SIMPÓSIO DAS EXATAS
UniFOA
2025

RELATO DE EXPERIÊNCIA

Redes adversárias generativas para geração de dígitos manuscritos: Uma implementação em pytorch

Vitor Amadeu Souza[1]

1 – UniFOA, Centro Universitário de Volta Redonda, Volta Redonda, RJ.

vitor.amadeu@foa.org.br

https://orcid.org/0009-0002-1857-6799

Resumo: Este artigo apresenta uma implementação de Redes Adversárias Generativas (GANs) utilizando o framework PyTorch para a geração de dígitos manuscritos do conjunto de dados MNIST. A arquitetura implementada consiste em redes neurais totalmente conectadas tanto para o gerador quanto para o discriminador, com o objetivo de aprender a distribuição dos dados reais e gerar amostras sintéticas convincentes. O treinamento foi conduzido ao longo de 50 épocas com um lote de 64 imagens, demonstrando a capacidade progressiva do modelo em gerar dígitos visualmente semelhantes aos reais. Os resultados indicam que mesmo com uma arquitetura relativamente simples, as GANs podem produzir imagens sintéticas de qualidade razoável, confirmando seu potencial para aplicações de geração de imagens.

Palavras-chave: Geração de Imagens. GAN. Python. MNIST.

INTRODUÇÃO

Redes Adversárias Generativas (GANs), introduzidas por Goodfellow et al. (2014), representam uma classe inovadora de modelos generativos que têm demonstrado resultados impressionantes na geração de conteúdo sintético, particularmente imagens. O funcionamento das GANs baseia-se em um cenário adversário, onde duas redes neurais, um gerador e um discriminador, competem entre si em um jogo de soma zero.

A base conceitual das GANs está no treinamento simultâneo de duas redes neurais: o gerador (G), que aprende a mapear de um espaço latente para o espaço de dados reais, e o discriminador (D), que aprende a distinguir entre dados reais e dados gerados. Durante o processo de treinamento, o gerador tenta produzir amostras cada vez mais realistas para "enganar" o discriminador, enquanto este último busca aprimorar sua capacidade de diferenciar entre dados reais e sintéticos (ARJOVSKY et al., 2017).

Desde sua introdução, as GANs têm encontrado aplicações em diversos domínios, incluindo síntese de imagens (KARRAS et al., 2019), tradução entre domínios de imagem (ZHU et al., 2017), geração de texto (ZHANG et al., 2019), e composição musical (DONG et al., 2018). No contexto da visão computacional, as GANs têm sido particularmente bem-sucedidas na geração de rostos humanos fotorrealistas, arte, e na transformação de esboços em imagens detalhadas.

O conjunto de dados MNIST, contendo 70.000 imagens de dígitos manuscritos (LECUN et al., 1998), representa um ponto de partida padrão para muitas tarefas de visão computacional e aprendizado de máquina. Por sua natureza relativamente simples, mas ainda desafiadora, o MNIST serve como uma excelente base de teste para arquiteturas generativas como as GANs.

Neste artigo, apresento uma implementação de GANs utilizando PyTorch, focada na geração de dígitos manuscritos do conjunto MNIST. Adoto uma abordagem com redes totalmente conectadas tanto para o gerador quanto para o discriminador, explorando seu desempenho na tarefa de geração de imagens sem recorrer a arquiteturas mais complexas como redes convolucionais.

METODOLOGIA

A implementação utilizou o framework PyTorch para o desenvolvimento de uma GAN composta por redes neurais totalmente conectadas. Os principais componentes da arquitetura são o Gerador (G), responsável por transformar vetores de ruído aleatório em imagens sintéticas, implementado como uma rede neural feedforward com quatro camadas. A arquitetura específica consiste em uma camada de entrada que mapeia o vetor latente de dimensão 100 para 128 neurônios, seguida por uma função de ativação ReLU; duas camadas intermediárias com 256 e 512 neurônios, respectivamente, ambas seguidas por funções de ativação ReLU; e uma camada de saída que produz um vetor de dimensão 784 (equivalente a uma imagem 28×28), seguida por uma função de ativação Tanh para assegurar que os valores de pixel estejam no intervalo [-1, 1].

