EEL7514: Estudo sobre GANs

Enrique T. R. Pinto
UFSC
CTC-EEL
Florianópolis, Brasil
enriquetrp@hotmail.com

Thiago Motta
UFSC
CTC-EEL
Florianópolis, Brasil
thiagomotta94@gmail.com

***Resumo*—Este trabalho final busca apresentar uma pesquisa informativa acerca das Redes Adversarias Geradoras (Generative Adversarial Networks), popularmente conhecidas como GANs, descrevendo algumas práticas utilizadas no ramo para resolver (ou apenas mitigar) problemas inerentes da estrutura das redes. Além de apresentar exemplos com o conjunto CIFAR-10, tentativas de gerar conjuntos de imagens com maior dimensão são apresentadas.**

***Index Terms*—GAN, image generation, CIFAR-10, minibatch discrimination, mode collapse**

I. Introduction

O conceito principal das GANs foi introduzido por Goodfellow em [1], onde descreve uma interação minimax na forma de um jogo de dois jogadores: uma rede geradora e uma rede discriminadora. Idealmente, o sistema atinge o equilíbrio de Nash, entretanto várias situações de falha existem durante o treinamento altamente dinâmico deste tipo de sistema, isso fez com que várias técnicas tenham sido desenvolvidas para estabilizar o processo de treinamento e facilitar a produção de resultados aproveitáveis.

Neste trabalho, serão discutidos:

- Os conceitos fundamentais descrevendo GANs;
- Cenários de falha e problemas recorrentes no treinamento;
- Ferramentas comuns utilizadas para estabilizar o treinamento das redes;
- Apresentação e discussão dos resultados obtidos.

II. Conceitos fundamentais

A. Jogo de soma zero

Um dos conceitos que define o treinamento das GANs é a ideia de um jogo de soma zero. Embora haja casos específicos em que o treinamento não é, a rigor, um jogo de soma zero, a grande maioria das GANs (e a proposta original) estão estruturadas sobre esta definição. Seguindo a teoria apresentada em [1]: define-se uma distribuição de ruído de entrada $p_z(z)$, e funções $G(z;\theta_g)$ e $D(x;\theta_d)$, representando o gerador e o discriminador respectivamente. O ruído z é entrada do gerador, que age como uma transformação do espaço do ruído para o "espaço de dados", que no caso estudado são imagens de 3 canais de cores. Os dados originais tem seus elementos são representados pela variável x. O discriminador possui apenas uma saída, que determina a probabilidade do sinal de entrada ter vindo de x em vez de p_g (a distribuição de imagens geradas por G). A função custo utilizada no jogo minimax é dada por

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))] \quad (1)$$

É importante notar que $\min\max$ é diferente de $\max\min$! Maximizar no discriminador e minimizar no gerador implica em um discriminador com o melhor desempenho atingível e um gerador que exerce seu potencial máximo de simular a distribuição original. A ordem contrária - minimizar no gerador e maximizar no discriminador - implica em um gerador péssimo e um discriminador otimizado para este gerador.

Na prática, o treinamento consiste em treinar o discriminador com um batch de dados reais, com labels '1', um batch de dados gerados, com labels '0'; e então treinar o conjunto "Gerador + Discriminador"(com os parâmetros do discriminador "congelados") com labels '1'. Assim o discriminador "aprende"progressivamente os dados reais e falsos, enquanto o gerador busca gerar dados que o discriminador reconheçam como reais. Nos casos treinados neste trabalho a perda (loss) utilizada tanto para discriminador como o gerador é a "binary crossentropy".

III. Problemas Recorrentes

Dada a natureza altamente dinâmica do sistema em questão, vários fatores e hiperparâmetros afetam a performance e o resultado final do treinamento. Inclusive, em várias situações, o treinamento não é garantido de atingir em um equilíbrio final, produzindo bons resultados em estágios intermediários do treinamento e colapsando para "lixo"nos momentos finais.

A. Instabilidade

É muito frequente que em vários momentos do treinamento alguma das perdas (do gerador ou discriminador) caia para 0 ou para um valor muito baixo. Mais frequentemente a perda do discriminador cai muito nos momentos iniciais do treinamento, pois o gerador neste momento gera apenas lixo, impedindo a propagação de gradientes para o treinamento do gerador. Como regra geral, quando a perda de qualquer uma das redes cai para valores muito baixos, a propagação de gradientes para o treinamento da outra rede fica altamente prejudicada.

