

КІЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

Кафедра обчислювальної математики

Кваліфікаційна робота

на здобуття ступеня бакалавра

за спеціальністю 113 Прикладна математика

на тему:

ЕКСТРАГРАДІЕНТНІ МЕТОДИ ДЛЯ НАВЧАННЯ GANs

Керівник дипломної роботи

професор

Семенов В. В.

«_____»_____ 2020р.

Виконав студент 4 курсу

Пушкін Д. Є.

«_____»_____ 2020р.

Київ

2020

Зміст

1 Вступ	2
2 Теорія GAN. Основні види GAN	2
2.1 Базові відомості про GAN	2
2.2 Non-saturating GAN	6
2.3 Wasserstein GAN (WGAN)	6
2.4 WGAN з штрафом на градієнт	10
3 Методи розв'язання варіаційних нерівностей в контексті GAN	12
3.1 Зведення тренування GAN до задачі варіаційної нерівності	12
3.2 Методи оптимізації варіаційних нерівностей	12
3.3 Приклад з іграшковим WGAN	15
3.4 Теореми про збіжність	20
4 Практичні результати	24
4.1 Метрика якості Inception Score	24
4.2 Conditional GAN (CGAN)	24
4.3 DCGAN	38
4.4 WGAN зі штрафом на градієнт	50
5 Висновки	62
6 Список використаної літератури	63

1 Вступ

Вперше модель GAN була описана зовсім нещодавно - у 2014 році у [1]. Ця модель дуже швидко стала популярною, адже не дивлячись на відсутність яких-небудь гарантій збіжності процесу тренування GAN, ця модель може показувати чудові результати у генеруванні різноманітних даних, зокрема генеруванні зображень. Головним недоліком базової моделі GAN є те, що її тренування вкрай нестабільне та потребує ретельного підбору гіперпараметрів та багатьох запусків, аби досягти бажаної якості. Багато наукових статей намагалися поліпшити стабільність тренування GAN. Основними напрямками досліджень були:

- 1) пошук вдалих архітектур для генератора та дискримінатора
- 2) знаходження найбільш вдалих функцій помилок
- 3) пошук найбільш вдалих методів для оптимізації GAN

У першому напрямку значного успіху досягнуто авторами статей [5] та [6]. У [5] автори описали сімейство архітектур, яке назвали Deep Convolutional GAN (DCGAN). За останні роки DCGAN зарекомендували себе з найкращого боку для майже всіх задач, які розв'язує GAN та використовуються у більшості моделей GAN у наш час. У [6] автори помітили, що GAN буде тренуватися значно швидше та стабільніше, коли він генерує однотипні дані (наприклад, лише одну цифру '0', а не усі цифри '0'-'9'). Для випадку, коли все ж потрібно генерувати об'єкти з різних класів, це мотивувало їх запропонувати модель умовного GAN. Ідея умовного GAN у тому, щоб передавати генератору та дискримінатору не тільки шум чи зображення, а і мітку класа, з якого треба згенерувати зображення або який відповідає переданому зображенню. У тих випадках, коли її можна застосувати, модель умовного GAN набагато підвищує швидкість та стабільність тренування GAN. Можна вважати, що задачу знаходження стабільних архітектур для GAN вирішено.

Щодо другого напрямку, у [2] автори теоретично обґрунтували, що серед найрозповсюдженіших відстаней між розподілами для функції помилки найкраще підходить відстань *Wasserstein-1*, або *Earth-Mover*. Це призвело до втнаходу моделі Wasserstein GAN і стало суттєвим проривом у теорії GAN. Втім, відстань Wasserstein-1 саму тому і рідко застосовується (принаймні, у глибинному навчанні), що її важко рахувати. Було запропоновано декілька методів це робити, на практиці у більшості задач найкращий результат дає спосіб, що використовується у WGAN зі штрафом на градієнт [4]. Але цьому способу бракує теоретичного обґрунтування коректності. Тому, хоча по другому напрямку відбулися суттєві просування, він ще не повністю закритий.

Основна мета даної роботи - дослідити третій напрямок, з яким виникає найбільше труднощів, адже досі для жодного оптимізаційного методу не доведено, що він збігається (хоча б при виконанні деяких достатньо загальних умов) при тренуванні GAN. Основна складність оптимізації полягає у тому, що, на відміну від звичайних нейронних мереж, де є лише одна функція помилки, у GAN функцій помилки дві - для генератора та для дискримінатора, і оптимізуючи одну з них, ми можемо погіршити якість по іншій. Тобто задача тренування GAN полягає не просто у мінімізації функції помилки, а у пошуку сідової точки для гри з двома гравцями. Наразі для тренування GAN найчастіше використовуються ті самі методи, що і для звичайних нейронних мереж - градієнтний спуск та його адаптивні варіації, такі, як Adam та RMSprop. У даній роботі показано, що ці методи є мало придатними для пошуку сідової точки та можуть не сходитись навіть у випадку простої білінійної гри (без кропітливого підбору параметрів у Adam чи RMSprop). Напомістъ, запропоновано використовувати адаптивні алгоритми, побудовані на базі екстраградієнтних методів. Ми побачимо, що при багатьох різних архітектурах GAN ці методи показують кращі результати, ніж звичайні градієнтні.

2 Теорія GAN. Основні види GAN

2.1 Базові відомості про GAN

GAN - породжуюча змагальна мережа, яка складається з двох мереж: генератора і дискримінатора. Генератор навчачеться генерувати дані із заданого розподілу p_d на просторі X . А саме, генератор в якості вхідних даних отримує дані z (які називають шумом) із деякого розподілу p_z на вже іншому просторі Z і переводить їх у дані з простору X , тим самим породжуючи новий розподіл p_g на X . Кінцева ціль GAN - зробити так, щоб згенерований розподіл p_g якнайточніше апроксимував справжній

розділ p_d . Формально генератор, що залежить від параметрів $\theta \in \mathbb{R}^{d_1}$, можна записати як

$$G = G_\theta(z) : \mathbb{R}^{d_1} \times Z \rightarrow X$$

У свою чергу, дискримінатор можна представити як наступну мережу, що залежить від параметрів $\varphi \in \mathbb{R}^{d_2}$:

$$D = D_\varphi(x) : \mathbb{R}^{d_2} \times X \rightarrow [0, 1]$$

тобто дискримінатор приймає на вхід дані з простору X та повертають число з $[0, 1]$, яке можна інтерпретувати як ймовірність того, що вхідні дані належали ап'ярному розподілу p_d , а не згенерованому p_g .

Ціль генератора - обманути дискримінатор, тобто зробити так, щоб дискримінатор не міг відрізняти згенеровані дані від справжніх. Ціль дискримінатора - навпаки, якнайточніше відрізняти справжні дані від згенерованих.

У оригінальній статті функцією помилки для дискримінатора вибирається бінарона крос-ентропія:

$$\mathcal{L}_D = -[\mathbb{E}_{x \sim p_d}(\log D(x)) + \mathbb{E}_{x \sim p_g} \log(1 - D(x))] \rightarrow \min$$

тобто дискримінатор намагається видавати максимальний результат на справжніх даних і мінімальний на згенерованих. Для генератора функція помилки вибирається наступна:

$$\mathcal{L}_G = \mathbb{E}_{x \sim p_g} \log(1 - D(x)) \rightarrow \min$$

Тобто генератор намагається мінімізувати вираз, який показує, наскільки дискримінатор "не довіряє" згенерованим даним.

Тренування GAN, яке полягає у одночасній оптимізації \mathcal{L}_D по параметрам φ та \mathcal{L}_G по параметрам θ , можна представити як гру з нульовою сумаю між генератором та дискримінатором, де цільовою функцією виступає $V(G, D) = [\mathbb{E}_{x \sim p_d}(\log D(x)) + \mathbb{E}_{x \sim p_g} \log(1 - D(x))]$. Дискримінатор намагається максимізувати $V(G, D)$, а генератор - мінімізувати.

Надалі для теоретичного обґрунтування нам необхідне припущення, що вибір моделей для генератора і дискримінатора не обмежений. Детальніше, зрозуміло, що генератор можна представити як функцію лише від параметрів θ , яка на вихід видає розподіл $G_\theta(z)$ за заданого розподілу $z \sim p_z$, тобто:

$$G = G(\theta) : \mathbb{R}^{d_1} \rightarrow \text{Prob}(X),$$

де $\text{Prob}(X)$ - множина всіх розподілів на просторі X . А дискримінатор можна представити як відображення у простір "ймовірносних класифікаторів" на X :

$$D = D(\varphi) : \mathbb{R}^{d_2} \rightarrow (X \xrightarrow{f} [0, 1])$$

Так ось, припустимо, що генератор G може породжувати усі розподіли з $\text{Prob}(X)$, а дискримінатор - усі "ймовірносні класифікатори" на X . Покажемо, що тоді при мінімаксій стратегії генератора G

$$\min_G \max_D V(G, D)$$

він буде породжувати в точності розподіл p_d , тобто $p_g = p_d$.

Лема 1

За заданого згенерованого розподілу $G(\theta) = p_g$ оптимальним значенням для дискримінатора буде

$$D^* = D(\varphi^*) = \frac{p_d}{p_d + p_g}.$$

Доведення

За заданого розподілу p_g , дискримінатору потрібно мінімізувати функцію

$$V(G, D) = \int p_d(x) \log(D(x)) + p_g(x) \log(1 - D(x)) dx$$

Для довільних $(a, b) \in \mathbb{R}_+^2 \setminus \{0, 0\}$ функція $y \rightarrow a \log(y) + b \log(1 - y)$ досягає максимума на $[0, 1]$ у точці $\frac{a}{a+b}$, що можна перевірити безпосереднім диференціюванням, що й завершує доведення.

Зauważення: зрозуміло, що нас цікавлять значення дискримінатора лише в $Supp(p_d) \cup Supp(p_g)$, в цих точках вираз $\frac{p_d}{p_d + p_g}$ визначений коректно.

Твердження 1

При своїй мінімаксній стратегії генератор G генерує розподіл $p_g = p_d$.

