Report paper: Generative Adversarial Nets

Giáp Ngọc Hiệu

**Tóm tắt nội dung**

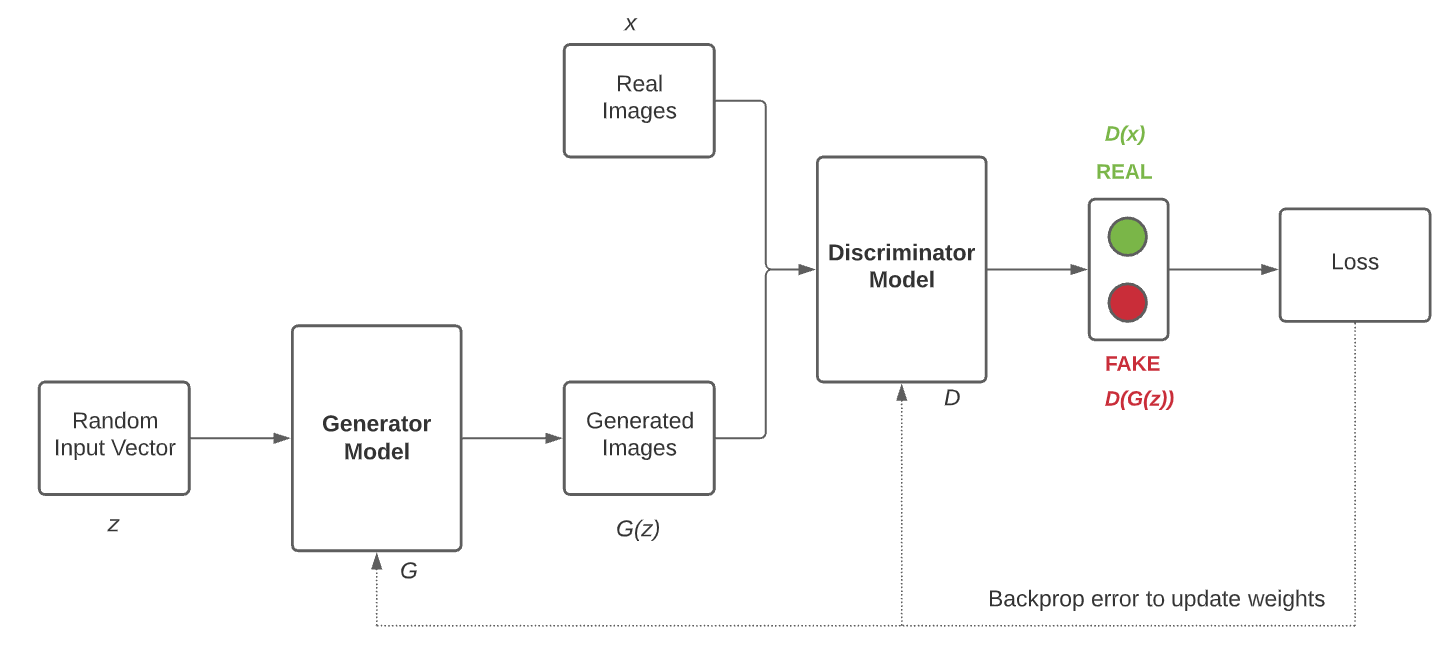
Bài report này là phần tóm tắt lại ý tưởng, nội dung chính, những điều làm được của paper

Generative Adversarial Nets

1. **Ý tưởng**

Các tác giả đề xuất một framework mới để ước tính các mô hình tổng quát thông qua một quy trình đối nghịch, trong đó tác giả đồng thời đào tạo hai mô hình: mô hình tổng quát G nắm bắt phân phối dữ liệu và mô hình phân biệt D ước tính xác suất mà một mẫu đến từ quá trình đào tạo dữ liệu thay vì G. Quy trình huấn luyện cho G là tối đa hóa xác suất D mắc lỗi. Khung này tương ứng với trò chơi hai người chơi minimax. Trong không gian của các hàm tùy ý G và D, tồn tại một giải pháp duy nhất, với G khôi phục phân phối dữ liệu huấn luyện và D bằng 1/2 ở mọi nơi. Trong trường hợp G và D được xác định bởi các perceptron đa lớp, toàn bộ hệ thống có thể được huấn luyện bằng lan truyền ngược. Không cần bất kỳ chuỗi Markov nào hoặc mạng suy luận gần đúng không kiểm soát trong quá trình đào tạo hoặc tạo mẫu.