O Discriminador (D), encarregado de classificar imagens como reais ou geradas, foi implementado como uma camada de entrada que recebe a imagem vetorizada (784 dimensões) e a mapeia para 512 neurônios, seguida por uma função de ativação LeakyReLU com inclinação negativa de 0.2; uma camada intermediária de 256 neurônios com ativação LeakyReLU; e uma camada de saída que produz um único valor, seguida por uma função Sigmoid para obter uma probabilidade entre 0 e 1, indicando a autenticidade da imagem.

O processo de treinamento foi configurado com os seguintes hiperparâmetros: dimensão do espaço latente: 100; formato da imagem: 28×28×1 (escala de cinza); tamanho do lote (batch size): 64; taxa de aprendizado: 0.0002; número de épocas: 50; dispositivo de processamento: GPU (CUDA) quando disponível, caso contrário CPU. Os dados do MNIST foram pré-processados através de normalização para o intervalo [-1, 1], consistente com a função de ativação Tanh do gerador.

O treinamento seguiu o procedimento padrão para GANs, com algumas particularidades: na preparação dos dados, o conjunto MNIST foi carregado utilizando as funcionalidades do torchvision, com aplicação de transformações para normalização; na inicialização dos modelos, tanto o gerador quanto o discriminador foram instanciados e movidos para o dispositivo de processamento designado; para funções de perda e otimizadores, foi utilizada a função de perda de entropia cruzada

binária (BCE) para ambas as redes, e os otimizadores Adam foram configurados com uma taxa de aprendizado de 0.0002 para ambos os modelos.

No ciclo de treinamento, a cada época, o processo consistiu em treinamento do gerador para maximizar a probabilidade de o discriminador classificar incorretamente as imagens geradas como reais, e treinamento do discriminador para minimizar o erro de classificação tanto em imagens reais quanto geradas. Para monitoramento, a cada época, foram registradas as perdas do gerador e do discriminador para avaliar o progresso do treinamento. A cada 10 épocas, foram geradas 16 imagens sintéticas para inspeção visual da qualidade da geração.

A avaliação do desempenho da GAN foi realizada através de análise das curvas de perda do gerador e do discriminador ao longo do treinamento, e inspeção visual das imagens geradas em diferentes estágios do treinamento. Uma limitação importante desta implementação foi a ausência de métricas quantitativas mais rigorosas, como Inception Score (IS) ou Fréchet Inception Distance (FID), que são comumente utilizadas para avaliar o desempenho de modelos generativos.

O código-fonte desta implementação está disponível no link: https://github.com/vitor-souza-ime/gan e o código-fonte foi testado no Google Colab.