Доведення

Враховуючи лему 1, генератору залишається мінімізувати наступну величину:

$$\begin{aligned} \min_G V(G, D^*) &= \mathbb{E}_{x \sim p_d} [\log D^*(x)] + \mathbb{E}_{x \sim p_g} [\log(1 - D^*(x))] = \\ &= \mathbb{E}_{x \sim p_d} \left[\log \frac{p_d(x)}{p_d(x) + p_g(x)} \right] + \mathbb{E}_{x \sim p_g} \left[\log \frac{p_g(x)}{p_d(x) + p_g(x)} \right] = \\ &= -\log(4) + \mathbb{E}_{x \sim p_d} \left[\log \frac{p_d(x)}{\left(\frac{p_d(x) + p_g(x)}{2} \right)} \right] + \mathbb{E}_{x \sim p_g} \left[\log \frac{p_g(x)}{\left(\frac{p_d(x) + p_g(x)}{2} \right)} \right] = \\ &= -\log(4) + KL(p_d \parallel \frac{p_d + p_g}{2}) + KL(p_g \parallel \frac{p_d + p_g}{2}) = -\log(4) + 2JS(p_d \parallel p_g) \end{aligned}$$

Мінімум останнього виразу досягається лише при $JS(p_d \parallel p_g) = 0$. Це рівносильно $p_d = p_g$, що й завершує доведення.

Також, враховуючи лему 1 і твердження 1, робимо висновок, що ця гра має рівноважну стратегію за Нешем, що досягається лише при $G(\theta^*) = p_d$, $D(\varphi^*) = f : f(x) = \frac{1}{2} \forall x \in Supp(p_d) \cup Supp(p_g)$. Тому тренування GAN зводиться до знаходження точки рівноваги Неша (θ^*, φ^*) у вищеописаній грі. Зазнаємо, що рівновага Неша в даному випадку є нестрогою: хоча дискримінатору строго невигідно відхилятися від своєї стратегії, генератору може перейти в будь-яку іншу стратегію без програнку для себе.

У пошуку рівноважних стратегій може допомогти наступна теорема.

Теорема 1

Розглянемо $V(G, D) = U(p_g, D)$ як функцію від p_g . Тоді наступний алгоритм

$$\begin{cases} D^{(t+1)} = \arg \max_D U(p_g^{(t)}, D) \\ p_g^{(t+1)} = p_g^{(t)} - \eta_{t+1} \nabla_{p_g} U(p_g^{(t)}, D^{(t+1)}) \end{cases} \quad (1)$$

збігається до точки рівноваги за Нешем (p_g^*, D^*) при довільному початковому наближенні $(p_g^{(0)}, D^{(0)})$ і достатньо малих параметрах η_t .

Доведення

Помітимо, що $U(p_g, D)$ лінійна по p_g . Скористаємося фактом, що субградієнт супремуму опуклих функцій містить субградієнт функції, для якої досягається максимум у цій точці. Тобто якщо $f(x) = \sup_{\alpha \in A} f_\alpha(x)$, $f(x)$ опуклі по x та $\beta \in \text{argmax}_{\alpha \in A} f_\alpha(x)$, то $\partial f_\beta(x) \in \partial f(x)$. На кожному кроці алгоритму робиться градієнтний крок для p_g при оптимальному D для заданого p_g . Функція $\sup_D U(p_g, D)$ опукла як супремум лінійних, а значить опуклих функцій, та має єдину оптимальну точку, як доведено у твердженні 1. Тобто можна вважати, що на кожній ітерації алгоритму робиться субградієнтний крок для мінімізації $\sup_D U(p_g, D)$. Тому для достатньо малих параметрів η_t , p_g зайдеться до p_d .

Наслідок

Якщо стратегієми генератора вважати розподіли, що він генерує, то гра є лінійною по стратегіям

генератора.

Зauważення

Для доведення важливо, що оновлення дискримінатору і генератора відбуваються почергово (для генератор використовується вже нове оптимальне значення дискримінатора). Пізніше ми побачимо, що при пошуку точки рівноваги за Нешем у багатьох випадках почергове оновлення змінних сходиться там, де при одночасному оновленні алгоритм не сходиться.

Теорема 1 стверджує, що при почерговому оновленні дискримінатора до оптимального значення при заданому значенні генератора та оновленні генератора градієнтним кроком, алгоритм сходиться до точки рівноваги за Нешем. Втім, на практиці результат цієї теореми застосувати не можна. По-перше, нейронні мережі, що представляють генератор і дисримінатор, можуть генерувати лише обмежену сім'ю розподілів p_g та обмежену сім'ю "ймовірносних класифікаторів" відповідно. По-друге, на практиці нам не відомо $\nabla_{p_g} U(p_g, D)$, ми можемо знайти лише стохастичну оцінку для $\nabla_\theta V(G(\varphi), D(\theta))$, тобто на практиці ми оптимізуємо параметри нейронної мережі θ , а не одразу розподіли p_g . По-третє, ми не можемо дозволити собі на кожному кроці оновлювати дисримінатор до оптимуму.

Тим не менш, на базі теореми 1 у [1] пропонується наступний алгоритм тренування GAN.

Параметри: T - число тренувальних кроків, K - кількість оновлень дисримінатора на одне оновлення генератора, η_t - learning rate.

Алгоритм тренування GAN

для кількості тренувальних кроків $t = \overline{1, T}$ робити:

для $k = \overline{1, K}$ робити:

згенерувати мінібатч із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

згенерувати мінібатч із m даних $\{x^{(1)}, \dots, x^{(m)}\}$ із розподілу справжніх даних p_x .

обчислити стохастичний градієнт для параметрів дисримінатора:

$$g_\varphi := \nabla_\varphi \left(\frac{1}{m} \sum_{i=1}^m [\log(D(x^{(i)})) + \log(D(G(z^{(i)})))] \right)$$

новити параметри дисримінатора:

$$\varphi := \varphi + \eta_t \cdot g_\varphi$$

згенерувати minibatch із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

обчислити стохастичний градієнт для параметрів генератора:

$$g_\theta := \nabla_\theta \left(\frac{1}{m} \sum_{i=1}^m [\log(1 - D(G(z^{(i)})))] \right)$$

новити параметри генератора:

$$\theta := \theta - \eta_t \cdot g_\theta$$

На практиці в більшості випадків тренування дисримінатора частіше за генератор не дає суттєвих переваг, тому використовують найпростіший варіант $K = 1$.

Наочанок зазначимо одну особливість алгоритму тренування GAN. З твердження 1 випливає, що при оптимальному дисримінаторі генератор буде намагатись мінімізувати дивіргенцію Єнсена-Шеннона $JS(p_d || p_g)$ між розподілами p_g і p_d . Більшість же інших алгоритмів генерування використовують гіпотезу максимальної правдоподібності:

$$\max_\theta \frac{1}{m} \sum_{i=1}^m \log P_\theta(x_i) \rightarrow \max$$

яка асимптотично еквівалентна мінімізації дивіргенції Кульбака-Лейблера $KL(p_d || p_g)$.

2.2 Non-saturating GAN

Вибір функції помилки для генератора $\mathcal{L}(G) = \mathbb{E}_{x \sim p_g} \log(1 - D(x)) \rightarrow \min$ є не дуже вдалим, бо він призводить до затухання градієнтів, коли дискримінатор впевнено відрізняє справжні дані від згенерованих ($D(x) \approx 0$), що зазвичай відбувається на початку тренування і сильно впливає на швидкість збіжності. Тому на практиці використовують іншу:

$$\mathcal{L}'(G) = -\mathbb{E}_{x \sim p_g} \log(D(x)) \rightarrow \min$$

яка є монотонним перетворенням минулої функції помилки, але вирішує проблему затухання градієнтів на початку тренування. Наприклад, якщо активацією останнього рівня генератора є сигмоїд, тобто $D(x) = \sigma(t)$, де t - вихід останнього рівня до активації, то $D(x) \approx 0 \Leftrightarrow t \approx -\infty$, і при підрахунку градієнів по всім змінним присутнє множення на градієнт останнього рівня $\frac{\partial \mathcal{L}(G)}{\partial t}$. Спростимо вираз, що диференціюється:

$$\log(1 - D(x)) = \log(1 - \sigma(t)) = \log\left(1 - \frac{1}{1 + e^{-t}}\right) = \log\left(\frac{e^{-t}}{1 + e^{-t}}\right) = -t - \log(1 + e^{-t})$$

тому можемо записати

$$\frac{\partial \mathcal{L}(G)}{\partial t} = \frac{\partial \mathbb{E}_{x \sim p_g} (-t - \log(1 + e^{-t}))}{\partial t} = \mathbb{E}_{x \sim p_g} \frac{\partial (-t - \log(1 + e^{-t}))}{\partial t} = \mathbb{E}_{x \sim p_g} - \frac{1}{1 + e^{-t}} \rightarrow 0, t \rightarrow -\infty$$

У випадку функції помилки $\mathcal{L}'(G)$ аналогічно отримуємо

$$\log D(x) = -\log(1 + e^{-t})$$

$$\frac{\partial \mathcal{L}'(G)}{\partial t} = \mathbb{E}_{x \sim p_g} \left(1 - \frac{1}{1 + e^{-t}}\right) \rightarrow 1, t \rightarrow -\infty$$

Бачимо, що у першому випадку градієнти домножуються на значення, близьке до 0, що і призводить до затухання градієнтів на рінніх стадіях. При другій же функції помилки градієнти домножуються на значення, близькі до 1.

При новій функції помилки гра між генератором і дискримінатором набуває наступного вигляду:

$$\begin{cases} \mathcal{L}(D_\varphi) = [\mathbb{E}_{x \sim p_d} (\log D_\varphi(x)) + \mathbb{E}_{z \sim p_z} \log(1 - D_\varphi(G_\theta(z)))] \rightarrow \min_\varphi \\ \mathcal{L}'(G_\theta) = -\mathbb{E}_{z \sim p_z} \log(D_\varphi(G_\theta(z))) \rightarrow \min_\theta \end{cases}$$

Така постановка задачі називається *non-saturating GAN*. Оскільки перетворення функції помилки для генератора було монотонне, то ця гра має ту саму точку рівноваги за Нешем, що і звичайна GAN. Але тепер гра має ненульову суму. Як наслідок, теорема 1 для non-saturating GAN не виконується. Проте алгоритм тренування GAN, описаний у попередньому пункті, для non-saturating GAN працює краще, ніж для звичайної GAN.

2.3 Wasserstein GAN (WGAN)

Теорія з цього підрозділу взята з [2]. Якщо не вказано інше, там же можна знайти доведення цих теорем. У цій роботі доведення опущенні, бо цей розділ - не головна мета даної роботи.

Почнемо з теорії. Розглянемо основні відстані між розподілами.

Визначення.