1. **Phương pháp**
   1. **Adversarial nets**



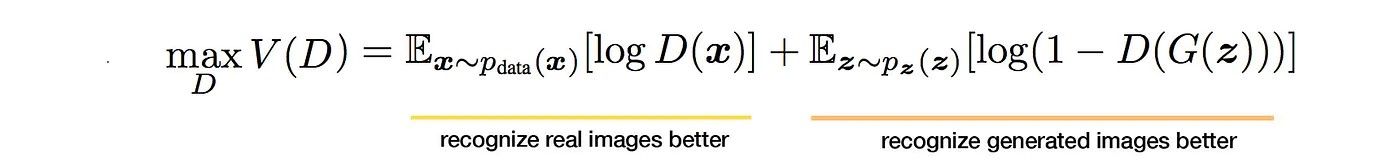
- Hai model được đào tạo là Generative model G và Discriminative model D. Quy trình đào tạo cho G là tối đa hóa xác suất D mắc lỗi.

- Hai model này có thể là bất kỳ loại model và thuật toán tối ưu nào. Trong bài báo này, cả hai đều là multilayer perceptron.

Nói cách khác, D và G chơi trò chơi minimax hai người chơi sau với hàm giá trị V(G, D):

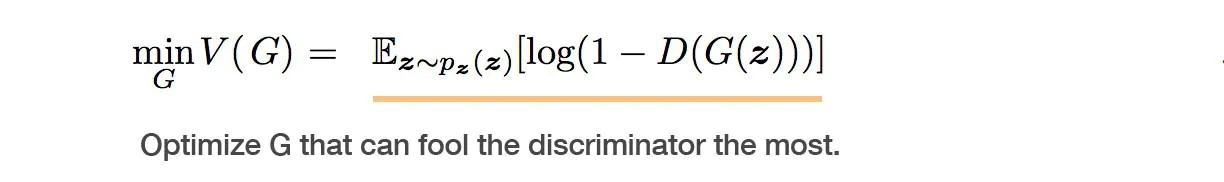


**Discriminator D**



* D phân loại mẫu đầu vào xem đó là dữ liệu thật hay dữ liệu giả do G tạo ra.
* D là một multilayer perceptron D(x;θ\_d) có đầu ra là một scalar. D(x) đại diện cho xác suất mà x đến từ dữ liệu chứ không phải từ G.
* D được huấn luyện để tối đa hóa xác suất gán nhãn chính xác cho cả các mẫu huấn luyện và các mẫu từ G.
* Vì D(x) đưa ra xác suất nằm trong khoảng từ 0 đến 1, log của D(x) nằm trong khoảng từ âm vô cùng đến 0.
* Khi D đoán đúng dữ liệu thực thì D(x) gần bằng 1, log D(x) gần bằng 0 có thể cực đại hàm trên.
* Khi D đoán đúng dữ liệu giả thì D(G(z)) gần bằng 0, log (1-D(G(z))) gần bằng 0 có thể cực đại hàm trên.

