$$\begin{aligned} \tilde{V}(D,\nu): &= \mathbb{E}_{x\sim\mu}[\log(D(x))] + \mathbb{E}_{z\sim\gamma}[\log(1-D(G(z)))] \\ &= \mathbb{E}_{x\sim\mu}[\log(D(x))] + \mathbb{E}_{y\sim\nu}[\log(1-D(G(y)))] \\ &= \int_{\mathbb{R}^n} \log(D(x))d\mu(x) + \int_{\mathbb{R}^n} \log(1-D(y))d\nu(y). \end{aligned} \tag{7}$$

Then, the min-max problem defined in Eq. (6) evolves to

$$\min_G \max_D V(D,G) = \min_G \max_D \left(\int_{\mathbb{R}^n} \log(D(x))d\mu(x) + \int_{\mathbb{R}^n} \log(1-D(y))d\nu(y) \right). \tag{8}$$

Now, suppose that the distributions μ and ν have densities given as $p(x)$ and $q(x)$, respectively. Note that this can only happen under the condition of $d \geq n$. This condition is necessary for GANs to ensure that the discriminator is sufficiently powerful to distinguish real samples from generated ones. When $d \geq n$, the discriminator possesses a greater number of parameters compared to the sample size in the training dataset. Consequently, this asymmetry facilitates the discriminator's ability to effectively differentiate between real and generated samples. If $d < n$, the discriminator may not effectively learn to distinguish real from generated samples, resulting in poor-quality generated samples.

By using the densities, we obtain

$$V(D,\nu) = \int_{\mathbb{R}^n} \Big(\log(D(x))p(x) + \log(1-D(x))q(x) \Big) dx.$$

With the help of the current evolution, the min-max problem given in Eq. (6) evolves to

$$\min_G \max_D V(D,G) = \min_G \max_D \int_{\mathbb{R}^n} \Big(\log(D(x))p(x) + \log(1-D(x))q(x) \Big) dx.$$

From the evolved problem, notice that the equation is equal to $\min_\nu \max_D \tilde{V}(D,\nu)$ under the condition $\nu = \gamma \circ G^{-1}$ for some generator G.

Proposition 1 ([1]). *For distributions μ and ν on $\mathbb{R}^n$ having densities $p(x)$ and $q(x)$, respectively*

$$\max_D V(D,\nu) = \max_D \int_{\mathbb{R}^n} \Big(\log(D(x))p(x) + \log(1-D(x))q(x) \Big) dx$$

is achieved by $D_{p,q}(x) = \frac{p(x)}{p(x)+q(x)}$ for $x \in supp(\mu) \cup supp(\nu)$.

Proof Let us define the integrand as

$$f(D(x)) = \log(D(x))p(x) + \log(1-D(x))q(x).$$

To find the optimal solution, we look at the first order condition $\frac{df(D(x))}{dD(x)} = 0$ and second order condition $\frac{df^2(D(x))}{dD(x)^2} = 0$. Hence, let us start with

$$\frac{df(D(x))}{dD(x)} = \frac{p(x)}{D(x)} - \frac{q(x)}{1-D(x)} = 0.$$

By solving this equality for $D(x)$ we find the critical point

$$D_{p,q}(x) = \frac{p(x)}{p(x)+q(x)}.$$

Now, let us compute the second derivative

$$\frac{d^2 f(D(x)}{dD(x)^2} = \frac{-p(x)}{D(x)^2} - \frac{q(x)}{(1-D(x))^2}.$$

Then, it is obvious that the second derivative is strictly negative for at least one of $p(x)$ or $q(x)$ being positive. Therefore, we find the optimal solution $D_{p,q}(x)$ as

$$D_{p,q}(x) = \frac{p(x)}{p(x)+q(x)}.$$

■

As a result of Proposition 1, we can give the following remark immediately.

Remark 1. *The discriminator optimal solution of the min-max problem satisfies $D_{p,q}(x) = \frac{p(x)}{p(x)+q(x)} \in [0,1]$, and this is the requirement for the optimal discriminator.*

Note that the optimal solution makes the following sense intuitively:

- If some sample x is favorably actual, we may anticipate $p(x)$ to be close to one and $q(x)$ to converge at zero. Hence, the optimal D assigns one to such samples.
- For a generated sample $x = G(z)$, we anticipate the optimal D to assign zero since $p(G(z))$ has to be close to zero. When we train G to its optimal value, density $q(x)$ gets very close to density $p(x)$, i.e. we obtain $D_{p,q}(G(z)) \approx 0.5$.

As a consequence of Proposition 1, we can introduce the following theorem immediately.

Theorem 1. *Suppose $p(x)$ is a probability density function defined on space $\mathbb{R}^n$. Additionally, consider a probability distribution ν having a density function denoted as $q(x)$ and a discriminator function $D : \mathbb{R}^n \longmapsto [0,1]$ as usual. Then, we have a min-max problem as follows [3],*

$$\min_{\nu}\max_{D}\tilde{V}(D,\nu) = \min_{\nu}\max_{D}\int_{\mathbb{R}^n}\Big(\log(D(x))p(x) + \log(1-D(x))q(x)\Big)dx, \tag{9}$$

and, we reach a solution with a special choice of $q(x) = p(x)$ and $D(x) = \frac{1}{2}, \forall x \in supp(p)$.

Proof Let us now assume $p(x) = q(x)$ for all $x \in supp(p)$. Then, we have $\bar{D}(x) = 1/2$ and $\int_{\mathbb{R}^n}\log(1/2)p(x)dx = \int_{\mathbb{R}^n}\log(1/2)q(x)dx = -\log(2)$ as both p and q are probability densities. For this special choice of p, q, and D, we obtain

$$\tilde{V}(D,\nu) = -\log(4).$$

Note further that by the definition of the Jensen-Shannon divergence, we have

$$\begin{aligned} 0 \leq JS(p\|q) &= 0.5(KL(p\|0.5(p+q)) + KL(q\|0.5(p+q)) \\ &= 2\log(2) + \int_{\mathbb{R}^n}\left(p(x)\log\left(\frac{p(x)}{p(x)+q(x)}\right) + q(x)\log\left(\frac{q(x)}{p(x)+q(x)}\right)\right)dx \\ &= \tilde{V}(D,\nu) + log(4). \end{aligned}$$

Therefore, $\tilde{V}(D,\nu)$ cannot be smaller than $-\log(4)$. Thus, we have proved that $q(x) = p(x)$ – and thus $\bar{D}(x) = 1/2$ – yields the minimum possible value of $\tilde{V}(D,v)$ for any ν for the given choice of $D(x) = p(x)/(p(x)+q(x))$. Consequently, we end up with the desired result. ■

Theorem 1 reveals that the solution to the min-max problem given by Eq. (9) is the result we seek under the hypothesis of the distributions having the same densities. Theorem 1 holds for all distributions in general.