1. *Earth-Mover (EM)*, або *Wasserstein-1* відстанню між розподілами \mathbb{P}_r та \mathbb{P}_θ називається

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_\theta)} \mathbb{E}_{(x, y) \sim \gamma} \|x - y\|$$

де $\Pi(\mathbb{P}_r, \mathbb{P}_\theta)$ - множина усіх сумісних розподілів, у яких перша величина має розподіл \mathbb{P}_r , друга - \mathbb{P}_θ .

2. *Дивіргенцією Кульбака-Лейблера* між розподілами \mathbb{P}_r та \mathbb{P}_θ називається

$$KL(\mathbb{P}_r, \mathbb{P}_\theta) = \int \mathbb{P}_r(x) \log \left(\frac{\mathbb{P}_r(x)}{\mathbb{P}_\theta(x)} \right) d\mu(x)$$

3. *Дивіргенцією Єнсена-Шеннона* між розподілами \mathbb{P}_r та \mathbb{P}_θ називається

$$JS(\mathbb{P}_r, \mathbb{P}_\theta) = \frac{1}{2}(KL(\mathbb{P}_r, \mathbb{P}_m) + KL(\mathbb{P}_\theta, \mathbb{P}_m)),$$

де $\mathbb{P}_m = (\mathbb{P}_r + \mathbb{P}_\theta)/2$.

4. *Повною варіацією* між розподілами \mathbb{P}_r та \mathbb{P}_θ називається

$$\delta(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{A \in \Sigma} |\mathbb{P}_r(A) - \mathbb{P}_\theta(A)|,$$

де Σ - множина борелівських множин простору X .

Надалі у деяких теоремах буде фігурувати припущення, що X - компактний простір. Зазначимо, що це припущення виконеться у випадку, який нас цікавить найбільше, коли X - простір зображень заданого розміру.

Теорема

Нехай \mathbb{P} - розподіл на компактному просторі X , $\{\mathbb{P}_n\}_{n \in \mathbb{N}}$ - послідовність розподілів на цьому ж просторі. Тоді, розглядаючи всі ліміти при $n \rightarrow \infty$:

1. Наступні твердження еквівалентні:

$$\delta(\mathbb{P}_n, \mathbb{P}) \rightarrow 0$$

$$JS(\mathbb{P}_n, \mathbb{P}) \rightarrow 0$$

2. Наступні твердження еквівалентні:

$$W(\mathbb{P}_n, \mathbb{P}) \rightarrow 0$$

$$\mathbb{P}_n \xrightarrow{\mathcal{D}} \mathbb{P},$$

де \mathcal{D} позначає збіжність випадкової величини за розподілом.

3. З $KL(\mathbb{P}_n, \mathbb{P}) \rightarrow 0$ або $KL(\mathbb{P}, \mathbb{P}_n) \rightarrow 0$ випливає (1)

4. З (1) випливає (2).

Ця теорема доводить, що топологія, породжена відстанню W , є слабшою за топології, породженими іншими відстанями. Проілюструємо це на прикладі.

Приклад

Нехай $Z \sim U[0, 1]$ - рівномірний розподіл на одиничному інтервалі. Позначимо через \mathbb{P}_0 розподіл $(0, Z) \in \mathbb{R}^2$, а через \mathbb{P}_θ розподіл $(\theta, Z) \in \mathbb{R}^2$, де $\theta \in \mathbb{R}$ - параметр. Тоді неважко переконатись, що:

$$W(\mathbb{P}_0, \mathbb{P}_\theta) = |\theta|$$

$$JS(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} \log(2), & \theta \neq 0 \\ 0, & \theta = 0 \end{cases}$$

$$KL(\mathbb{P}_0, \mathbb{P}_\theta) = KL(\mathbb{P}_\theta, \mathbb{P}_0) = \begin{cases} +\infty, & \theta \neq 0 \\ 0, & \theta = 0 \end{cases}$$

$$\delta(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} 1, & \theta \neq 0 \\ 0, & \theta = 0 \end{cases}$$

Бачимо, що коли $\theta \rightarrow 0$, то $\mathbb{P}_\theta \rightarrow 0$ в розумінні ЕМ - відстанні, але у розумінні JS, KL, оберненого KL та δ - ні.

Цей приклад ілюструє, що у випадку, коли треба апроксимувати розподіл, носій якого - многовид меншої розмірності, то це можна зробити градієнтним спуском лише по відстані ЕМ, для всіх інших розглянутих відстаней градієнти будуть нульові, допоки носій згенерованого і заданого розподілу не будуть перетинатись по множині ненульової міри.

Як ми вже зазначали, у звичайній GAN функція помилки генератора пропорційна відстані Єнсена-Шеннона між розподілами (у припущеннях, що дискримінатор приймає оптимальне значення). Тепер ми бачимо, що це не дуже вдалий вибір відстані. Ідея *Wasserstein GAN* - апроксимувати функцію помилки генератора відстанню Wasserstein-1. Наступна теорема строго обґрутує переваги використання відстані Wasserstein-1 над іншими.

Теорема

Нехай \mathbb{P}_r - фіксована функція розподілу на компактному просторі X , z - випадкова величина на іншому просторі Z . Нехай $g_\theta(z) : Z \times \mathbb{R}^d \rightarrow X$ - функція, виражена нейронною мережею, а через \mathbb{P}_θ будемо позначати функцію розподілу випадкової величини $g_\theta(z)$. Тоді:

- 1) функція $W(\mathbb{P}_r, \mathbb{P}_\theta)$ неперервна по θ .
- 2) якщо всі активації нейронної мережі $g_\theta(z)$ є диференційовними, то $W(\mathbb{P}_r, \mathbb{P}_\theta)$ диференційовна по θ майже скрізь.
- 3) твердження 1) та 2) не виконуються для дивіргенцій Кульбака-Лейблера $KL(\mathbb{P}_r, \mathbb{P}_\theta)$ та Єнсена-Шеннона $JS(\mathbb{P}_r, \mathbb{P}_\theta)$.

Доведення

1) Нехай $\theta, \theta' \in \mathbb{R}^d$ - довільні параметри. Розглянемо сумісний розподіл $\gamma = (g_\theta(Z), g_{\theta'}(Z))$. Зрозуміло, що $\gamma \in \Pi(\mathbb{P}_\theta, \mathbb{P}_{\theta'})$. Тому

$$W(\mathbb{P}_\theta, \mathbb{P}_{\theta'}) \leq \mathbb{E}_{(x,y) \sim \gamma} \|x - y\| = \mathbb{E}_z \|g_\theta(z) - g_{\theta'}(z)\|$$

З неперервності g по θ маємо $g_{\theta'}(z) \rightarrow g_\theta(z)$ при $\theta' \rightarrow \theta$, тобто функції $g_{\theta'}$ прямують до g_θ поточково. Оскільки X - компактний простір, то відстань між будь-якими його елементами рівномірно обмежена деякою константою M . Зокрема, $\|g_\theta(z) - g_{\theta'}(z)\| \leq M$. Тоді за теоремою Лебега про мажорантну збіжність

$$W(\mathbb{P}_\theta, \mathbb{P}_{\theta'}) \leq \mathbb{E}_z \|g_\theta(z) - g_{\theta'}(z)\| \rightarrow 0, \theta' \rightarrow \theta$$

Звідси

$$|W(\mathbb{P}_r, \mathbb{P}_\theta) - W(\mathbb{P}_r, \mathbb{P}_{\theta'})| \leq W(\mathbb{P}_\theta, \mathbb{P}_{\theta'}) \rightarrow 0, \theta' \rightarrow \theta$$

що й доводить неперервність $W(\mathbb{P}_r, \mathbb{P}_\theta)$.

Втім, інфімум з означення Wasserstein-1 відстані є вкрай непіддаливий. Для того, щоб апроксимувати цю відстань, нам знадобиться наступна теорема.

Теорема Канторовича - Рубінштейна

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r} [f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta} [f(x)]$$

де $\|f\|_L \leq 1$ позначає множину всіх 1-Ліпшицевих функцій $f : X \rightarrow \mathbb{R}$.

Доведення дивись у [3].

Помітимо, що якщо супремум брати по усім K -ліпшицевим функціям $\|f\|_L \leq K$, то супремум набуде значення $K \cdot W(\mathbb{P}_r, \mathbb{P}_\theta)$. У [2] запропонували шукати супремум на просторі функцій, що задаються певною нейронною мережею, у якій параметри усіх лінійних рівнів по модулю обмежені деякою константою C та активації є ліпшицевими функціями (що виконується для всіх основних активацій).

Зрозуміло, що така нейронна мережа представляє собою Ліпшицеву функцію, у якій константа Ліпшиця K залежить лише від числа C , активації нейронної мережі та її архітектури (точніше, кількості нейлонів на кожному рівні). Ця нейронна мережа і буде дискримінатором для WGAN. Ми сподіваємося, що за рахунку широкого вибору параметрів дискримінатора, знайдеться такий із них, при якому дискримінатор буде достатньо точно апроксимувти оптимальну функцію f на просторі $\|f\|_L \leq K$. Тому тепер функцію помилки генератора за заданого дискримінатора $D(\varphi)$ з параметрами $\varphi \in \Phi$, будемо рахувати як:

$$\max_{\varphi \in \Phi} [\mathbb{E}_{x \sim \mathbb{P}_r} D_\varphi(x) - \mathbb{E}_{z \sim p_z(z)} D_\varphi(G_\theta(z))]$$

і будемо робити градієнтні кроки, диференціюючи цю величину, що апроксимує $W(\mathbb{P}_r, \mathbb{P}_\theta)$, по θ . Цей метод обґрунтовує наступна теорема.

Теорема

Нехай \mathcal{P}_r - деякий розподіл на X . Через p_θ позначатимемо розподіл $g_\theta(z)$, де z - випадкова величина з розподілом p_z та g_θ - нейронна мережа з параметрами θ і диференційовними активаціями. Тоді існує функція $f : X \rightarrow \mathbb{R}$, на якій досягається максимум виразу:

$$\max_{\|f\|_L \leq 1} [\mathbb{E}_{p \sim \mathbb{P}_r} f(x) - \mathbb{E}_{p \sim \mathbb{P}_\theta} f(x)]$$

та виконується:

$$\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z \sim p_z(z)} [\nabla_\theta f(g_\theta(z))]$$

коли обидві частини рівності визначені.

Нагадаємо, що однією з причин, чому теорема 1.1 була незастосовна на практиці. У випадку звичайної GAN робити градієнтний крок по функції помилки як функції від розподілів - не те саме, що робити градієнтний крок функції помилки як від функції від параметрів генератора. Важливість останньої теореми полягає у тому, що вона стверджує, що у випадку, коли функція помилки генератора дорівнює Wasserstein-1 відстані між розподілами, це те саме.