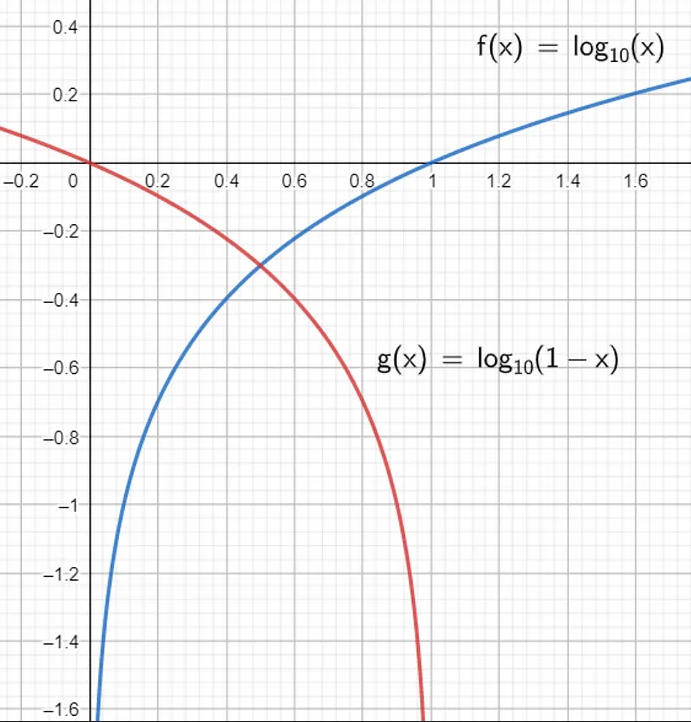
**Generator G**



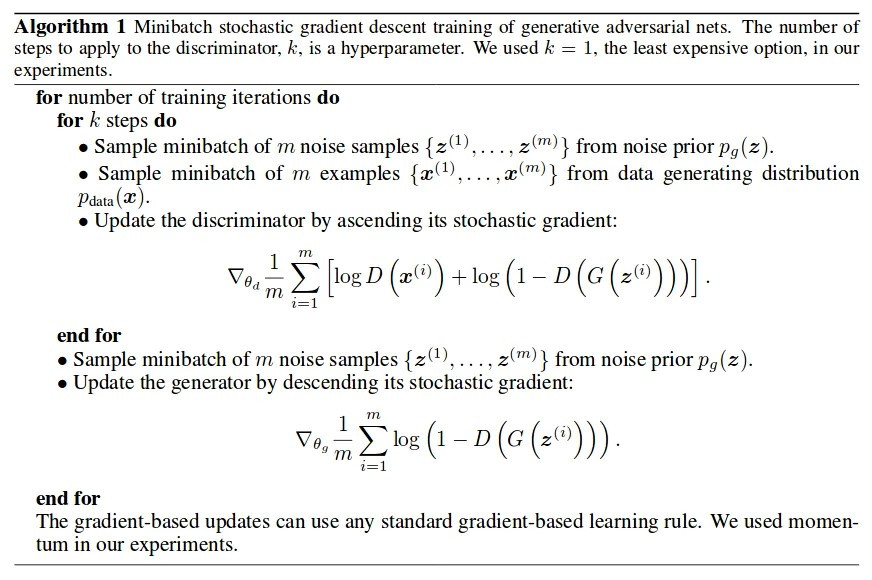
* G tạo ra mẫu trông giống như mẫu x trong tập dữ liệu thực.
* Để tìm hiểu phân phối của generator p\_g trên dữ liệu x, đầu vào được xác định trước là noise variables p\_z(z), sau đó biểu diễn ánh xạ tới không gian dữ liệu dưới dạng G(z;θ\_g), trong đó G là hàm khả vi được biểu diễn bởi multilayer perceptron với tham số θ\_ g.
* G cũng được đào tạo đồng thời để giảm thiểu log(1-D(G(z))).
* Mục tiêu của G là tạo ra các mẫu sao cho D không thể phân biệt được đó là dữ liệu thật hay giả và cuối cùng D chỉ có thể đoán ngẫu nhiên p = 1/2.
* Trong thực tế, phương trình (1) có thể không cung cấp đủ gradient để G học tốt. Trong thời gian đầu học, khi G kém, D có thể từ chối các mẫu với độ tin cậy cao vì chúng khác biệt rõ ràng với dữ liệu huấn luyện. Trong trường hợp này,

log(1 − D(G(z))) bão hòa, tức là - log(1 -D(G(z))) → 0. Thay vì huấn luyện G để cực tiểu hóa log(1 − D(G(z))), chúng ta có thể huấn luyện G để cực đại hóa

log D(G(z)). Hàm mục tiêu này dẫn đến cùng một điểm cố định của động lực học của G và D nhưng cung cấp độ dốc mạnh hơn nhiều ngay từ đầu trong quá trình học.