Theorem 2. *Suppose that μ again is a probability distribution function given on space $\mathbb{R}^n$ as in Theorem 1. Then, for a probability distribution ν and a discriminator $D : \mathbb{R}^n \longmapsto [0,1]$, we can introduce a min-max problem as follows [3]*

$$\min_{\nu}\max_{D}\tilde{V}(D,\nu) = \min_{\nu}\max_{D}\int_{\mathbb{R}^n}\Big(\log(D(x))d\mu(x) + \log(1-D(x))d\nu(x)\Big), \tag{10}$$

whose solution is achieved with the special choice $\nu = \mu$ and $D(x) = \frac{1}{2}$ μ−a.e.

Proof We first show that with the special choice of $\nu = \mu$ and $D(x) = \frac{1}{2}$ μ-almost everywhere, the min-max problem in Equation (10) is solved.

First, let's consider the objective function $\tilde{V}(D,\nu)$:

$$\tilde{V}(D,\nu) = \int_{\mathbb{R}^n} \left(\log(D(x))d\mu(x) + \log(1-D(x))d\nu(x)\right).$$

Substituting $\nu = \mu$ and $D(x) = \frac{1}{2}$, we have:

$$\begin{aligned}
\tilde{V}(D,\nu) &= \int_{\mathbb{R}^n} \left(\log\left(\frac{1}{2}\right)\mu(x) + \log\left(1-\frac{1}{2}\right)\mu(x)\right) \\
&= \int_{\mathbb{R}^n} \left(-\log(2)\mu(x) - \log\left(\frac{1}{2}\right)\mu(x)\right) \\
&= -\log(2)\int_{\mathbb{R}^n} \mu(x)dx + \log\left(\frac{1}{2}\right)\int_{\mathbb{R}^n} \mu(x)dx.
\end{aligned}$$

Since μ is a probability distribution function, the integral $\int_{\mathbb{R}^n} d\mu(x)$ is equal to 1. Therefore, the objective function simplifies to:

$$\tilde{V}(D,\nu) = -\log(2) + \log\left(\frac{1}{2}\right) = -2\log(2).$$

Hence, with the choice of $\nu = \mu$ and $D(x) = \frac{1}{2}$ μ-almost everywhere, the objective function $\tilde{V}(D,\nu)$ is minimized. To complete the proof, we need to show that for any other choice of ν and D, the objective function $\tilde{V}(D,\nu)$ is not smaller than 0.Let us consider an arbitrary choice of ν' and D' (where $\nu' \neq \mu$ or $D' \neq \frac{1}{2}$ μ-almost everywhere). Without loss of generality, assume that there exists a set $A \subset \mathbb{R}^n$ with positive measure such that $D'(x) \neq \frac{1}{2}$ for all $x \in A$. Since μ is a probability distribution function, we have $\mu(A) > 0$. Therefore, we can rewrite the objective function as:

$$\begin{aligned}
\tilde{V}(D',\nu') &= \int_{\mathbb{R}^n} \left(\log(D'(x))d\mu(x) + \log(1-D'(x))d\nu'(x)\right) \\
&\geq \int_A \left(\log\left(D'(x)\right)d\mu(x) + \log\left(1-D'(x)\right)d\nu'(x)\right).
\end{aligned}$$

Now, consider the term $\log(D'(x))d\mu(x)$ for $x \in A$. Since $D'(x) \neq \frac{1}{2}$ for all $x \in A$, we have $\log(D'(x)) < 0$ for all $x \in A$. Therefore, $\log(D'(x))d\mu(x) < 0$ for $x \in A$. On the other hand, consider the term $\log(1-D'(x))d\nu'(x)$ for $x \in A$. Since $D'(x) \neq \frac{1}{2}$ for all $x \in A$, we have $1-D'(x) \neq \frac{1}{2}$ for all $x \in A$. Therefore, $\log(1-D'(x)) < 0$ for all $x \in A$. Since ν' is a probability distribution, $d\nu'(x) \geq 0$ for all x. Hence, $\log(1-D'(x))d\nu'(x) \leq 0$ for $x \in A$. Combining these results, we have $\log(D'(x))d\mu(x) + \log(1-D'(x))d\nu'(x) < 0$ for $x \in A$. Therefore, $\tilde{V}(D',\nu') < 0$. Since ν' and D' were chosen arbitrarily, we can conclude that for any other choice of ν and D, the objective function $\tilde{V}(D,\nu)$ is not smaller than 0. Hence, the solution to the min-max problem in Equation (10) is achieved with the special choice $\nu = \mu$ and $D(x) = \frac{1}{2}$ μ-almost everywhere. This completes the proof. ■

Like many min-max problems, we may utilize the alternative optimization algorithm to find an optimal solution to the problem introduced by Eq. (9) that alternates by updating the discriminator and density q. Here, the updating process contains first updating the discriminator for density q, and second, updating density q with recently updated D. Notice that updating density q

means updating the generator. This process is repeated until we find an equilibrium point for the optimization.

Proposition 2. *If in each step of the training process, D is qualified to achieve an optimum point given $q(x)$, which is pursued by an update of approximating density $q(x)$ to further develop the criterion of minimization given as*

$$\min_q \int_{\mathbb{R}^n} \Big(\log(D(x))p(x) + \log(1 - D(x))q(x) \Big) dx.$$

At this stage, the approximating density q converges to the target density p.