O caso recíproco ocorre mais frequentemente em momentos intermediários do treinamento. Quando a perda do gerador

cai muito, isto indica que o gerador atingiu um modo capaz de ”enganar”o discriminador com muito sucesso. As imagens geradas neste caso são várias formas de ruído ou cores sólidas.

B. Mode Collapse

Outro caso indesejado é quando o gerador aprende a gerar apenas uma ”modo”, isto é, todas as imagens produzidas pelo gerador são iguais ou muito parecidas. Isto indica que o gerador identificou uma forma eficiente de reduzir sua loss, gerando apenas uma imagem, e assim aprimora apenas este modo.

C. Alta sensibilidade a parâmetros e hiperparâmetros

Pequenas variações em taxa de aprendizado, otimizador, topologia das redes, batch normalization, quantidade de filtros nas layers convolucionais, tamanho dos kernels, uso de dropouts no discriminador, etc., alteram significativamente o processo de treinamento e, consequentemente, os resultados obtidos. Das variáveis mencionadas, aquela que talvez seja a mais relevante é a taxa de aprendizado: a bibliografia em geral, recomenda taxas de aprendizado pequenas (da ordem de 0.0001) para que o treinamento seja suficientemente estável.

IV. Ferramentas e técnicas no treinamento de GANs

Devido a grande variedade de problemas encontrados, no constante processo de desenvolvimento das GANs várias técnicas foram desenvolvidas para cada um destes problemas.

A. Estabilidade

Dificultar o treinamento do discriminador, em geral, aprimora a estabilidade das GANs. Isto ocorre pois o discriminador aprende, muito antes do gerador aprender a criar imagens plausíveis, a separar as imagens falsas das verdadeiras. Diminuir a divergência (Kullback-Leibler) entre a distribuição de dados original e os dados gerados através da adição de ruído em ambas contribui para estabilizar o treinamento.

Utilizar batch normalization em todas as layers, exceto na última layer do gerador e na primeira do discriminador, auxilia na estabilidade do treinamento e na qualidade dos resultados obtidos.

B. Propagação de Gradientes

Problemas como *vanishing gradients* e *exploding gradients* são comuns em GANs. Gradientes fracos são evitados utilizando a ativação *leaky ReLU*, que permite a propagação de gradientes mesmo quando não está ”ativada”, e não utilizando *max pooling*.

C. Mode Collapse

Para evitar o *mode collapse*, diferentes topologias de redes foram propostas, e.g. utilizando *minibatch discrimination*, proposto em [2], ou alterando a função perda (e.g. Wasserstein GAN [3]). Técnicas utilizando ”espaço latente guiado”(*guided latent space*) existem, como proposto no artigo da Medium ”Reducing Mode Collapse in GANs using Guided Latent Spaces”, escrito por Parth Kashikar. A ideia neste caso é utilizar um classificador corretamente treinado no conjunto de dados e de alguma forma utilizá-lo no cálculo da perda e na geração de dados.

D. Artefatos Checkerboard

Como apresentado em [4], o uso de convoluções com strides não unitários para realização do *upsampling* no gerador e *downsampling* no discriminador pode causar artefatos ”checkerboard”, padrões xadrez repetitivos como exemplificados na Fig. 1.

O uso de layers ”Upsampling2D”para *upsampling* e layers de ”Average Pooling”para downsampling evita este tipo de artefato!

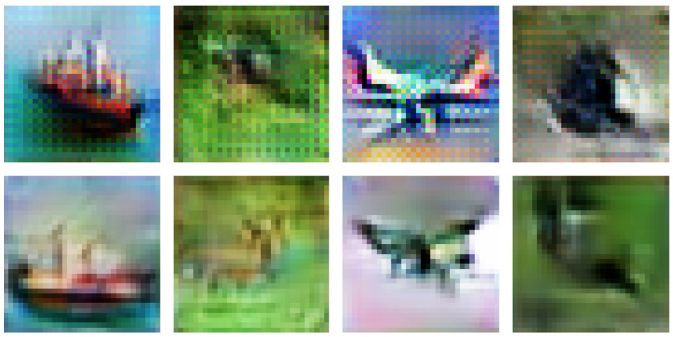

Figura 1. Exemplo de imagens com artefatos, na linha de cima, e imagens sem artefatos, linha de baixo.