Залишається питання, як оптимізувати дискримінатор з параметрами, на які накладено обмеження: $\varphi \in \Phi = \{\varphi : \varphi_i \leq C \quad \forall i\}$. У статті [2] пропонувалось просте обрізання всіх параметрів φ_i на відрізок $[-C, C]$ через простоту та уже непоганий результат. Зрозуміло, що можна на кожній ітерації робити проектування на гіперкуб, або сферу, змінивши при цьому допустиму область Φ .

Алгоритм тренування WGAN

для кількості тренувальних кроків $t = \overline{1, T}$ робити:

для $k = \overline{1, K}$ робити:

згенерувати мінібатч із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

згенерувати мінібатч із m даних $\{x^{(1)}, \dots, x^{(m)}\}$ із розподілу справжніх даних p_x .

обчислити стохастичний градієнт для параметрів дискримінатора:

$$g_\varphi := \nabla_\varphi \left(\frac{1}{m} \sum_{i=1}^m [D_\varphi(x^d) - D_\varphi(G_\theta(z^{(i)}))] \right)$$

новити параметри дискримінатора:

$$\varphi := \varphi + \eta_t \cdot g_\varphi$$

спроектувати або обрізати параметри на допустиму область

$$\varphi = P_\Phi(\varphi) \quad or \quad \varphi = clip(\varphi, \Phi)$$

згенерувати minibatch із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

обчислити стохастичний градієнт для параметрів генератора:

$$g_\theta := -\nabla_\varphi \left(\frac{1}{m} \sum_{i=1}^m D_\varphi(G_\theta(z^{(i)})) \right)$$

новити параметри генератора:

$$\theta := \theta - \eta_t \cdot g_\theta$$

У [2] пропонується взяти $K = 5$.

2.4 WGAN з штрафом на градієнт

У [4] було помічено, що на практиці при простому обрізанні параметрів на відрізок $[-C, C]$ дуже значна частина параметрів під час навчання прийматимуть граничні значення на відрізку, що погано впливає на виразність дискримінатора. Альтернативний спосіб обмежети ліпшицеву норму дискримінатора спирається на наступне спостереження.

Твердження

Нехай \mathbb{P}_r та \mathbb{P}_θ - два розподіли на компактному метричному просторі X , f^* - розв'язок $\max_{\|f\|_L \leq 1} [\mathbb{E}_{p \sim \mathbb{P}_r} f(x) - \mathbb{E}_{p \sim \mathbb{P}_\theta} f(x)]$, γ^* - сумісний розподіл, що є розв'язком $W(\mathbb{P}_r, \mathbb{P}_\theta) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_\theta)} \mathbb{E}_{(x,y) \sim \gamma} \|x - y\|$. Позначимо $x_t = tx + (1-t)y$, $0 \leq t \leq 1$. Тоді, якщо $\gamma^*(x = y) = 0$ та f^* - диференційовна, то

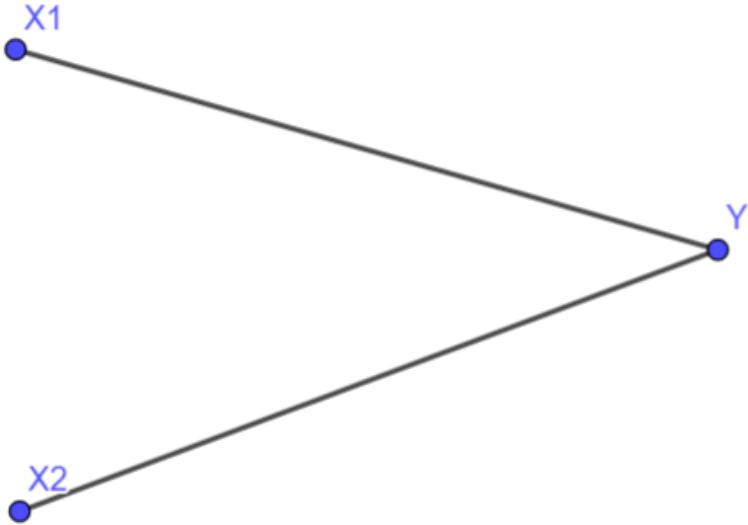
$$\mathbb{P}_{(x,y) \sim \gamma^*} \left[\nabla f^*(x_t) = \frac{y - x_t}{\|y - x_t\|} \right] = 1$$

Наслідок: підставляючи у теорему $t = 0$ та $t = 1$, отримаємо, що $\|\nabla f(x)\| = 1$ майже напевне на $\text{Supp}(\mathbb{P}_r \cup \mathbb{P}_\theta)$ - об'єднанні носіїв розподілів \mathbb{P}_r і \mathbb{P}_θ .

Як відомо, для того, щоб диференційовна функція була 1-ліпшицевою, необхідно і достатньо щоб її градієнт у кожній точці по нормі не перевищував 1. Більш того, з теореми випливає, що для функції f^* її градієнт у точках $x_t = tx + (1-t)y$, $0 \leq t \leq 1$ по нормі дорівнює 1 майже напевне. Ідея WGAN зі штрафом на градієнт полягає у тому, щоб замість того, щоб вводити обмеження на параметри дискримінатора, змусити градієнти у відповідних точках приймати приймати значення, рівні 1 по модулю. Для цього функція помилка для дискримінатора змінюється на наступну:

$$\mathcal{L}_D(\varphi) = \max_\varphi [\mathbb{E}_{x \sim \mathbb{P}_r} D_\varphi(x) - \mathbb{E}_{z \sim p_z(z)} D_\varphi(G_\theta(z))] + \lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} [(\|\nabla_{\hat{x}} D(\hat{x})\| - 1)^2]$$

де перший доданок відповідає на Wasserstein-1 відстань між розподілами - штраф на градієнт - змушує дискримінатор бути 1-ліпшицевою функцією. У формулі ми неявно визначили розподіл випадкової величини $\hat{x} = tx + (1-t)y$, $x \in \mathbb{P}_\theta$, $y \in \mathbb{P}_r$, $t = U[0, 1]$ - рівномірно розподілена на $[0, 1]$ випадкова величина. Для генератора функція помилки залишається тією самою. Зрозуміло, що коли носій розподілу \mathbb{P}_r чи \mathbb{P}_θ не опуклий, то навіть строгое виконання умови $\|\nabla f(\hat{x})\| = 1$ для всіх $\hat{x} = tx + (1-t)y$ гарантує 1-ліпшицевості дискримінатора D . Дійсно, нехай $\text{Supp}(\mathbb{P}_r = \{y\})$, $\text{Supp}(\mathbb{P}_\theta = \{x_1, x_2\})$, а дискримінатор на точках ламаної x_1yx_2 приймає значення, що рівне відстані від данної точки до точки x_1 вздовж цієї ламаної.



в такому випадку похідна дискримінатора для всіх точок ламаної рівна 1, але умова 1-ліпшицевості не виконується, наприклад, для точок X_1 і X_2 . Тим не менш, за відсутності теоретичних обґрунтувань, на практиці метод WGAN зі штрафом на градієнт заракоментдував себе дуже добре. Значною перевагою цього підходу є те, що він не потребує ніяких обмежень на параметри, і за рахунок чого дискримінатор має більшу виразність, ніж у звичайному WGAN.

Алгоритм тренування WGAN зі штрафом на градієнт

для кількості тренувальних кроків $t = \overline{1, T}$ робити:

для $k = \overline{1, K}$ робити:

згенерувати мінібатч із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

згенерувати мінібатч із m даних $\{x^{(1)}, \dots, x^{(m)}\}$ із розподілу справжніх даних p_x .

згенерувати мінібатч із m випадкових величин $\{\varepsilon^{(1)}, \dots, \varepsilon^{(m)}\}$ із $U[0,1]$.

порахувати згенеровані дані $\tilde{x}^{(i)} := G_\theta(z^{(i)})$

$\hat{x}^{(i)} := \varepsilon^{(i)}\tilde{x}^{(i)} + (1 - \varepsilon^{(i)})x^{(i)}$

порахувати помилку на i -тій парі даних

$$L^{(i)} := D(x^{(i)}) - D(\tilde{x}^{(i)}) + \lambda(\|\nabla_{\hat{x}} D_\varphi(\hat{x})\| - 1)^2$$

обчислити стохастичний градієнт для параметрів дискримінатора:

$$g_\varphi := \nabla_\varphi \left(\frac{1}{m} \sum_{i=1}^m L^{(i)} \right)$$

новити параметри дискримінатора:

$$\varphi := \varphi + \eta_t \cdot g_\varphi$$

згенерувати minibatch із m даних $\{z^{(1)}, \dots, z^{(m)}\}$ із розподілу шумів p_z .

обчислити стохастичний градієнт для параметрів генератора:

$$g_\theta := -\nabla_\varphi \left(\frac{1}{m} \sum_{i=1}^m D_\varphi(G_\theta(z^{(i)})) \right)$$

новити параметри генератора:

$$\theta := \theta - \eta_t \cdot g_\theta$$

3 Методи розв'язання варіаційних нерівностей в контексті GAN

3.1 Зведення тренування GAN до задачі варіаційної нерівності

Задачу тренування GAN з дискримінатором $D = D(\varphi)$ та генератором $G = G(\theta)$ можна представити як одночасну мінімізацію функції похибки дискримінатора $\mathcal{L}_D(\varphi, \theta)$ по параметрам φ та функції похибки генератора $\mathcal{L}_G(\varphi, \theta)$ по параметрам θ . Точніше, нам потрібно знайти точку рівноваги за Нешем (φ^*, θ^*) , тобто точку (φ^*, θ^*) , для якої виконується:

$$\begin{cases} \varphi^* = \arg \min_{\varphi} \mathcal{L}_D(\varphi, \theta^*) \\ \theta^* = \arg \min_{\theta} \mathcal{L}_G(\varphi^*, \theta) \end{cases}$$

Розглянемо випадок, коли функції помилки \mathcal{L}_D і \mathcal{L}_G - гладкі (тобто всі активації генератора і дискримінатора неперервно диференційовні). Тоді для того, щоб точка (φ^*, θ^*) була розв'язком системи, необхідно, щоб вона була стаціонарною точкою. У випадку, коли на параметри φ та θ не накладено жодних обмежень, умова стаціонарності записується наступним чином

$$\|\nabla_{\varphi} \mathcal{L}_D(\varphi^*, \theta^*)\| = \|\nabla_{\theta} \mathcal{L}_G(\varphi^*, \theta^*)\| = 0$$

У більш загальному випадку, коли на параметри накладено обмеження $\varphi \in \Phi$, $\theta \in \Theta$, при цьому множини Φ та Θ - опуклі, стаціонарна точка - визначається як та, для якої часткові похідні по всім допустимим напрямкам невід'ємні:

$$\nabla_{\varphi} \mathcal{L}_D(\varphi^*, \theta^*)^T (\varphi - \varphi^*) \geq 0, \quad \nabla_{\theta} \mathcal{L}_G(\varphi^*, \theta^*)^T (\theta - \theta^*) \geq 0 \quad \forall (\varphi, \theta) \in \Phi \times \Theta$$

Визначимо $w = (\varphi, \theta)$, $w^* = (\varphi^*, \theta^*)$, $\Omega = \Phi \times \Theta$. Тоді умову стаціонарності можна коротко записати як

$$F(w^*)^T (w - w^*) \geq 0 \quad \forall w \in \Omega$$

Тут $F(w) = [(\nabla_{\varphi} \mathcal{L}_D(\varphi, \theta))^T, (\nabla_{\theta} \mathcal{L}_G(\varphi, \theta))^T]^T$, та мається на увазі, що нерівність ≥ 0 виконується покоординатно.