Proof First, we show that if the discriminator D is qualified to achieve an optimum point given $q(x)$ in each step of the training process, then the approximating density q converges to the target density p. Let us consider the objective function to be minimized:

$$\min_q \int_{\mathbb{R}^n} \big(\log(D(x))p(x) + \log(1 - D(x))q(x) \big)\, dx.$$

In each step of the training process, the discriminator D is qualified to achieve an optimum point given $q(x)$. This means that for a fixed $q(x)$, the discriminator D is updated to maximize the objective function with respect to D. Let's denote this updated discriminator as D_q^*. Now, let us consider the objective function with the updated discriminator D_q^*:

$$\min_{q^*} \int_{\mathbb{R}^n} \Big(\log(D_q^*(x))p(x) + \log(1 - D_q^*(x))q(x) \Big)\, dx.$$

Since the discriminator D_q^* is optimized for a fixed $q(x)$, the objective function becomes:

$$\begin{aligned} \min_q \quad & \int_{\mathbb{R}^n} \Big(\log(D_q^*(x))p(x) + \log(1 - D_q^*(x))q(x) \Big)\, dx \\ = \quad & \min_q \left(\int_{\mathbb{R}^n} \Big(\log(D_q^*(x))p(x) \Big)\, dx + \min \left(\int_{\mathbb{R}^n} \Big(\log(1 - D_q(^*x))q(x) \Big)\, dx \right) \right). \end{aligned}$$

The first term $\min_q \int_{\mathbb{R}^n} \Big(\log(D_q(x))p(x) \Big)\, dx$ does not depend on $q(x)$ and can be treated as a constant. Therefore, minimizing this term is equivalent to maximizing $\int_{\mathbb{R}^n} \Big(\log(D_q(x))p(x) \Big)\, dx$. Similarly, the second term $\min_q \int_{\mathbb{R}^n} \Big(\log(1 - D_q(x))q(x) \Big)\, dx$ does not depend on $p(x)$ and can be treated as a constant.
Therefore, minimizing this term is equivalent to maximizing $\int_{\mathbb{R}^n} \Big(\log(1 - D_q((x))q(x) \Big)\, dx$. Since the objective function is the sum of these two terms, minimizing the objective function is equivalent to maximizing both $\int_{\mathbb{R}^n} \Big(\log(D_q(x))p(x) \Big)\, dx$ and $\int_{\mathbb{R}^n} \Big(\log(1 - D_q(x))q(x) \Big)\, dx$.
Now, let us consider the first term $\int_{\mathbb{R}^n} \Big(\log(D_q(x))p(x) \Big)\, dx$. Since $D_q(x)$ is optimized for a fixed $q(x)$, it can be considered as a constant with respect to $p(x)$. Therefore, maximizing this term is equivalent to maximizing $\int_{\mathbb{R}^n} p(x)dx$.
Similarly, let us consider the second term $\int_{\mathbb{R}^n} \Big(\log(1 - D_q(x))q(x) \Big)\, dx$. Since $D_q(x)$ is optimized for a fixed $q(x)$, it can be considered as a constant with respect to $q(x)$. Therefore, maximizing

this term is equivalent to maximizing $\int_{\mathbb{R}^n} q(x)dx$. Since the objective function is the sum of these two terms, maximizing the objective function is equivalent to maximizing both $\int_{\mathbb{R}^n} p(x)dx$ and $\int_{\mathbb{R}^n} q(x)dx$.

Now, let us consider the convergence of the approximating density q to the target density p. As we maximize the objective function, we aim to maximize both $\int_{\mathbb{R}^n} p(x)dx$ and $\int_{\mathbb{R}^n} q(x)dx$. To achieve this, the approximating density q needs to converge to the target density p. Therefore, if in each step of the training process, the discriminator D is qualified to achieve an optimum point given $q(x)$, then the approximating density q converges to the target density p.

This completes the proof. ■

In each step of the process, first, we find the optimal discriminator $D^*(x)$ for the current density $q(x)$. Later, update density $q(x)$ given the currently updated discriminator $D(x)$ to improve the accuracy. Repeating such a process finally leads us to the desired solution. In practice, nevertheless, we infrequently focus on optimizing discriminator D for a provided generator G. Instead, we generally focus on updating D a little while ago swapping to update generator G.

It is worth emphasizing here that the unconstrained min-max problems given by Eqs. (9) and (10) are not the same as the original min-max problem introduced in Eq. (6) or the equivalent to Eq. (7), where the probability distribution ν is constrained to $\nu = \gamma \circ G^{-1}$. However, it is useful in applications to suppose Eqs. (6) and (7) exhibit identical properties introduced in Theorem 2 and Proposition 2. We can suppose the same, even after further restricting the discriminator and generator functions are neural networks defined as $D = D_\omega$ and $G = G_\theta$ as instead. Then, set $\nu_\theta = \gamma \circ G_\theta^{-1}$. Under this setting, the min-max problem becomes $\min_\theta \max_\omega V(D_\omega, G_\theta)$, where

$$\begin{aligned} V(D_\omega, G_\theta) &= \mathbb{E}_{x\sim\mu}[\log(D_\omega(x))] + \mathbb{E}_{z\sim\gamma}[\log(1 - D_\omega(G_\theta(z))] \\ &= \int_{\mathbb{R}^n} \Big(\log(D_\omega(x))d\mu(x) + \log(1 - D_\omega(x))d\nu_\theta(x) \Big). \end{aligned} \tag{11}$$

Eq. (11) is the key to executing the fundamental optimization problem. Here, since we do not know the explicit form of μ (target distribution), we should approximate the expectations through sample averages. Thus, Eq. (11) helps us to find an approximation to $V(D_\omega, G_\theta)$. More precisely, suppose a set $\mathcal{A}$ that is a subset of samples drawn from the training/original dataset χ (a minibatch) defined above and suppose a set $\mathcal{B}$ that is a minibatch of samples in space $\mathbb{R}^d$ sampled from γ. Under these assumptions, we can approximate as [3]

$$\begin{aligned} \mathbb{E}_{x\sim\mu}[\log(D_\omega(x))] &\approx \frac{1}{|\mathcal{A}|} \sum_{x\in\mathcal{A}} \log(D_\omega(x)), \\ \mathbb{E}_{z\sim\gamma}[\log(1 - D_\omega(G_\theta(z)))] &\approx \frac{1}{|\mathcal{B}|} \sum_{z\in\mathcal{B}} \log(1 - D_\omega(G_\theta(z))). \end{aligned}$$

Note that a minibatch in the GANs framework refers to a small subset of training examples fed to the network in each training iteration. The minibatch size is typically chosen to balance the computational efficiency of training and the quality of GANs. Smaller minibatches can lead to faster training; however, they may result in a noisier gradient estimate and slower convergence. On the other hand, larger minibatches can provide a more accurate gradient estimate; however, they may require more memory and computational resources to process. During training, generator and discriminator networks are trained simultaneously by optimizing their respective loss functions using backpropagation. The minibatches of real data samples and generated samples are used to compute the discriminator's loss, while the generator's loss is computed using the generated

samples only. By using minibatches in GANs, the networks can efficiently learn the complex distribution of the data and generate high-quality synthetic samples.