E. Otimizador

A escolha do algoritmo otimizador e da taxa de aprendizado (o passo do otimizador) corretos são fundamentais para um bom treinamento. O uso de momento muito elevado pode instabilizar o treinamento devido a natureza altamente dinâmica dos modelos.

Taxas de aprendizado da ordem de 0.001 são, em geral, muito elevadas e produzem instabilidade no treinamento ou lixo nas imagens geradas. O valor proposto em [1] é de 0.0002 utilizando otimizador ADAM com beta de 0.5. Estas restrições são parcialmente devidas ao ruído estatístico nos gradientes. Otimizadores robustos a este cenário estão e estudo, com um exemplo deste tipo de algotitmo proposto em [5].

Um método de estabilizar o treinamento obter melhores resultados alterando parâmetros do otimizador consiste em escolher duas taxas de aprendizado independentes para o gerador e para o discriminador, conhecido como ”Two time update rule”e inicialmente proposto em [6]. Esta técnica também pode facilitar que o sistema atinja um equilíbrio de Nash não local, o que geralmente representa *mode collapse*.

F. Tamanho do Batch

A influência direta do tamanho do minibatch no treinamento de GANs ainda é uma questão em aberto e amplamente discutida. Existe um aparente *tradeoff* na escolha do tamanho do minibatch: menor minibatch permite mais iterações em menos tempo, entretanto produz gradientes mais ruidosos e,

além disso, pode prejudicar o resultado final produzido pelo gerador, praticamente invalidando o treinamento.

G. Experience Replay

Para evitar que o discriminador aprenda apenas o padrão das imagens falsas do gerador atual, treinar com imagens geradas por geradores antigos pode auxiliar o discriminador a identificar um padrão mais "independente do tempo". Isso evita que o discriminador esqueça (*forgetting*) modos anteriormente gerados.

H. Filtros das Camadas Convolucionais

Mais filtros e filtros maiores geralmente produzem resultados com melhor definição, entretanto, devido à quantidade maior de parâmetros, requer mais esforço computacional por iteração e pode até dificultar a propagação de gradientes. Filtros de *kernel size* igual a 5 apresentaram um bom compromisso neste aspecto para os modelos treinados.

I. Tamanho das imagens

O treinamento com imagens de dimensão maior ou igual a (128,128,3) se torna computacionalmente pesado e propenso a gerar resultados ruins ou subótimos. Um dos problemas é que as características *macro* da imagem estão mais distantes, necessitando de filtros maiores e mais filtros para captá-las. Apenas aumentar a complexidade da rede pode facilmente requerer elevado esforço computacional, muitas vezes inviável para "*consumer level hardware*". Uma das tentativas de sucesso para gerar imagens de alta definição a partir de GANs foi feita pelo time da NVIDIA com suas *Progressive Growing GANs* [7]. Neste caso a GAN "ganha"novas camadas durante o treinamento conforme a resolução das imagens geradas aumenta. Os exemplos obtidos no artigo são extremamente realistas, mas necessitaram de longo treinamento e hardware caro.

V. Redes Utilizadas

Neste trabalho, dois principais variações de GANs foram utilizados: DCGAN, proposta em [8], e FCC-GAN, proposta em [9].

A rede DCGAN utilizada é idêntica à FCC-GAN, exceto pelas camadas densas extra adicionadas. Durante os treinos, variações no tamanho dos filtros e número de filtros das camadas convolucionais.

Para adequar ao tamanho das imagens do conjunto de dados, (32,32,3) para o CIFAR-10 e (128,128,3) para as imagens de flores do dataset Linnaeus 5, a dimensão inicial do espaço latente foi adaptada, pois adicionar mais layers com *upsamling* não apresentou os resultados desejados.