Задачею варіаційної нерівності (VIP) будемо називати наступну:

$$\text{знайти } w^* \in \Omega: F(w^*)^T (w - w^*) \geq 0 \quad \forall w \in \Omega$$

Множину $VIP^* \subset \Omega$ точок, які задовольняють варіаційній нерівності, будемо називати *оптимальною множиною*.

3.2 Методи оптимізації варіаційних нерівностей

Розглянемо основні оптимізаційні методи, які використовуються для варіаційних нерівностей, а також їх адаптивний варіант Adam, який гарно себе проявив для задачі тренування нейронних мереж.

1. Метод градієнтного спуску

$$w_{t+1} = P_{\Omega}(w_t - \eta_t F(w_t)),$$

$w_0 \in \Omega$ - довільне.

В якості наближень можна розглядати не значення параметрів на останній ітерації, а усереднене значення параметрів:

$$\bar{w}_T = \sum_{t=0}^{T-1} \rho_t w_t$$

Тут $\{\rho_t\}_{t=0}^{T-1}$ - вагові множники, $\rho_t \geq 0$, $\sum_{t=0}^{T-1} \rho_t = 1$.

Такий метод називається **метод градієнтного спуску з усередненням**.

На практиці найрозважливішими виборами вагових коефіцієнтів є рівномірне та експоненційне. Рівномірне задається ваговими коефіцієнтами $\rho_0 = \rho_1 = \dots = \rho_{T-1} = \frac{1}{T}$ та його можна рахувати ітераційно за формулою $\bar{w}_T = \frac{1}{T} w_{T-1} + \frac{T-1}{T} \bar{w}_{T-1}$. Експоненційне середнє рахується як $\bar{w}_T = \beta w_{T-1} + (1 - \beta) \bar{w}_{T-1}$, де $0 < \beta < 1$ - параметр.

Описимо адаптивний варіант цього методу Adam.

Adam

параметри: $0 < \beta_1, \beta_2 < 1, \varepsilon > 0$ (для запобігання ділення на 0), $\eta > 0$ - швидкість навчання.
 початок: вибираємо довільне $w_0, m_0 := 0, v_0 := 0$.
 на ітерації t робимо:

$$\begin{aligned} g_t &:= F(w_{t-1}) \\ m_t &:= \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (\text{зміщена еспоненційно усереднена оцінка градієта}) \\ v_t &:= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (\text{зміщена еспоненційно усереднена оцінка квадрата градієта}) \\ \hat{m}_t &:= m_t / (1 - \beta_1^t) \quad (\text{незміщена еспоненційно усереднена оцінка градієта}) \\ \hat{v}_t &:= v_t / (1 - \beta_2^t) \quad (\text{незміщена еспоненційно усереднена оцінка квадрата градієта}) \\ w_t &:= P_\Omega(w_{t-1} - \eta \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)) \end{aligned}$$

2. Екстраградієнтний метод (Метод Корпелевич)

Іншим способом розв'язання VIP є екстраградієнтний метод Корпелевич. Ідея методу полягає у тому, щоб рахувати значення на наступній ітерації через градієнт у (екстраполяційній) точці, що відрізняється від точки на попередній ітерації:

$$\begin{cases} w_{t+1/2} = P_\Omega(w_t - \eta F(w_t)) \\ w_{t+1} = P_\Omega(w_t - \eta F(w_{t+1/2})) \end{cases}$$

Екстраградієнтний метод можна сприймати як наближення неявної методу Ейлера:

$$\begin{cases} w_{t+1} = w_t - P_\Omega(w_t - \eta F(w_{t+1})) \end{cases}$$

який теоретично є більш стійким та має кращі показники збіжності, ніж явний метод, який співпадає зі звичайним градієнтним методом. На практиці неявний метод Ейлера в застосовувати проблематично, бо в загальному випадку він вимагає на кожному кроці розв'язувати нелінійне рівняння. Тому виник екстраградієнтний метод як апроксимація цього методу.

Опишемо метод Adam, побудований на основі екстраградієнтного методу (параметри ті самі, що і у звичайному Adam).

extra-Adam

початок: вибираємо довільне $w_0, m_0 := 0, v_0 := 0$.

на ітерації t робимо:

$$\begin{aligned} g_{t-1/2} &:= F(w_{t-1}) \\ m_{t-1/2} &:= \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1/2} \\ v_{t-1/2} &:= \beta_2 v_{t-1} + (1 - \beta_2) g_{t-1/2}^2 \\ \hat{m}_{t-1/2} &:= m_{t-1/2} / (1 - \beta_1^{2t-1}) \\ \hat{v}_{t-1/2} &:= v_{t-1/2} / (1 - \beta_2^{2t-1}) \\ w_{t-1/2} &:= P_\Omega(w_{t-1} - \eta \cdot \hat{m}_{t-1/2} / (\sqrt{\hat{v}_{t-1/2}} + \varepsilon)) \\ g_t &:= F(w_{t-1/2}) \\ m_t &:= \beta_1 m_{t-1/2} + (1 - \beta_1) g_t \\ v_t &:= \beta_2 v_{t-1/2} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &:= m_t / (1 - \beta_1^{2t}) \\ \hat{v}_t &:= v_t / (1 - \beta_2^{2t}) \\ w_t &:= P_\Omega(w_{t-1} - \eta \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)) \end{aligned}$$

3. Метод Tseng'a

Цей метод задається формулами:

$$\begin{cases} w_{t+1/2} = P_\Omega(w_t - \eta F(w_t)) \\ w_{t+1} = w_{t+1/2} - \eta (F(w_{t+1/2}) - F(w_t)) \end{cases}$$

Помітимо, що при відсутності обмежень метод Tseng'a співпадає з екстраградієнтним методом Корпелевич. Дійсно, в цьому випадку:

$$\begin{cases} w_{t+1/2} = w_t - \eta F(w_t) \\ w_{t+1} = w_{t+1/2} - \eta(F(w_{t+1/2}) - F(w_t)) = (w_{t+1/2} + \eta F(w_t)) - \eta F(w_{t+1/2}) = w_t - \eta F(w_{t+1/2}) \end{cases}$$

Перейдемо до Adam, побудованого на основі Tseng'a. Він буде відрізнятись від extra-Adam лише останньою формулою.

Tseng Adam

початок: вибираємо довільне w_0 , $m_0 := 0$, $v_0 := 0$.

на ітерації t робимо:

$$\begin{aligned} g_{t-1/2} &:= F(w_{t-1}) \\ m_{t-1/2} &:= \beta_1 m_{t-1} + (1 - \beta_1) g_{t-1/2} \\ v_{t-1/2} &:= \beta_2 v_{t-1} + (1 - \beta_2) g_{t-1/2}^2 \\ \hat{m}_{t-1/2} &:= m_{t-1/2} / (1 - \beta_1^{2t-1}) \\ \hat{v}_{t-1/2} &:= v_{t-1/2} / (1 - \beta_2^{2t-1}) \\ w_{t-1/2} &:= P_\Omega(w_{t-1} - \eta \cdot \hat{m}_{t-1/2} / (\sqrt{\hat{v}_{t-1/2}} + \varepsilon)) \\ g_t &:= F(w_{t-1/2}) \\ m_t &:= \beta_1 m_{t-1/2} + (1 - \beta_1) g_t \\ v_t &:= \beta_2 v_{t-1/2} + (1 - \beta_2) g_t^2 \\ \hat{m}_t &:= m_t / (1 - \beta_1^{2t}) \\ \hat{v}_t &:= v_t / (1 - \beta_2^{2t}) \\ w_t &:= P_\Omega[w_{t-1/2} - \eta \cdot (\hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon) - \hat{m}_{t-1/2} / (\sqrt{\hat{v}_{t-1/2}} + \varepsilon))] \end{aligned}$$

4. Екстраградієнтний метод з минулого (Метод Попова)

Головним недоліком екстраградієнтного методу Корпелевич та методу Tseng'a є те, що вони потребують для одної ітерації рахувати градієнт у двох точках. Метод Попова виник як спроба вирішити цю проблему, при цьому не втративши показники збіжності екстраградієнтного методу. Ідея *екстраградієнтного методу з минулого* полягає у тому, щоб замість градієнта $\nabla F(w_t)$ використовувати його наближення $\nabla F(w_{t-1/2})$, знайдене на минулій ітерації.

$$\begin{cases} w_{t+1/2} = P_\Omega(w_t - \eta F(w_{t-1/2})) \\ w_{t+1} = P_\Omega(w_t - \eta F(w_{t+1/2})) \end{cases}$$

Щодо адаптивного варіанту, він задається наступним чином

Adam Попова

початок: вибираємо довільне w_0 , $m_{-1/2} := 0$, $v_{-1/2} := 0$.