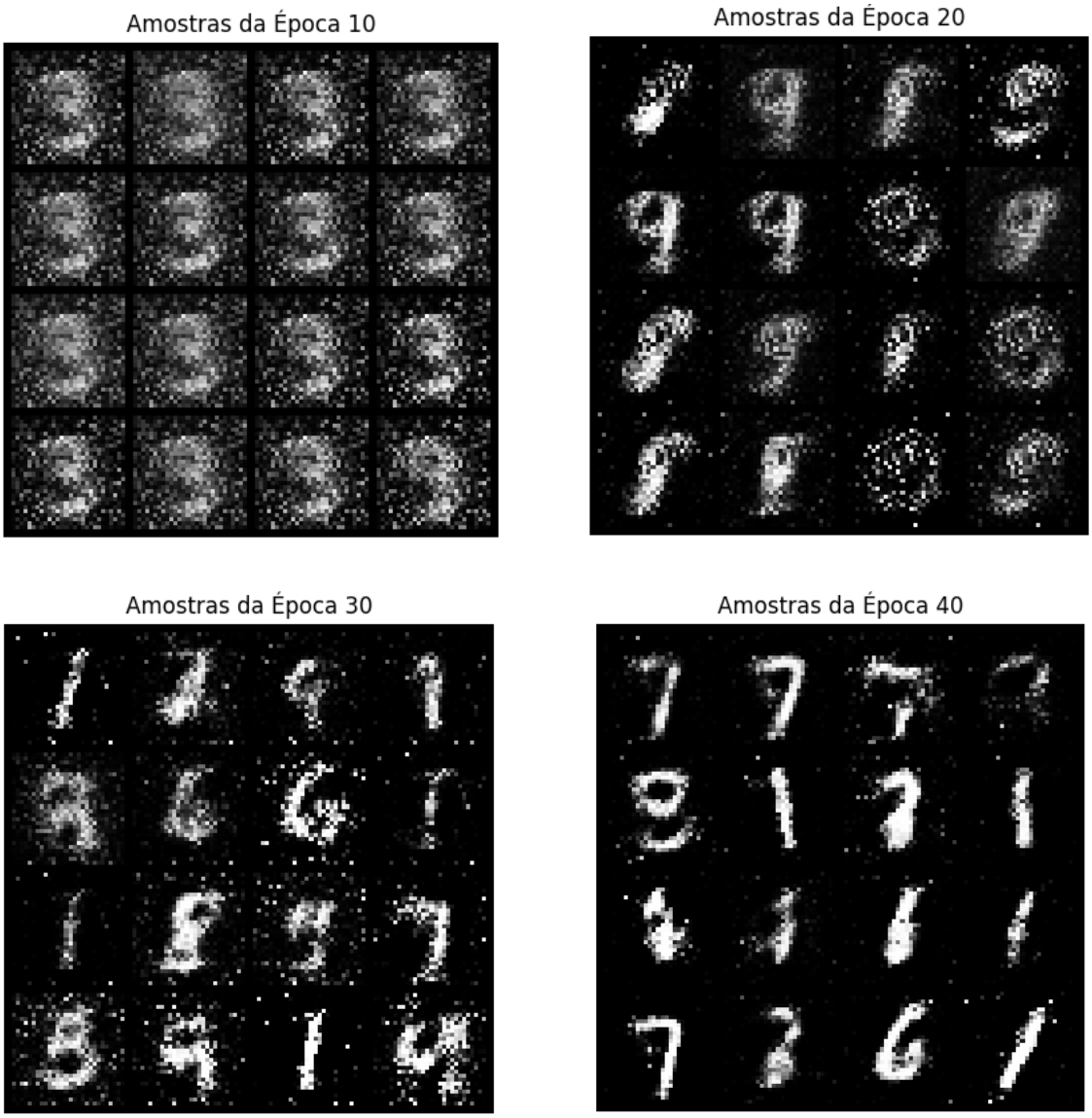
RESULTADOS E DISCUSSÃO

O treinamento da GAN ao longo das 50 épocas revelou padrões interessantes na dinâmica entre o gerador e o discriminador. Inicialmente, ambos os modelos apresentaram perdas relativamente altas, indicando a dificuldade tanto do gerador em produzir imagens convincentes quanto do discriminador em classificá-las corretamente.

A evolução das perdas do gerador e do discriminador ao longo das épocas de treinamento mostrou que, após um período inicial de oscilações, as perdas tendem a se estabilizar, sugerindo um equilíbrio no jogo adversário entre as duas redes. Um fenômeno interessante observado foi a ocorrência de momentos em que a perda do discriminador aumentava brevemente, correspondendo a períodos em que o gerador conseguia produzir imagens mais convincentes. Este comportamento está alinhado com a teoria das GANs, onde melhorias em um modelo frequentemente levam a

desafios temporários para o outro. A figura 1 apresenta o treinamento a cada época que foi gerado pelo modelo.