* 1. **Algorithm 1**

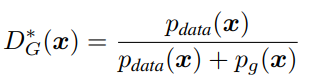


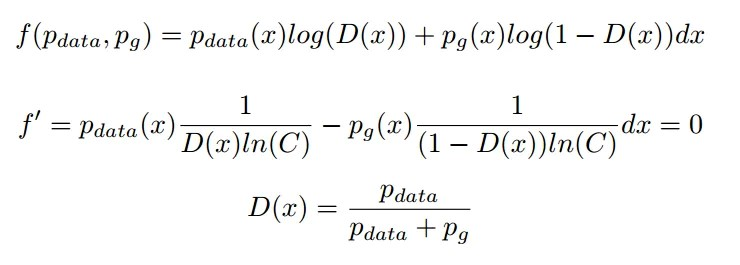
* 1. **Theoretical Results**

Trong phần tiếp theo, chúng tôi trình bày một phân tích lý thuyết về các mạng đối thủ, về cơ bản chỉ ra rằng tiêu chí đào tạo cho phép một người khôi phục phân phối tạo dữ liệu khi G và D được cung cấp đủ dung lượng, tức là trong giới hạn không tham số. Xem Hình 1 để biết cách giải thích ít trang trọng hơn, mang tính sư phạm hơn về phương pháp này. Trong thực tế, chúng ta phải triển khai trò chơi bằng cách sử dụng cách tiếp cận số, lặp đi lặp lại. Việc tối ưu hóa D để hoàn thành trong vòng đào tạo bên trong là điều cấm kỵ về mặt tính toán và trên các bộ dữ liệu hữu hạn sẽ dẫn đến việc khớp quá mức. Thay vào đó, chúng tôi xen kẽ giữa k bước tối ưu hóa D và một bước tối ưu hóa G. Điều này dẫn đến việc D được duy trì gần giải pháp tối ưu của nó, miễn là G thay đổi đủ chậm.

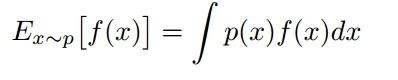
**Global Optimality of pg = pdata**

Mệnh đề 1: Đối với G cố định, bộ phân biệt tối ưu D là:



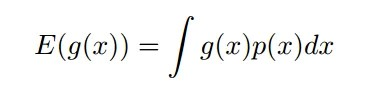


Ta có giá trị kỳ vọng của f(x) đối với phân bố p(x):



Định lý LOTUS nói rằng nếu g( x ) = x và người ta biết p( x )

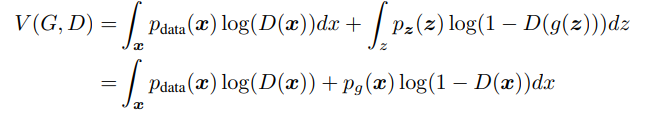
E ( g( x )) có thể được tìm thấy bằng cách,



Và ta biết rằng:



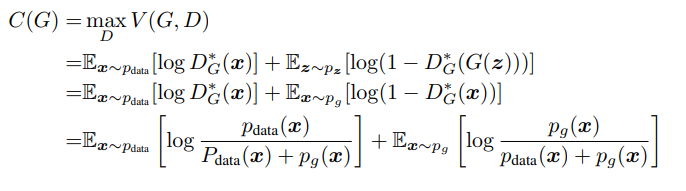
Tiêu chí đào tạo cho bộ phân biệt D, với bất kỳ trình tạo G nào, là tối đa hóa V (G, D):



Với (a, b) ∈ R2 \ {0, 0} bất kỳ, hàm y → a log(y) + b log(1 − y) đạt cực đại trong [0, 1] tại a / (a+b)

Mục tiêu đào tạo cho D có thể được hiểu là tối đa hóa log-likelihood để ước tính xác suất có điều kiện P (Y = y|x), trong đó Y cho biết liệu x đến từ pdata (với y = 1) hay từ

pg ( với y = 0). Phương trình (1) bây giờ có thể được phát biểu lại thành:

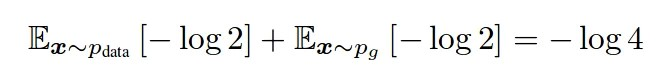


Định lý 1. Cực tiểu toàn cục của tiêu chí huấn luyện ảo C(G) đạt được khi và chỉ khi

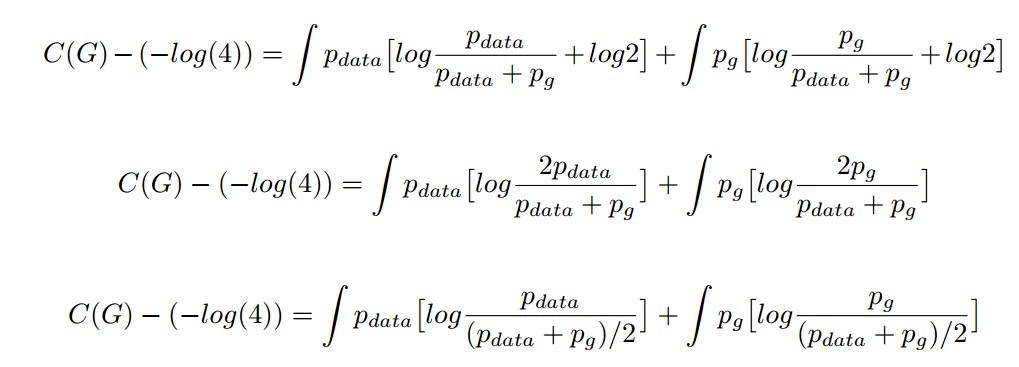
pg = pdata. Tại thời điểm đó, C(G) đạt được giá trị −log 4.