3 f-divergence and f-GAN concepts

Recall our motivating problem defined for GAN having a probability distribution μ, known simply for the training samples at hand. We want to find a distribution ν through an iterative process. By beginning with a probability distribution ν and iteratively updating ν, we approximate the target distribution μ with ν. To approximate μ, first, we need to measure the distance between distributions μ and ν. The vanilla GAN uses the discriminator to approximate target distribution μ. However, we can use other measures to identify the distance between distributions.

f-divergence

We can measure the dissimilarity between any two distributions, in our case target distribution μ and approximated distribution ν, with the Kullback-Leibler (KL) divergence. Let $p(x)$ and $q(x)$ be the corresponding probability density functions of μ and ν defined on $\mathbb{R}^n$. Then, the distance between densities p and q is defined in the following form

$$\mathbb{D}_{KL}(p\|q) := \int_{\mathbb{R}^n} \log\left(\frac{p(x)}{q(x)}\right) p(x)dx.$$

Here, notice that $\mathbb{D}_{KL}(p\|q)$ is finite only if $q(x) \neq 0$ on $supp(p)$ almost everywhere. At this stage, we can conclude the following results for KL-divergence [4]:

- If $p(x) > q(x)$, x is a point in the real data with a high probability. This case is the heart of the '*mode dropping*' phenomenon. It occurs when we have large regions having high values of p, whereas having small values in q. Here, it is important to remark that if $p(x) > 0$ and $q(x) \to 0$, the integrand of $\mathbb{D}_{KL}$ rises to infinity very quickly. This means that such a cost function sets an exceptionally elevated cost to the generator's distribution that does not cover some data parts.
- If $p(x) < q(x)$, x has a low chance of being a data point, instead of a high chance of being a generated point. It is faced when we observe the generator producing an unrealistic image.If we observe $p(x) \to 0$ and $q(x) > 0$ we find that the value inside the $\mathbb{D}_{KL}$ shifts to 0. This means that such a cost function pays an exceptionally low cost for generating fake samples.

Remark 2. *Regarding GANs, $\mathbb{D}_{KL}(p\|q)$ has a unique minimum at $p(x) = q(x)$. Furthermore, it does not require knowing the unknown density $p(x)$ to estimate. However, it is impressive to notice that $\mathbb{D}_{KL}(p\|q)$ is not symmetrical for $p(x)$ and $q(x)$ [3, 4].*

Even though KL-divergence is widely used in the applications of GAN, there are other measures to identify the dissimilarity between distributions. For instance, the Jensen-Shannon (JS) divergence is given as

$$\mathbb{D}_{JS}(p\|q) := \frac{1}{2}\mathbb{D}_{KL}(p\|M) + \frac{1}{2}\mathbb{D}_{KL}(q\|M),$$

where $M = \frac{p(x)+q(x)}{2}$ is a divergence measure derived from KL-divergence.
The most significant benefit of JS-divergence is that it is well-defined for any densities $p(x)$, and $q(x)$ and symmetric concerning the densities ($\mathbb{D}_{JS}(p\|q) = \mathbb{D}_{JS}(q\|p)$) while KL-divergence is not symmetric.
Following Proposition 1, the minimization part of the min-max problem in the context of the vanilla GAN is exactly the minimization over density q of $\mathbb{D}_{JS}(p\|q)$ for a given p. As things stand,

$\mathbb{D}_{KL}$ and $\mathbb{D}_{JS}$ divergences are both particular cases of the $f-divergence$ where a more general form is introduced in [5] for such divergence measures.

Consider a strictly convex function $f(x)$ with a domain $I \subseteq \mathbb{R}$ that satisfies $f(1) = 0$. Additionaly, for computation purposes, we interiorize $f(x) = +\infty$, $\forall x \notin I$ convention. Then, we can introduce the f-divergence concept as introduced in [3].

Definition 1. *Consider two probability density functions $p(x)$ and $q(x)$ defined on space $\mathbb{R}^n$. Then, the $f-divergence$ between these two densities is*

$$\mathbb{D}_f(p\|q) = \mathbb{E}_{x\sim q}\left[f\left(\frac{p(x)}{q(x)}\right)\right] = \int_{\mathbb{R}^n} f\left(\frac{p(x)}{q(x)}\right)q(x)dx,$$

where we adopt $f(\frac{p(x)}{q(x)})q(x) = 0$ if $q(x) = 0$.

Remark 3. *Since the $f-divergence$ is not symmetric ($\mathbb{D}_f(p\|q) \neq \mathbb{D}_f(q\|p)$) in general, we can confuse which density divides and which density in the fraction. If we obey the original setting introduced in [5], then the definition of $\mathbb{D}_f(p\|q)$ will be our $\mathbb{D}_f(q\|p)$. In this study, we adopt the definition introduced in [7], where the f-GAN concept is first introduced.*

Proposition 3. *Suppose $f(\cdot)$ is a strictly convex function defined on $I \subseteq \mathbb{R}$ and $f(1) = 0$. Further, suppose either $supp(p) \subseteq supp(q)$ (equivalent to $p \ll q$) or $f(x) > 0$ for $x \in [0,1]$. Then, for $\mathbb{D}_f(p\|q) \geq 0$ and $\mathbb{D}_f(p\|q) = 0$, the necessary and sufficient condition is $p(x) = q(x)$.*

Proof Using the convexity property of function f and Jensen's inequality, we have

$$\mathbb{D}_f(p\|q) = \mathbb{E}_{x\sim q}\left[f\left(\frac{p(x)}{q(x)}\right)\right] \geq f\left(\mathbb{E}_{x\sim q}\left[\frac{p(x)}{q(x)}\right]\right) = f\left(\int_{supp(q)} p(x)dx\right) = f(r),$$

where the equality holds if and only if the ratio $q(x)/p(x)$ is a constant or function f is linear on the range of the ratio $p(x)/q(x)$. The range of $p(x)/q(x)$ depends on the probability distributions $p(x)$ and $q(x)$ being considered. In general, the ratio $p(x)/q(x)$ can take any positive value, zero, or infinity, depending on the values of $p(x)$ and $q(x)$ for a given x. However, in the context of importance sampling, it is common to consider the ratio $p(x)/q(x)$ as a weighting function for sampling from the target distribution $p(x)$. In this case, the $p(x)/q(x)$ range is typically restricted to a finite interval to ensure that the importance weights are bounded and can be effectively used for sampling.