Figura 1 – Amostras das épocas 10 a 50



Fonte: O autor

As amostras geradas em diferentes estágios do treinamento demonstram uma progressão clara na qualidade e na definição dos dígitos sintéticos. Nas primeiras épocas (10-20), as imagens geradas apresentaram características vagas de dígitos, com formas amorfas e pouca definição. À medida que o treinamento progrediu (épocas 30-40), formas mais definidas começaram a emergir, com contornos mais claros e uma estrutura mais próxima dos dígitos reais.

Ao final do treinamento (época 50), o gerador foi capaz de produzir imagens que claramente se assemelham a dígitos manuscritos, embora ainda apresentem algumas imperfeições e artefatos. É notável que, mesmo com uma arquitetura relativamente simples baseada em redes totalmente conectadas, o modelo conseguiu capturar aspectos essenciais da distribuição dos dígitos MNIST.

Para aprimorar o desempenho, sugere-se implementar arquiteturas convolucionais tanto para o gerador quanto para o discriminador, seguindo modelos como DCGAN (RADFORD et al., 2016); incorporar técnicas de estabilização como normalização espectral (MIYATO et al., 2018) e regularização por gradiente (GULRAJANI et al., 2017); e explorar variantes mais recentes como StyleGAN (KARRAS et al., 2019) para potencialmente melhorar a qualidade das imagens geradas.

CONCLUSÕES

Este trabalho apresentou uma implementação de Redes Adversárias Generativas (GANs) utilizando PyTorch para a geração de dígitos manuscritos do conjunto MNIST. Os resultados demonstram que mesmo com uma arquitetura relativamente simples, baseada em redes neurais totalmente conectadas, é possível obter resultados visualmente satisfatórios na geração de imagens sintéticas.

Desta forma, as GANs continuam a representar uma abordagem poderosa para a geração de conteúdo sintético, com aplicações potenciais em diversos domínios além da visão computacional. Este trabalho contribui para a compreensão prática de sua implementação e comportamento, fornecendo insights valiosos para futuras investigações neste campo em rápida evolução.

REFERÊNCIAS

ARJOVSKY, M.; CHINTALA, S.; BOTTOU, L. Wasserstein GAN. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, 34., 2017, Sydney. Proceedings. Sydney: PMLR, 2017. v. 70, p. 214-223.

DONG, H. W. et al. MuseGAN: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In: AAAI CONFERENCE ON ARTIFICIAL INTELLIGENCE, 32., 2018. Proceedings. Palo Alto, CA: AAAI, 2018. v. 32, n. 1.

GOODFELLOW, I. et al. Generative adversarial nets. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 27., 2014, Montreal. Proceedings. Red Hook, NY: Curran Associates, 2014. p. 2672-2680.

GULRAJANI, I. et al. Improved training of Wasserstein GANs. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 30., 2017. Proceedings. Red Hook, NY: Curran Associates, 2017. p. 5767-5777.

KARRAS, T.; LAINE, S.; AILA, T. A style-based generator architecture for generative adversarial networks. In: IEEE/CVF CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, 2019, Long Beach. Proceedings. Piscataway, NJ: IEEE, 2019. p. 4401-4410.

LECUN, Y. et al. Gradient-based learning applied to document recognition. Disponível em: http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf. Acesso em: 8 maio 2025.

MIYATO, T. et al. Spectral normalization for generative adversarial networks. Disponível em: https://arxiv.org/abs/1802.05957. Acesso em: 8 maio 2025.

RADFORD, A.; METZ, L.; CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. Disponível em: https://arxiv.org/abs/1511.06434. Acesso em: 8 maio 2025.

ZHANG, H. et al. StackGAN++: Realistic image synthesis with stacked generative adversarial networks. arXiv preprint, arXiv:1710.10916, 2017. Disponível em: https://doi.org/10.48550/arXiv.1710.10916. Acesso em: 8 maio 2025.

ZHU, J. Y. et al. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE INTERNATIONAL CONFERENCE ON COMPUTER VISION, 2017, Venice. Proceedings. Piscataway, NJ: IEEE, 2017. p. 2223-2232.