Cho pg = pdata. Sau đó, D \* ( x ) = 1/2, bằng cách tối đa hóa trò chơi. Sau đó,

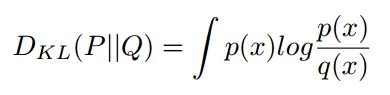
C ( G ) = log ( 1/2 ) + log ( 1/2 ) = -log4. Để thấy rằng đây cũng là điểm tối thiểu toàn cầu của C ( G ) ta trừ đi,



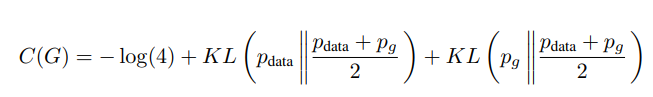
từ C ( G ) và nhận được,



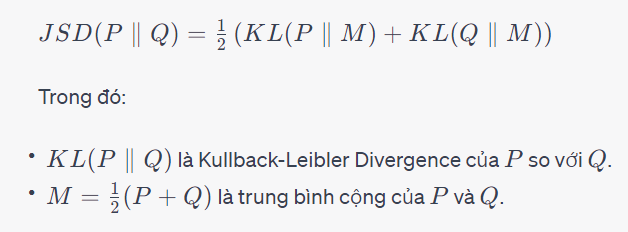
KL divergence là thước đo mức độ khác biệt của một bản phân phối nhất định so với bản phân phối thứ hai. Định nghĩa của KL divergence là,



Bằng cách áp dụng điều này vào phương trình trên, ta nhận được,



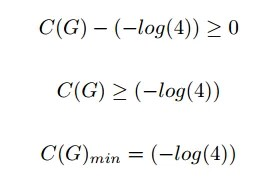
Theo định nghĩa, Jenson- Shanon divergence giữa hai bản phân phối được đưa ra là,



Sử dụng nó, ta nhận được



Do Jenson- Shanon divergence giữa hai phân phối luôn không âm và chỉ bằng 0 khi chúng bằng nhau,



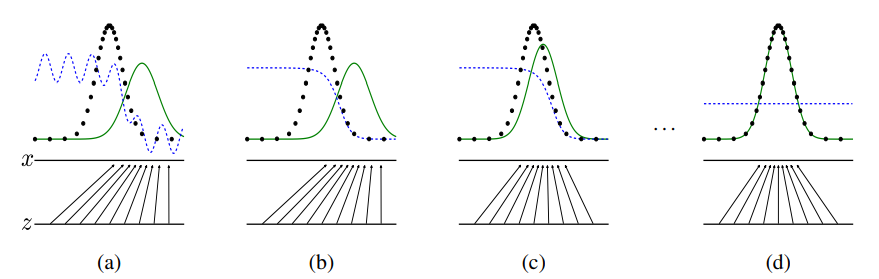
nên ta chỉ ra rằng C∗ = − log(4) là cực tiểu toàn cục của C(G) và nghiệm duy nhất là pg = pdata (D\*(x) = ½), tức là mô hình tổng quát sao chép hoàn hảo quá trình tạo dữ liệu .

**Convergence of Algorithm 1**

Mệnh đề 2. Nếu G và D có đủ khả năng và ở mỗi bước của Thuật toán 1, bộ phân biệt được phép đạt đến mức tối ưu cho G và pg được cập nhật để cải thiện



thì pg hội tụ về pdata



Mạng được huấn luyện bằng cách cập nhật đồng thời phân phối của Discriminator (D, màu xanh lam, đường đứt nét) để nó phân biệt giữa các mẫu từ phân phối tạo dữ liệu (màu đen, đường chấm) px với các mẫu của phân phối tổng hợp pg (G) (xanh lục, đường vẽ liền). Đường ngang phía dưới là miền mà từ đó z được lấy mẫu, trong trường hợp này là đồng nhất. Đường ngang ở trên là một phần của miền xác định của x. Các mũi tên hướng lên chỉ ra cách ánh xạ x = G(z) áp đặt pg phân bố không đồng đều trên các mẫu được biến đổi. G co lại ở những vùng có mật độ cao và giãn nở ở những vùng có mật độ pg thấp.