Function f is a strictly convex function, so it may only be previous or for that matter, we should have $p(x) = rq(x)$ on $supp(q)$ for the equality to hold. Suppose we have the $r \leq 1$ condition. If we have $supp(p) \subseteq supp(q)$, then we obtain $r = 1$, and hence, we have $\mathbb{D}_f(p\|q) \geq 0$. Such an equality holds if and only if we have $p = q$. Suppose $f(t) > 0$, $\forall t \in [0,1)$, then we also have $\mathbb{D}_f(p\|q) \geq f(r) \geq 0$. For $r < 1$, we have $\mathbb{D}_f(p\|q) \geq f(r) \geq 0$. Therefore, if $\mathbb{D}_f(p\|q) = 0$, the conditions $r = 1$ and $p = q$ hold. ■

At this stage, we should note that $f-divergence$ can be specified for arbitrary probability distributions μ and ν on probability space Ω. Let τ be a third probability distribution that satisfies $\mu, \nu \ll \tau$, more specifically both μ and ν are absolutely continuous concerning the third probability distribution τ. For instance, suppose $\tau = \frac{1}{2}(\mu + \nu)$. Let $p = \frac{d\mu}{d\tau}$ and $q = \frac{d\nu}{d\tau}$ be Radon-Nikodym derivatives of p and q, respectively. We characterize the $f-divergence$ of probability distributions

μ and ν as [3]

$$\mathbb{D}_f(\mu\|\nu) := \int_\Omega f\left(\frac{p(x)}{q(x)}\right)q(x)d\tau = \mathbb{E}_{x\sim\nu}\left[f\left(\frac{p(x)}{q(x)}\right)\right]. \tag{12}$$

Here, once more we adopt the convention $f\left(\frac{p(x)}{q(x)}\right)q(x) = 0$ if $q(x) = 0$. Here, it is clear that this definition is free from the choice of the probability measure τ.
In the application of the $f-divergence$, the greatest difficulty is the unknown explicit expression of the target distribution denoted by μ. Hence, in the vanilla GAN setting, to calculate the $f-divergence$ ($\mathbb{D}_f(p\|q)$), we should express the divergence in terms of the average of samples. In [6], this problem is solved with the help of the convex conjugate of the convex function at hand.

Definition 2. *Suppose $f(\cdot)$ is a convex function on the interval defined as $I \subseteq \mathbb{R}$. The convex conjugate of f is simply a generalization of the celebrated Legendre transform. The convex conjugate $f^* : \mathbb{R} \longmapsto \mathbb{R}\cup\{\pm\infty\}$ is given as [3]*

$$f^*(y) = \sup_{t\in I}\Big\{ty - f(t)\Big\}.$$

We can introduce the following remark as an immediate result of the definition.

Remark 4. *The convex conjugate of convex functions is also called the Fenchel transform or Fenchel-Legendre transform.*

As we mentioned above, we may extend the convex conjugate f^* to $\mathbb{R}$ by defining $f(x) = +\infty$ for all $x \notin I$. Therefore, a more precise indication of f^* is illustrated in the following lemma.

Lemma 1. *Let $f(x)$ be a strictly convex and continuously differentiable function on $I \subseteq \mathbb{R}$, where $I^\circ = (a, b)$ with $a, b \in [-\infty, +\infty]$. Then [3],*

$$f^*(y) = \begin{cases} yf'^{-1}(y), & y \in f'(I^\circ) \\ \lim_{t\mapsto b^-}(ty - f(t)), & y \geq \lim_{t\mapsto b^-} f'(t) \\ \lim_{t\mapsto a^+}(ty - f(t)), & y \leq \lim_{t\mapsto a^+} f'(t). \end{cases}$$

Proof Define $g(t) = ty - f(t)$. Then, $g'(t) = y - f'(t)$ on $I \subseteq \mathbb{R}$, which is strictly decreasing since $f(t)$ is convex. Here, $g(t)$ is a function that is strictly concave on the domain defined with I. Note that, if $y = f'(t^*)$ for some $t^* \in I^\circ$, t^* is called a critical point of function g. Therefore, t^* has to be a global maximum of g. Therefore, $g(t)$ reaches its maximum at point $t = t^* = f'^{-1}(y)$. Now, suppose y is not in the range of f', in that case, $g'(t) > 0$ or $g'(t) < 0$ on I°. Suppose the case $g'(t) > 0\ \forall t \in I^\circ$. Here, it is clear that the supremum of function $g(t)$ is attained while $t \mapsto b^-$ because $g(t)$ is a monotonously increasing function. In a similar fashion, the second case $g'(t) < 0$, $\forall t \in I^\circ$ may be derived. ■
Based on Lemma 1, we can give the following remark:

Remark 5. *Note that $+\infty$ is a potential f^* value. Hence, the domain of f^* ($Dom(f^*)$) is characterized as sets where f^* is finite.*

A result of Lemma 1, under the assumption that f is a continuously differentiable function, $\sup_{t\in I}\{ty - f(t)\}$ is achieved for some $t \in I$ if and only if, y is in the range of $f'(t)$. Such a result is

clear when $y \in f'(I^\circ)$, however, it is arguable relatively effortlessly for finite boundary points in domain I. More commonly, without the differentiability assumption, $\sup_{t\in I}\{ty - f(t)\}$ is achieved if and only if $y \in \partial f(t)$ for some $t \in I$ ($\partial f(t)$ is set of subderivatives). We summarize some of the important properties of the convex conjugate in the following proposition [3]:

Proposition 4. *Let $f(x)$ be a convex function defined on $\mathbb{R}$ having a range $\mathbb{R} \cup \{\pm\infty\}$. Then, its convex conjugate f^* is a convex and lower-semi continuous function. Moreover, if f is a lower-semi continuous function, f satisfies Fenchel duality $f = (f^*)^*$.*

Calculation of f-divergence using the convex dual

To calculate the $f-$*divergence* from samples, [6] proposes using the convex dual of function f. Let μ and ν be probability two measures that satisfy $\mu, \nu \ll \tau$ for some probability measure τ, with $p = d\mu/d\tau$ and $q = d\nu/d\tau$. In the best scenario of $\mu \ll \nu$, by $f(x) = (f^*)^*(x)$, we retain

$$\begin{aligned}
\mathbb{D}_f(\mu\|\nu) : &= \int_\Omega f\left(\frac{p(x)}{q(x)}\right) q(x) d\tau(x) \\
&= \int_\Omega \sup_t \{t\frac{p(x)}{q(x)} - f^*(t)\} q(x) d\tau(x) \\
&= \int_\Omega \sup_t \{tp(x) - f^*(t)q(x)\} d\tau(x) \\
&\geq \int_\Omega \Big(T(x)p(x) - f^*(T(x))q(x)\Big) d\tau(x) \\
&= \mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))],
\end{aligned} \tag{13}$$

where $T(\cdot)$ denotes any Borel function. Therefore, by considering T overall Borel functions, one obtains

$$\mathbb{D}_f(\mu\|\nu) \geq \sup_T \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))]\Big). \tag{14}$$

In addition, $\forall x$, $\sup_t\{t\frac{p(x)}{q(x)} - f^*(t)\}$ is achieved for some $t = T^*(x)$ if $\frac{p(x)}{q(x)}$ is in the f^* subderivatives range [6]. Hence, if it holds for $\forall x$, we obtain

$$\mathbb{D}_f(\mu\|\nu) = \mathbb{E}_{x\sim\nu}[T^*(x)] - \mathbb{E}_{x\sim\mu}[f^*(T^*(x))].$$

Such equality holds, generally under some light conditions.