на ітерації t робимо:

$$\begin{aligned} w_{t-1/2} &:= P_\Omega(w_{t-1} - \eta \cdot \hat{m}_{t-3/2} / (\sqrt{\hat{v}_{t-3/2}} + \varepsilon)) \\ g_{t-1/2} &:= F(w_{t-1/2}) \\ m_{t-1/2} &:= \beta_1 m_{t-3/2} + (1 - \beta_1) g_{t-1/2} \\ v_{t-1/2} &:= \beta_2 v_{t-3/2} + (1 - \beta_2) g_{t-1/2}^2 \\ \hat{m}_{t-1/2} &:= m_{t-1/2} / (1 - \beta_1^t) \\ \hat{v}_{t-1/2} &:= v_{t-1/2} / (1 - \beta_2^t) \\ w_t &:= P_\Omega[w_{t-1} - \eta \cdot \hat{m}_{t-1/2} / (\sqrt{\hat{v}_{t-1/2}} + \varepsilon)] \end{aligned}$$

5. Оптимістичний зеркальний спуск (OMD)

Ідея цього методу - використовувати градієнт з минулої ітерації для "уточнення" напрямку для наступної ітерації. Він задається як:

$$w_{t+1} = w_t - 2\eta F(w_t) + \eta F(w_{t-1})$$

Насправді, цей метод тісно пов'язаний з методом Попова, коли на параметри не накладено обмеження. У цьому випадку перші рівняння у формулі для t -ої і $(t-1)$ -ої ітерації методу Попова мають вигляд:

$$w_{t+1/2} = w_t - \eta F(w_{t-1/2})$$

$$w_{t-1/2} = w_{t-1} - \eta F(w_{t-3/2})$$

віднімемо їх:

$$w_{t+1/2} - w_{t-1/2} = (w_t - w_{t-1}) - \eta F(w_{t-1/2}) + \eta F(w_{t-3/2})$$

Використовуючи $w_t = w_{t-1} - \eta F(w_{t-1/2})$, отримаємо:

$$w_{t+1/2} = w_{t-1/2} - 2\eta F(w_{t-1/2}) + \eta F(w_{t-3/2})$$

Бачимо, що у випадку відсутності обмежень методи Попова і OMD співпадають з точністю до зсунення номеру ітерації на $1/2$.

Втім, на відміну від методу Попова, для OMD невідомо, чи буде він сходитись у випадку опуклих обмежень, якщо на кожному кроці проектувати результат на множину обмежень. Тому цей алгоритм навмисне написаний без проектування.

оптимістичний Adam

початок: вибираємо довільне w_0 , $m_{-1/2} := 0$, $v_{-1/2} := 0$.

на ітерації t робимо:

$$\begin{aligned} g_t &:= F(w_t) \\ m_t &:= \beta_1 m_{t-1} + (1 - \beta_1)g_t \\ v_t &:= \beta_2 v_{t-1} + (1 - \beta_2)g_t^2 \\ \hat{m}_t &:= m_t / (1 - \beta_1^{t+1}) \\ \hat{v}_t &:= v_t / (1 - \beta_2^{t+1}) \\ w_{t+1} &:= w_t - 2\eta \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon) + \eta \cdot \hat{m}_{t-1} / (\sqrt{\hat{v}_{t-1}} + \varepsilon) \end{aligned}$$

3.3 Приклад з іграшковим WGAN

Для оптимізації нейронних мереж, у тому числі і GAN, найчастіше використовують методи, побудовані на базі звичайного градієнтного методу. Щоб проілюструвати, що цей метод мало придатний до трнування GAN, розглянемо наступний іграшковий приклад.

Нехай потрібно апроксимувати розподіл випадкової величини x . Генератор $G = G(\theta)$ приймає приймає на вхід випадкову величину z і видає θz . Дискримінатор заданий функцією $D_\varphi(x) = \varphi x$ (ми розглядаємо саме WGAN). Маємо наступну гру:

$$\min_{\theta} \max_{\varphi} \varphi \mathbb{E}[x] - \varphi \theta \mathbb{E}[z]$$

Для максимального спрощення цільової функції припустимо, що $\mathbb{E}[x] = 0$, $\mathbb{E}[z] = -1$. Тоді маємо задачу

$$\min_{\theta \in \mathbb{R}} \max_{\varphi \in \mathbb{R}} \theta \varphi$$

з сідовою точкою $(\theta^*, \varphi^*) = (0, 0)$. Цій грі відповідає VIP з оператором $F(\theta, \varphi) = (\varphi, -\theta)$. Розглянемо дві варіації градієнтного методу: з одночасним та почерговим оновленням змінних (для простоти візьмемо константну швидкість навчання η):

З одночасних оновленням:

$$\begin{cases} \theta_{t+1} = \theta_t - \eta \varphi_t \\ \varphi_{t+1} = \varphi_t + \eta \theta_t \end{cases} \quad (2)$$

З почерговим оновленням:

$$\begin{cases} \theta_{t+1} = \theta_t - \eta \varphi_t \\ \varphi_{t+1} = \varphi_t + \eta \theta_{t+1} \end{cases} \quad (3)$$

Виявляється, ці два варіанти мають абсолютно різну поведінку у цій задачі.

Твердження 1

При одночасному оновленні норма похибки розсодиться зі швидкістю геометричної прогресії:

$$\theta_t^2 + \varphi_t^2 = (1 + \eta^2)^t (\theta_0^2 + \varphi_0^2)$$

При почерговому оновленні послідовність (θ_t, φ_t) є обмеженою, але не сходиться до $(0,0)$:

$$\theta_t^2 + \varphi_t^2 = \Theta(\theta_0^2 + \varphi_0^2),$$

де $u_t = \Theta(v_t) \Leftrightarrow \exists \alpha, \beta > 0 : \alpha v_t \leq u_t \leq \beta v_t$.

Якщо ж брати рівномірне середнє параметрів $(\bar{\theta}_t, \bar{\varphi}_t) = \frac{1}{t} \sum_{i=0}^{t-1} (\theta_i, \varphi_i)$, то при одночасному оновленні змінних $(\bar{\theta}_t, \bar{\varphi}_t)$ розходиться:

$$\bar{\theta}_t^2 + \bar{\varphi}_t^2 = \Theta \left(\frac{\theta_0^2 + \varphi_0^2}{\eta^2 t^2} (1 + \eta^2)^t \right)$$

а при почерговому оновленні сходиться зі швидкістю $O(1/t^2)$:

$$\bar{\theta}_t^2 + \bar{\varphi}_t^2 = O \left(\frac{\theta_0^2 + \varphi_0^2}{\eta^2 t^2} \right)$$

Доведення

Для одночасного оновлення:

$\|(\theta_{t+1}, \varphi_{t+1})\|^2 = \theta_{t+1}^2 + \varphi_{t+1}^2 = (\theta_t - \eta \varphi_t)^2 + (\varphi_t + \eta \theta_t)^2 = (1 + \eta^2)(\varphi_t^2 + \theta_t^2) = (1 + \eta^2)\|(\theta_t, \varphi_t)\|^2$, тому

$$\|(\theta_t, \varphi_t)\|^2 = (1 + \eta^2)^t \|(\theta_t, \varphi_t)\|^2$$

Тепер перепишему формулу для ітерації у вигляді:

$$\begin{cases} \eta \varphi_t = \theta_t - \theta_{t+1} \\ \eta \theta_t = \varphi_{t+1} - \varphi_t \end{cases} \quad (4)$$

просумуємо по $t = \overline{0, T-1}$, отримаємо: $\bar{\varphi}_T = \frac{\theta_0 - \theta_T}{\eta T}$, $\bar{\theta}_T = \frac{\varphi_T - \varphi_0}{\eta T}$, тоді

$$\bar{\theta}_T^2 + \bar{\varphi}_T^2 = \frac{(\theta_0 - \theta_T)^2 + (\varphi_0 - \varphi_T)^2}{\eta^2 T^2} =$$

$$= \frac{((1 + \eta^2)^T + 1)(\theta_0^2 + \varphi_0^2) - 2\theta_0\theta_T - 2\varphi_0\varphi_T}{\eta^2 T^2} =$$

$$= \Theta \left(\frac{(1 + \eta^2)^T (\theta_0^2 + \varphi_0^2)}{\eta^2 T^2} \right)$$

Для почергового оновлення перепишемо ітерацію у вигляді:

$$\begin{cases} \theta_{t+1} = \theta_t - \eta \varphi_t \\ \varphi_{t+1} = \varphi_t + \eta \theta_{t+1} = \varphi_t - \eta(\theta_t - \eta \varphi_t) \end{cases} \quad (5)$$

Або у векторному вигляді:

$$\begin{bmatrix} \theta_{t+1} \\ \varphi_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & -\eta \\ \eta & 1 - \eta^2 \end{bmatrix} \begin{bmatrix} \theta_t \\ \varphi_t \end{bmatrix} \quad (6)$$

Матриця $M = \begin{bmatrix} 1 & -\eta \\ \eta & 1 - \eta^2 \end{bmatrix}$ при $\eta < 2$ має 2 комплексно спряжених власних значення:

$$\lambda_{\pm} = 1 - \eta \frac{\eta \pm i\sqrt{4 - \eta^2}}{2}$$

і квадрати їх модулів рівні $\det(M) = 1$. Ми можемо діагоналізувати матрицю M , тобто існує така невироджена матриця P , що $M = P^{-1} \text{diag}(\lambda_+, \lambda_-)P$. Тоді

$$\begin{bmatrix} \theta_t \\ \varphi_t \end{bmatrix} = M^t \begin{bmatrix} \eta_0 \\ \varphi_0 \end{bmatrix} = P^{-1} \text{diag}(\lambda_+^t, \lambda_-^t) P \begin{bmatrix} \eta_0 \\ \varphi_0 \end{bmatrix} \quad (7)$$

відповідно,

$$\theta_t^2 + \varphi_t^2 = \left\| \begin{bmatrix} \theta_t \\ \varphi_t \end{bmatrix} \right\|^2 = \left\| P^{-1} \text{diag}(\lambda_+^t, \lambda_-^t) P \begin{bmatrix} \eta_0 \\ \varphi_0 \end{bmatrix} \right\|^2 \leq \|P^{-1}\|^2 \|P\|^2 (\theta_0^2 + \varphi_0^2) \quad (8)$$

аналогічно

$$\theta_0^2 + \varphi_0^2 = \left\| M^{-t} \begin{bmatrix} \eta_t \\ \varphi_t \end{bmatrix} \right\|^2 = \left\| P \text{diag}(\lambda_+^{-t}, \lambda_-^{-t}) P^{-1} \begin{bmatrix} \eta_t \\ \varphi_t \end{bmatrix} \right\|^2 \leq \|P\|^2 \|P^{-1}\|^2 (\theta_t^2 + \varphi_t^2) \quad (9)$$

звідси якщо $\theta_0^2 + \varphi_0^2 > 0$, то послідовність (θ_t, φ_t) обмежена, але не сходиться до 0.