(a) Xét một cặp đối nghịch gần hội tụ: pg tương tự như pdata và D là một bộ phân loại chính xác một phần.

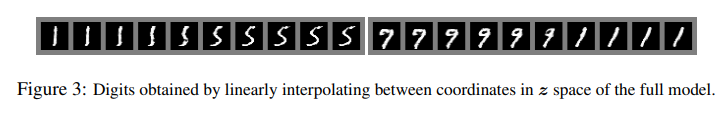
(b) Trong vòng lặp bên trong của thuật toán D được huấn luyện để phân biệt các mẫu với dữ liệu, hội tụ đến D∗(x) = pdata(x) / (pdata(x)+pg (x)) .

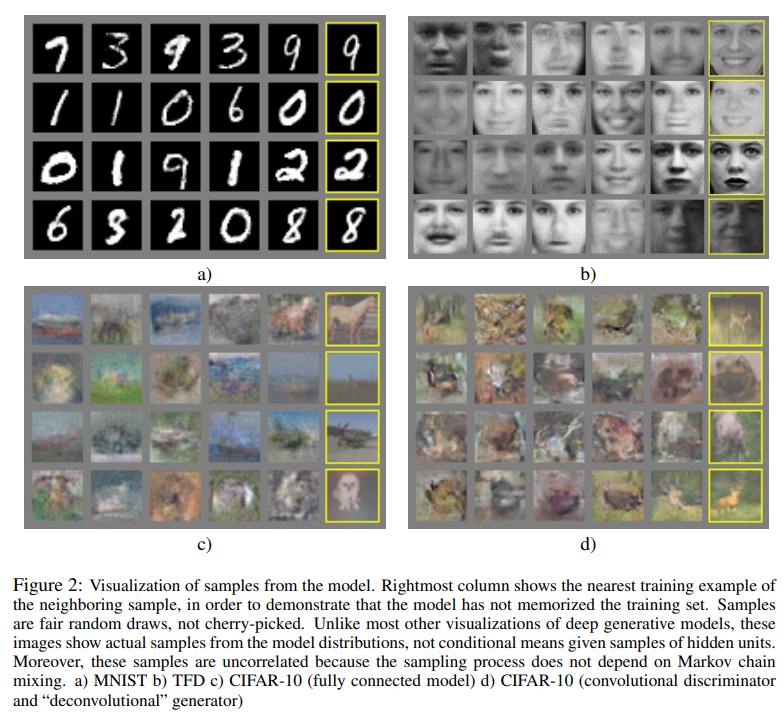
(c) Sau khi cập nhật lên G, độ dốc của D đã hướng dẫn G(z) chuyển sang các vùng có nhiều khả năng được phân loại là dữ liệu hơn.

(d) Sau vài bước huấn luyện, nếu G và D có đủ năng lực, chúng sẽ đạt đến điểm mà cả hai đều không thể hội tụ vì pg = pdata. Bộ phân biệt không thể phân biệt giữa hai phân phối, tức là D(x) = 1/2

1. **Kết quả**

Một số mẫu được lấy từ G:





1. **Note**

- Ưu điểm:

+ GAN có thể tạo ra hình ảnh chất lượng cao trông giống thật đối với con người. + Nó có thể hữu ích trong việc tạo ra các mẫu dữ liệu đa dạng, hữu ích cho việc đào tạo các mô hình học máy.

+ GAN có khả năng học từ dữ liệu có ít hoặc không có thông tin nhãn, điều này rất hữu ích cho các nhiệm vụ học tập không giám sát

- Nhược điểm:

+ Non-convergence: các tham số của mô hình dao động, mất ổn định và không bao giờ hội tụ

+ Mode collapse: generator chỉ tạo ra một số lượng mẫu hạn chế thay vì đa dạng như mong muốn.

+ Diminished gradient: Discriminator quá thành công đến nỗi gradient của Generator biến mất và không học được gì

+ Mất cân bằng giữa Discriminator và Generator gây ra tình trạng quá khớp

+ Rất nhạy cảm với việc lựa chọn siêu tham số.