Theorem 3. *Let $f(\cdot)$ be a strictly convex and continuously differentiable function on the domain $I \subseteq \mathbb{R}$ and let μ and ν be Borel two probability distributions on space $\mathbb{R}^n$ that satisfy $\mu \ll \nu$. Then, we have [6]*

$$\mathbb{D}_f(\mu\|\nu) = \sup_T \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))]\Big), \tag{15}$$

where $\sup_T$ is considered an overall Borel functions defined as $T : \mathbb{R}^n \longmapsto Dom(f^*)$. In addition, if the probability measure p satisfies $p(x) \in I$, $\forall x$, $T^*(x) := f'(p(x))$ is an optimizer of Eq. (15).

Proof We have obtained the upper bound for the problem in Eq. (14) showing the lower bound part will finish the proof. Let $p(x) = d\mu(x)/d\nu(x)$. Let us analyze Eq. (13) in detail by assuming

$q(x) = 1$, and $\sup_t \left\{ tp(x) - f^*(t) \right\}$ for each x. Let us express $g_x(t) = tp(x) - f^*(t)$, $S = Dom(f^*)$ and suppose $S^\circ = (a, b)$ where $a, b \in \mathbb{R} \cup \{\pm\infty\}$. Then, we can introduce a sequence $T_k(x)$ as follows:
If density function $p(x)$ is in the range of $f^{*\prime}$, say for instance $p(x) = f^{*\prime}(t_x)$, we formed $T_k(x) = t_x \in S$. If $p(x) - f^{*\prime} > 0$ for all t, then, $g_x(t)$ is a strictly increasing function. Hence, the supremum of $g_x(t)$ is achieved at the upper boundary point b. Therefore, we assign $T_k(x) = b_k \in S$, where $b_k \mapsto b^-$. Here, if $p(x) - f^{*\prime}(t) < 0$, $\forall t$, $g_x(t)$ becomes a strictly decreasing function. Therefore, in this case, the supremum of $g_x(t)$ is achieved at the lower boundary point a, and we assign $T_k(x) = a_k \in S$, where $a_k \mapsto a^+$. By Lemma 1 and its proof, we know that

$$\lim_{k \mapsto \infty} \Big(T_k(x)p(x) - f^*(T_k(x)) \Big) = \sup_t \{tp(x) - f^*(t)\}.$$

Thus,

$$\lim_{k \mapsto \infty} \Big(\mathbb{E}_{x\sim\nu}[T_k(x)] - \mathbb{E}_{x\sim\mu}[f^*(T_k(x))] \Big) = \mathbb{D}_f(\mu \| \nu).$$

To show the proof of the last, suppose $p(x) \in I$. Then, again by Lemma 1, define $s(t) = f'^{-1}(t)$ for t in the range of f', then we can write

$$f^{*\prime}(t) = \Big(ts(t) - f(s(t)) \Big)' = s(t) + ts'(t) - f'(s(t))s'(t) = s(t).$$

Hence, we have $g'_x(t) = p(x) - f^{*\prime}(t) = p(x) - f'^{-1}(t)$. Then, $g_x(t)$ has a maximum at $t = f'(p(x))$. This result proves that $T^* = f'(p(x))$ is an optimizer for Eq. (15). ■
Note that Theorem 3 holds for only $\mu \ll \nu$. However, one may give the following theorem for other cases.

Theorem 4. *Let $f(t)$ be a convex function where the domain of f^* includes (a, ∞) for some $a \in \mathbb{R}$. Let μ and ν be two Borel probability measures on $\mathbb{R}^n$ that satisfy $\mu \not\ll \nu$. Then,*

$$\sup_T \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))] \Big) = +\infty,$$

holds. Here, $\sup_T$ is considered an overall Borel function defined as $T : \mathbb{R}^n \longmapsto Dom(f^*)$.

Proof Consider a new distribution defined as $\tau = \frac{1}{2}(\mu + \nu)$. Then, these two densities satisfy $\mu, \nu \ll \tau$. Moreover, let $p = d\mu/d\tau$ and $q = d\nu/d\tau$ be the Radon-Nikodym derivatives of the given densities. Here, we know that $\mu \not\ll \nu$. Therefore, we can find a set S_0 with $\mu(S_0) > 0$ on which $q(x) = 0$. Now, fix a point t_0 in the domain of f^*. Let us define $T_k(x) = k$ for $x \in S_0$, and $T_k(x) = t_0$ otherwise. Then we can introduce,

$$\mathbb{E}_{x\sim\mu}[T_k(x)] - \mathbb{E}_{x\sim\nu}[f^*(T_k(x))] \geq k\mu(S_0) - f^*(t_0)(1 - \nu(S_0)) \longmapsto +\infty$$

holds. This result leads us to the desired proof. ■
At this stage, notice that the domain of f^* has no boundary from above, and Eq. (15) is not satisfied unless we have $\mu \ll \nu$. In many studies, we face a singular target distribution μ, as the training

data we are handling might have a lower-dimensional manifold. Hence, we can introduce the following theorem.

Theorem 5. *Consider a function $f(\cdot)$ that is a lower semicontinuous convex function and the domain I^* of f^* has $\sup I^* = b^* < +\infty$. Let μ and ν be two Borel probability measures on space $\mathbb{R}^n$, and $\mu = \mu_s + \mu_{ab}$, where $\mu_s \perp \nu$ and $\mu_{ab} \ll \nu$. Then [3],*

$$\sup_T \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))] \Big) = \mathbb{D}_f(\mu\|\nu) + b^*\mu_s(\mathbb{R}^n),$$

where $\sup_T$ is carried over all Borel functions given as $T : \mathbb{R}^n \longmapsto Dom(f^*)$.