Тепер випадок усереднених параметрів $(\bar{\theta}_t, \bar{\varphi}_t)$. Перепишемо формулу для ітерації у вигляді

$$\begin{cases} \eta \varphi_t = \theta_t - \theta_{t+1} \\ \eta \theta_t = \varphi_t - \varphi_{t-1} \end{cases} \Rightarrow \begin{cases} \bar{\varphi}_T = \frac{\theta_0 - \theta_T}{\eta T} \\ \bar{\theta}_T = \frac{\varphi_{T-1} - \varphi_0 - \eta \theta_0}{\eta T} \end{cases} \quad (10)$$

Тому, оскільки $\theta_t^2 + \varphi_t^2 = \Theta(\varphi_0^2 + \theta_0^2)$,

$$\bar{\varphi}_t^2 + \bar{\theta}_t^2 = O\left(\frac{\varphi_0^2 + \theta_0^2}{\eta^2 t^2}\right)$$

Тепер розглянемо, як на цьому ж прикладі поводить себе екстраградієнтний метод Корполевича (тут розглядаємо лише "гірший випадок" одночасній оновлень. Неважко перевірити, що і при одночасних, і при почергових оновленнях швидкість збіжності буде лінійна: для одночасних $O((1 - \eta^2 + \eta^4)^t)$, для почергових - $O(1/(1 + \eta^2)^t) = O((1 - \eta^2 + \eta^4 - O(\eta^6))^t)$. Виграш почергових оновлень над одночасними ϵ , але незначний).

Твердження 2

При екстраградієнтному методі Корпелевича квадрат норми ітерацій $N_t = \theta_t^2 + \varphi_t^2$ лінійно сходиться до 0 при $0 < \eta < 1$:

$$N_{t+1} = (1 - \eta^2 + \eta^4) N_t$$

Доведення

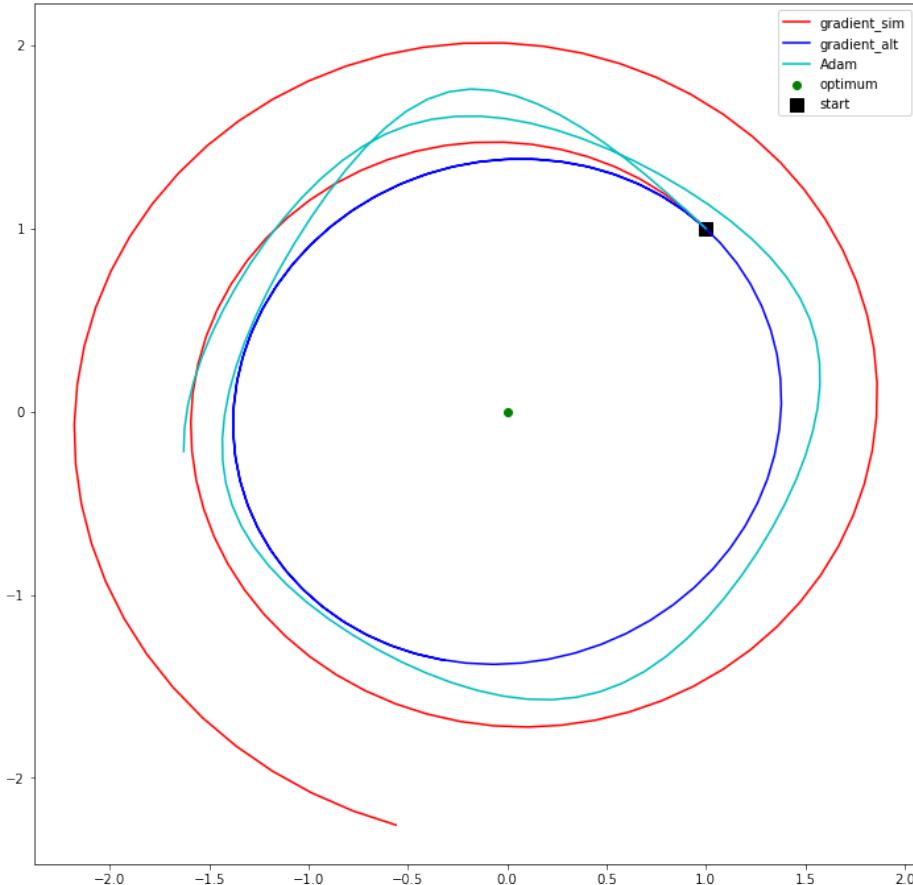
Перепишемо формулу ітерації у вигляді:

$$\begin{cases} \theta_{t+1} = \theta_t - \eta(\varphi_t + \eta \theta_t) = (1 - \eta^2)\theta_t - \eta \varphi_t \\ \varphi_{t+1} = \varphi_t + \eta(\theta_t - \eta \varphi_t) = (1 - \eta^2)\varphi_t + \eta \theta_t \end{cases}$$

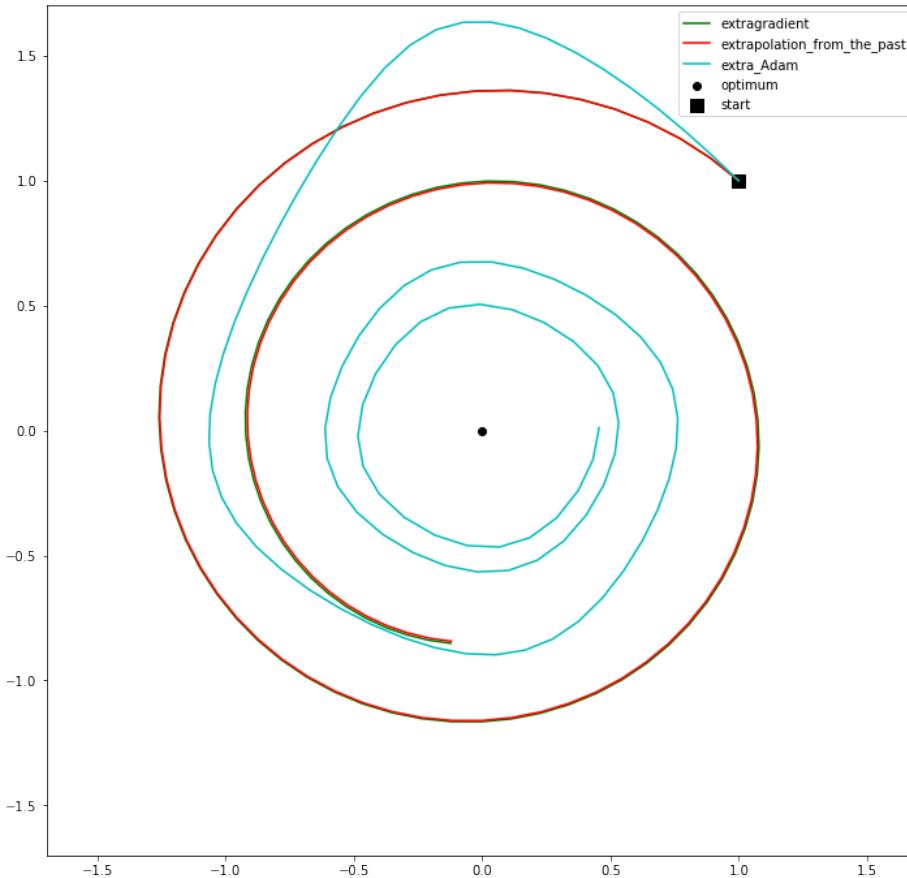
$$\text{Тоді } \theta_{t+1}^2 + \varphi_{t+1}^2 = ((1 - \eta^2)\theta_t - \eta\varphi_t)^2 + ((1 - \eta^2)\varphi_t + \eta\theta_t)^2 = (1 - \eta^2 + \eta^4)(\theta_t^2 + \varphi_t^2)$$

Бачимо, що на відміну від звичайного градієнтного методу, екстраградієнтний збігається до оптимальної точки навіть без усереднення, причому лінійно.

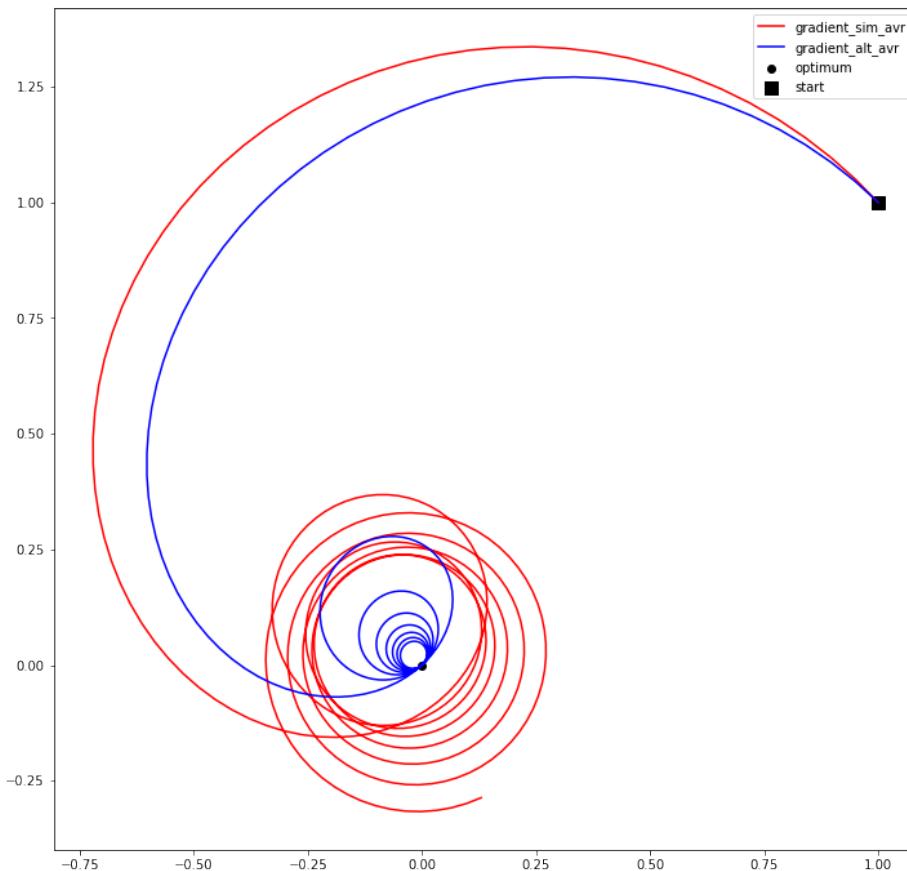
Зобразимо, як на цьому прикладі працюють описані нами (і не тільки) методи.