VI. Discussão dos resultados

Analisando as imagens produzidas, como aquelas representadas nas Figs. 2 e 4, percebe-se que as topologias utilizadas capturam aspectos da estética geral do conjunto de dados. Frequentemente as imagens geradas falham em aspectos como perspectiva, problema exemplificado nas imagens de carro

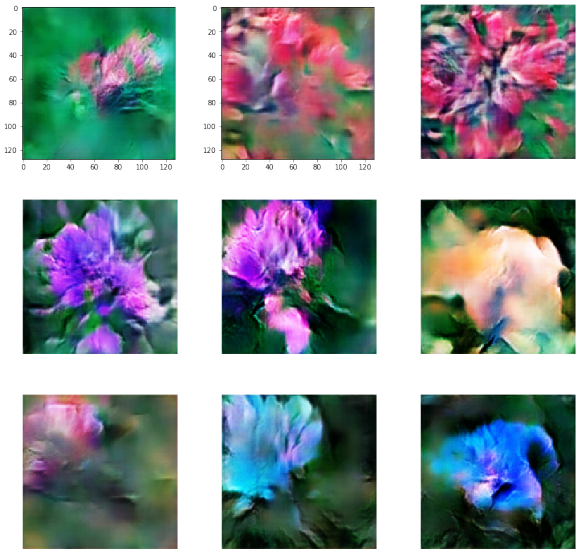

Figura 2. Imagens selecionadas dentre as geradas pela FCC-GAN treinada em um dataset de flores.

Figura 3. Imagens selecionadas do dataset original de flores (classe de flores do Linnaeus 5).

geradas pela GAN CIFAR-10, em que carros aparentam estar orientados em duas direções simulaneamente.

No caso das imagens de flores, as apresentadas na Fig. 2 foram selecionadas manualmente por apresentarem as características desejadas, entretanto muitas das imagens geradas possuem apenas texturas e características que são remanescentes do conjunto de dados.

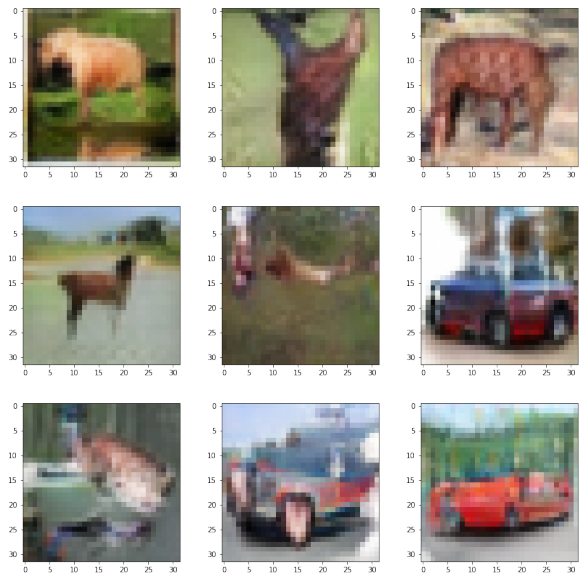

Figura 4. Imagens selecionadas dentre as geradas pela FCC-GAN treinada no conjunto CIFAR-10.

VII. Conclusão

Neste trabalho pode ser percebida a alta complexidade envolvida no treinamento e montagem de GANs. Uma grande variedade de problemas relacionados ainda existem devido ao tema ser uma proposição recente. Felizmente, as redes adversárias geradoras são alvo de intenso estudo, e novas técnicas, incluindo funções perda alternativas, *self-attention* e novos otimizadores e topologias de rede, estão permitindo o rápido desenvolvimento do ramo.

Espera-se que este trabalho tenha servido como uma útil revisão bibliográfica, apresentando um *overview* amplo e informativo. Os resultados obtidos são resultado de intensa experimentação e pesquisa, limitada pelos recursos computacionais oferecidos pela plataforma Google Colab. Várias topologias de redes foram testadas, entretanto o longo tempo de treinamento necessário, além do limite de uso do GPU do Colab, dificultam a velocidade de avaliação de diferentes técnicas e redes.

Referências

[1] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[2] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016.

[3] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017.

[4] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. [Online]. Available: http://distill.pub/2016/deconv-checkerboard

[5] T. Chavdarova, G. Gidel, F. Fleuret, and S. Lacoste-Julien, "Reducing noise in gan training with variance reduced extragradient," 2019.

[6] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," 2017.

[7] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," 2017.

[8] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015.

[9] S. Barua, S. M. Erfani, and J. Bailey, "Fcc-gan: A fully connected and convolutional net architecture for gans," 2019.