Proof First, let us define $\tau = \frac{1}{2}(\mu + \nu)$. Then, it is clear that $\mu, \nu \ll \tau$. Here, $\mu = \mu_{ab} + \mu_s$ decomposition is unique and assured by the celebrated *Lebesgue decomposition theorem*, where $\mu_{ab} \ll \nu$ and $\mu_s \perp \nu$. Furthermore, let $p_{ab} = d\mu_{ab}/d\tau$, $p_s = d\mu_s/d\tau$, and $q = d\nu/d\tau$ be the Radon-Nikodym derivatives of the densities. Here, we can divide $\mathbb{R}^n$ into $\mathbb{R}^n = \Omega \cup \Omega^c$, where $\Omega = supp(q)$. Then, we have $q(x) = p_{ab}(x) = 0$ for $x \in \Omega^c$ since we have $\mu_s \perp \nu$. Hence,

$$\begin{aligned}
\sup_T \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}[f^*(T_k(x))] \Big) &= \sup_T \int_\Omega \Big(T(x)p_{ab}(x) - f^*(T(x)))q(x) \Big) d\tau \\
&\quad + \sup_T \int_{\Omega^c} T(x)p_{ab}(x)d\tau \\
&= \sup_T \int_\Omega \left(T(x)\frac{p_{ab}(x)}{q(x)} - f^*(T(x)) \right) q(x)d\tau + b^*\mu_s(\Omega^c) \\
&= \int_\Omega f\left(\frac{p_{ab}(x)}{q(x)}\right) q(x)d\tau + b^*\mu_s(\mathbb{R}^n) \\
&= \int_\Omega f\left(\frac{p(x)}{q(x)}\right) q(x)d\tau + b^*\mu_s(\mathbb{R}^n) \\
&= \mathbb{D}_f(\mu\|\nu) + b^*\mu_s(\mathbb{R}^n).
\end{aligned}$$

■

Variational divergence minimization (VDM) with f-GANs

It is possible to generalize the standard vanilla GAN with the help of $f-$*divergence* measures. For a given probability distribution μ, f-GAN aims to minimize the distance between distributions via $\mathbb{D}_f(\mu\|\nu)$, concerning the probability distribution ν. Fulfilled in the sample space, f-GAN solves the min-max problem given as

$$\min_\nu \sup_T \Big(\mathbb{E}_{x\sim\nu}[T(x)] - \mathbb{E}_{x\sim\mu}[f^*(T(x))] \Big). \tag{16}$$

The f-GAN framework came on to stage primarily in [7], and the optimization problem given in Eq. (16) guides us to the (VDM).
Note that the *VDM* looks identical to the min-max problem given for the vanilla GAN. Here, the Borel function T is named a critic function, or shortly a critic. With the assumption $\mu \ll \nu$, by Theorem 3 it is equal to $\min_\nu \mathbb{D}_f(\mu\|\nu)$. As we mentioned earlier, one possible problem of the f-GAN is facing $\mu \not\ll \nu$ in Theorem 4. Then, Eq. (16) is generally not equal to $\min_\nu \mathbb{D}_f(\mu\|\nu)$. Luckily, some particularly selected f, such a case is not a problem anymore.

Theorem 6. *Suppose $f(t)$ is such a function that is lower semicontinuous and strictly convex, and the domain denoted as I^* of convex conjugate f^* satisfies $\sup I^* = b^* \in [0, \infty)$. Additionally, suppose that f is a continuously differentiable function on its domain and satisfies $f(t) > 0, \forall t \in (0, 1)$, and let μ be Borel probability measures on space $\mathbb{R}^n$. Under these assumptions, we obtain our unique optimizer for [7]*

$$\inf_{\nu} \sup_{T} \Big(\mathbb{E}_{x\sim\nu}[T(x)] - \mathbb{E}_{x\sim\mu}[f^*(T(x))] \Big),$$

as $\nu = \mu$. Here, $\sup_T$ is assessed overall Borel functions $T : \mathbb{R}^n \longmapsto Dom(f^*)$ while $\inf_\nu$ is assessed overall potential Borel probability measures.

Proof From Theorem 5, for any Borel probability measure ν, we can write the following

$$\sup_{T} \Big(\mathbb{E}_{x\sim\mu}[T(x)] - \mathbb{E}_{x\sim\nu}[f^*(T(x))] \Big) = \mathbb{D}_f(\mu\|\nu) + b^*\mu_s(\mathbb{R}^n) \geq \mathbb{D}_f(\mu\|\nu).$$

By Proposition 3, such equality holds if and only if $\nu = \mu$. Consequently, $\nu = \mu$ becomes our unique optimizer for GANs. ■

Some remarks on special solutions

Remark 6. *Suppose that both the density functions $p(x)$ and $q(x)$ satisfy $p(x) = q(x)$. Then, the optimal value becomes $D^*(x) = 1/2$. For such a special case, we have a loss function as [1]*

$$\begin{aligned} L(G^*, D^*) &= \int_{\mathbb{R}^n} \Big(p(x)\log(D^*(x)) + q(x)\log(1 - D^*(x)) \Big) dx \\ &= \log(\frac{1}{2}) \int_x p(x)dx + \log(\frac{1}{2}) \int_{\mathbb{R}^n} q(x)dx \\ &= -2log(2). \end{aligned}$$

Furthermore, if we calculate JS divergence, we have

$$\begin{aligned} \mathbb{D}_{JS}(\mu\|\nu) &= \frac{1}{2}\mathbb{D}_{KL}\Big(\mu\|\frac{\mu+\nu}{2}\Big) + \frac{1}{2}\mathbb{D}_{KL}\Big(\nu\|\frac{\mu+\nu}{2}\Big) \\ &= \frac{1}{2}\Big(\log(2) + \int_{\mathbb{R}^n} p(x)\log\Big(\frac{p(x)}{p(x)+q(x)}\Big)dx \Big) \\ &\quad + \frac{1}{2}\Big(\log(2) + \int_{\mathbb{R}^n} q(x)\log\Big(\frac{q(x)}{p(x)+q(x)}\Big)dx \Big) \\ &= \frac{1}{2}\Big(\log(4) + L(G, D^*) \Big). \end{aligned} \tag{17}$$

By rearranging Eq. (17), we find

$$L(G, D^*) = 2\mathbb{D}_{JS}(\mu\|\nu) - 2\log(2).$$

As an immediate result, we can also give the following remark.

Remark 7. *Under the assumptions given in the preceding remark, the followings hold [3]:*

- *Fundamentally, the objective of a GAN loss function is to quantify the similarity between the generated data distribution ν and the real sample distribution μ by using $\mathbb{D}_{JS}$ under the optimal discriminator D*

condition. The best generator (G^*) *imitates the distribution of real data, which leads us to the minimum given as* $L(G^*, D^*) = -2\log 2$.

- *If we train the discriminator* D *until it convergences, its error approximates* 0. *This indicates that the* $\mathbb{D}_{JS}$ *between the distributions has reached its maximum (it is easy to see that* $0 \leq \mathbb{D}_JS(\mu\|\nu) \leq \ln(2)$*). We can find it only if their distributions are not continuous (meaning: their densities are not absolutely continuous functions) or the distributions have disjoint supports. One potential reason behind the noncontinuity of the distribution is if their supports rely on low-dimensional manifolds. For such a case, there is substantial empirical and theoretical evidence to believe that the generated data distribution* ν *is focused on a low-dimensional manifold for many datasets.*
- *If both* μ *and* ν *rest in low-dimensional manifolds, they are almost undoubtedly disjoint. If the distributions have disjoint supports, we can always find a perfect discriminator that divides real and fake samples 100% accurately.*

4 Concluding remarks

In this study, we discovered and explored the mathematical background of GANs to illustrate a deep understanding of them for further extensions. Hence, in this study, we took a detailed tour of the mathematics behind GANs. After the celebrated work of Goodfellow et al. [1], new adversarial training objectives and techniques for generative modeling have been developed, such as Wasserstein GANs [8, 9]. Furthermore, GANs have been widely applied to new fields of research, including mathematical finance [10–12], time series generation [13, 14], audio synthesis [15], and fraud detection in financial datasets [16]. The underlying mathematics for these models are obviously different from what we have discussed above, but the study is a good starting point nonetheless.

Declarations

Ethical approval

The authors state that this research complies with ethical standards. This research does not involve either human participants or animals.

Consent for publication

Not applicable.

Conflicts of interest

The authors declare that they have no conflict of interest.

Funding

This research has been funded by the German Ministry of Education Research (BMBF) within the project *Analytisch-generative Netzwerke zur Systemidentifikation (AGENS)* (Grant no: 05M20UKA).

Author's contributions

B.Y.: Conceptualization, Methodology, Software, Data Curation, Writing-Original draft preparation. R.K.: Visualization, Investigation, Supervision, Validation, Writing-Reviewing and Editing. Both authors have made substantial contributions to the conception and design of the work. They have read and agreed to the published version of the manuscript.

Authors' information

- Dr. Bilgi Yilmaz holds an M.Sc. and Ph.D. degree in Financial Mathematics from the Institute of Applied Mathematics at Middle East Technical University, Türkiye. Currently, Dr. Yilmaz is a post-doctoral researcher at RPTU Kaiserslautern, Germany.
- Prof. Dr. Ralf Korn studied mathematics and business administration at the University of Mainz, Germany. After completing his doctorate and habilitation in Mathematics, Professor Korn received a call to the Technical University of Kaiserslautern in 1999, where he has been a professor of Financial Mathematics in the Mathematics Department ever since. One year later, he founded the Financial Mathematics department of the Fraunhofer Institute for Industrial Mathematics in Kaiserslautern, which he has headed ever since.
  In addition, he has been one of the co-founders of the European Institute for Quality Management of Financial Mathematical Products and Processes EI-QFM in Kaiserslautern since 2010. Professor Korn is a member of the German Society for Insurance and Financial Mathematics (DGVFM), where he was a member of the board from 2003 to 2009 and has been deputy chairman of the board since 2011.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S. et al. Generative adversarial nets. *Advances in Neural Information Processing Systems*, 27, (2014).

[2] de Meer Pardo, F. Enriching financial datasets with generative adversarial networks. *MS thesis, Delft University of Technology, The Netherlands*, (2019).

[3] Wang, Y. A mathematical introduction to generative adversarial nets (GAN). *ArXiv Prints, ArXiv:2009.00169,* (2020). [CrossRef]

[4] Arjovsky, M. and Léon, B. Towards principled methods for training generative adversarial Networks. *ArXiv Prints, arXiv:1701.04862,* (2017). [CrossRef]

[5] Syed A.M. and Samuel, D.S. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological),* 28(1), 131-142, (1996). [CrossRef]

[6] Nguyen, X., Wainwright, M.J. and Jordan M.I. Estimating divergence functionals and the likelihood ratio by convex risk minimization. In Proceedings, *IEEE Transactions on Information Theory*, 56(11), pp. 5847-5861, (2010, October). [CrossRef]

[7] Nowozin, S., Cseke, B. and Tomioka, R. f-GAN: Training generative neural samplers using variational divergence minimization. In Proceedings, *Advances in Neural Information Processing Systems 29 (NIPS),* (2016, December).

[8] Arjovsky, M., Chintala, S. and Bottou, L. Wasserstein generative adversarial networks. In Proceedings *Proceedings of the 34th International Conference on Machine Learning (PMLR),* pp. 214-223, (2017, July).

[9] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A.C. Improved training of Wasserstein GANs. In Proceedings *Advances in neural information processing systems 30 (NIPS),* (2017, December).

[10] Ni, H., Szpruch, L., Wiese, M., Liao, S. and Xiao, B. Conditional sig-wasserstein gans for time series generation. *ArXiv Preprint, arXiv:2006.05421,* (2020). [CrossRef]

[11] Wiese, M., Bai, L., Wood, B. and Buehler, H. Deep hedging: learning to simulate equity option markets. *ArXiv Preprint, arXiv:1911.01700* (2019). [CrossRef]

[12] Wiese, M., Knobloch, R., Korn, R. and Kretschmer P. Quant GANs: deep generation of financial time series. *Quantitative Finance,* 20(9), 1419-1440, (2020).

[13] Ni, H., Szpruch, L., Sabate-Vidales, M., Xiao, B., Wiese, M. and Liao, S. Sig-Wasserstein GANs for time series generation. In Proceedings *of the Second ACM International Conference on AI in Finance (ICAIF),* pp. 1-8, (2021, November). [CrossRef]

[14] Yoon, J., Jarrett, D. and Van der Schaar, M. Time-series generative adversarial networks. *Advances in Neural Information Processing Systems 32 (NeurIPS),* (2019, December).

[15] Donahue, C., McAuley, J. and Puckette, M. Adversarial audio synthesis. *ArXiv Preprint, arXiv:1802.04208,* (2018). [CrossRef]

[16] Schreyer, M., Sattarov, T., Reimer, B. and Borth, D. Adversarial learning of deepfakes in accounting. *ArXiv Preprint, arXiv:1910.03810,* (2019). [CrossRef]

Mathematical Modelling and Numerical Simulation with Applications (MMNSA)
(https://dergipark.org.tr/en/pub/mmnsa)

How to cite this article: Yilmaz, B. & Korn, R. (2023). Understanding the mathematical background of Generative Adversarial Networks (GANs). *Mathematical Modelling and Numerical Simulation with Applications,* 3(3), 234-255. https://doi.org/10.53391/mmnsa.1327485