Розглядається звичайний градієнтний метод та його адаптивний варіант Adam. gradient sim відповідає за градієнтний метод з одночасним оновленням, gradient alt - з почерговим. Використовувався Adam (з одночасними оновленнями) з параметрами $\beta_1 = 0.5$, $\beta_2 = 0.9$. Бачимо, що жоден із цих методів не сходитьсья (вірогідно, можна зробити так, щоб Adam сходився, але після кропітливого підбору параметрів). Для всіх методів використовувалась швидкість навчання $\eta = 0.1$ і робилося 100 ітераційних кроків.

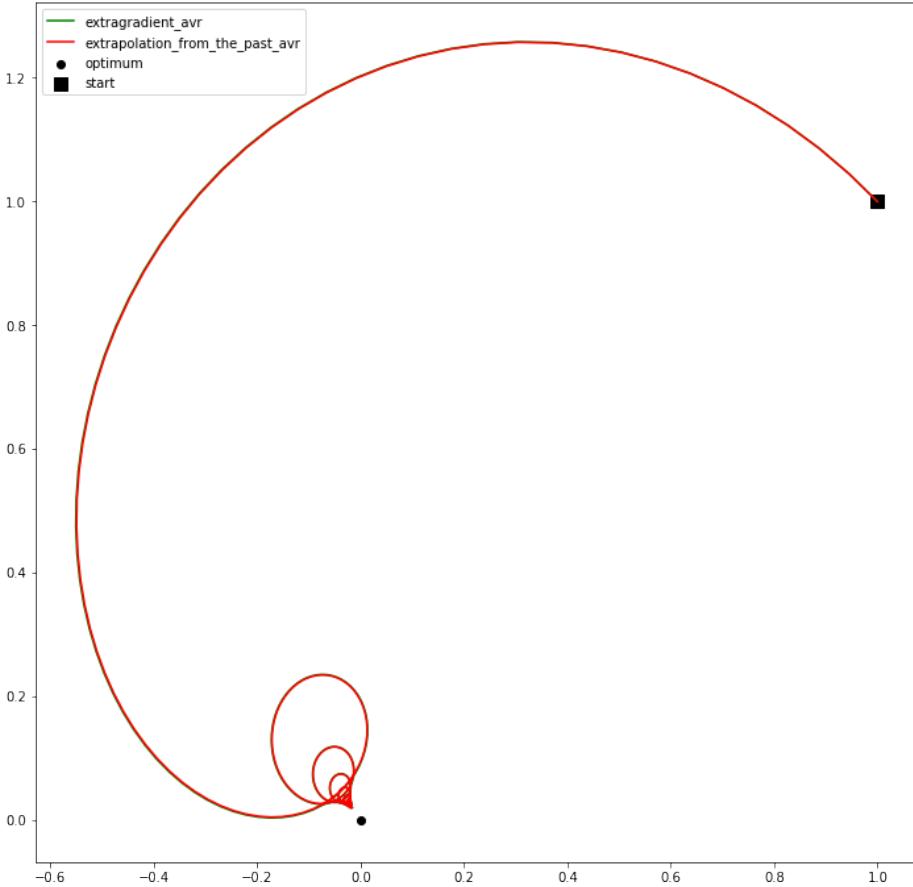


Тут розглядається екстраградієнтний та естраградієнтний метод з минулого, а також адаптивний варіант екстраградієнного методу extra Adam (параметри і кількість кроків ті самі). Ми не реалізували метод Tseng'a та метод оптимістичного зеркального спуску, бо при відсутності обмежень вони (майже) співпадають з екстраградієнтним та екстраградієнтним з минулого методами відповідно. Бачимо, що для цього прикладу екстраградієнтний та екстраградієнтний з минулого методи майже співпадають і повільно сходяться до оптимуму. Метод extra Adam сходиться швидше за всі інші методи.



Тут gradient sim avr відповідає за усереднений градієнтний метод з одночасним оновленням. Бачимо,

що як ми і довели, його ітерації приймають обмежені значення, але не сходяться. Gradient alt avr відповідає за усереднений градієнтний метод з почерговими оновленнями. Як і було доведено, він сходиться до оптимуму, але довго продовжує коливатись. Тут ми зробили по 500 для обох методів.



Тут extragradient avr та extrapolation from the past відповідають за усереднені екстраградієнтний та екстраградієнтний з минулого методи. Кількість ітерацій та сама - 500. Бачимо, що ці методи знову співпали і прямують до оптимуму швидше, ніж усереднений градієнтний з почерговим оновленням.

Отож, навіть для такої простої білінійної гри, використовуючи звичайний градієнтний метод, ми спромоглися зйтися до оптимуму тільки використовуючи усереднення та роблячи оновлення почергово. Втім, для спрощених нейронних мереж використовувати почергове оновлення - неприпустимо довго по часу. Може бути корисним оновлювати параметри дискримінатора та генератора почергово, але це зовсім не те саме, що спочатку оновлювати всі параметри дискримінатора почергово, потім - всі параметри дискримінатора почергово. Єдина надія, що ми зможемо зйтися - покластися на адаптивні методи, типу Adam, і сподіватись, що вони зайдуться за рахунок того, що "нечесно" рахують градієнт. Але ми бачили, що на цьому прикладі екстраполяційний Adam працював значно краще, ніж звичайний. Цей приклад підтверджує, що екстраполяційні методи працюють краще для задач пошуку сідової точки у грі між двома гравцями. В подальшому для справжніх GANів ми будемо використовувати лише адаптивні методи, побудовані на базі вище розглянутих методів.

3.4 Теореми про збіжність

У минулій секції ми на конкретному прикладі побачили, що для білінійної гри між двома гравцями звичайний градієнтний метод не сходився до розв'язку, а всі інші розглянуті нами сходились. Якщо ж розглядати більш широкий скла ігор - опукло-ввігнуті, який відповідає монотонному та ліпшицевому оператору L , то відомо, що екстраградієнтний метод Корполевича, метод Попова та метод Tseng'a слабко сходяться до розв'язку (у випадку скінченно-вимірного простору слабка збіжність співпадає із сильною). Відповідно, метод оптимістичного спуску також збігається до розв'язку, якщо немає обмежень на параметри, бо він співпадає з методом Попова з точністю до зсуву ітерацій на $1/2$. Доведення можна знайти, наприклад, у [7]. Якщо ж оператор F задовільняє більш сильні умови: μ -сильна монотонність та ліпшицевість (гра є строго опукло-ввігнутою), то то варіаційна нерівність

має єдиний розв'язок, та методи градієнтного та екстраградієнтного спуску при правильному підборі параметрів сходяться до нього з лінійно швидкістю (доведення дивись нижче).

Втім, всі ці ситуації абсолютно не застосовна до навчання GAN, бо, у загальному випадку, гра зовсім не буде схожою на опукло-ввігнуту.

Теорема (градієнтний метод для ліпшицевих сильно опуклих операторів)

Якщо F - μ -сильно монотонний та L -Ліпшицевий оператор, множина обмежень Ω - замкнена, опукла, то градієнтний метод зі швидкістю навчання $0 < \lambda < \mu L^{-2}$ сходиться до розв'язку VIP з лінійною швидкістю.

Лема

Якщо при виконанні алгоритму отримали $x_{n+1} = x_n$, то $x_n \in VIP^*$.

Доведення

З рівності $x_{n+1} = P_\Omega(x_n - \lambda F(x_n))$ випливає

$$(x_{n+1} - x_n + \lambda F(x_n), x - x_{n+1}) \geq 0 \quad \forall x \in \Omega$$

Враховуючи $x_{n+1} = x_n$, останню рівність можна переписати як

$$(F(x_n), x - x_n) \geq 0 \quad \forall x \in \Omega$$

що і означає, що $x_n \in \Omega$.

Доведення теореми

Введемо оператор $Tw = P_\Omega(w - \lambda F(w)), 0 < \lambda < \mu L^{-2}$.

Що діє з Ω в Ω . Покажимо, що він є стискаючим при $0 < \lambda < \mu L^{-2}$.

Маємо:

$$\begin{aligned} \|Tx - Ty\|^2 &= \|P_\Omega(x - \lambda F(x)) - P_\Omega(y - \lambda F(y))\|^2 \leq \|x - \lambda F(x) - y + \lambda F(y)\|^2 = \\ &\|x - y\|^2 + \lambda^2 \|Fx - Fy\|^2 - 2\lambda(Fx - Fy, x - y) \leq (1 + \lambda^2 L^2 - 2\mu\lambda) \|x - y\|^2 \end{aligned}$$

тобто

$$\|Tx - Ty\| \leq q(\lambda) \|x - y\|,$$

де $(\lambda) = \sqrt{1 + \lambda^2 L^2 - 2\mu\lambda}$. При $0 < \lambda < \mu L^{-2}$ виконується $0 < q(\lambda) < 1$. Це означає, що оператор T - стискаючий.

Зазначимо, що замкнена множина Ω є повним метричним простором з метрикою $d(x, y) = \|x - y\|$. Алгоритм, записаний у вигляді $x_{n+1} = T(x_n)$, є класичним прикладом пошуку нерухомої точки z оператору T , тобто $x_n \rightarrow z$. З леми 1 випливає, що точка z буде розв'язком варіаційної нерівності. Більш того, маємо

$$\|x_n - x_k\| \leq q(\lambda)^n \|x_0 - x_{k-n}\| \quad \forall k \geq n$$

Спрямувавши k до нескінченності, отримаємо

$$\|x_n - z\| \leq q(\lambda)^n \|x_0 - z\| \quad \forall n$$

Загуваження:

Найменше значення $q(\lambda)$ досягається при $\lambda_0 = \mu L^{-2}$ і дорівнює $q(\lambda_0) = \sqrt{1 - \mu^2 L^{-2}}$.

Теорема (екстраградієнтний метод для ліпшицевих сильно опуклих операторів)

Якщо F - μ -сильно монотонний та L -Ліпшицевий оператор, то екстраградієнтний з минулого метод з параметром $\eta = \frac{1}{4L}$ сходиться до оптимального значення w^* з лінійною швидкістю, а саме:

$$\|w_t - w^*\|^2 \leq \left(1 - \frac{\mu}{4L}\right)^t \|w_0 - w^*\|^2 \quad \forall t \geq 0$$

Доведення

Доведемо трохи більш загальний результат:

$$\|w_{t+1} - w^*\|^2 + \frac{1}{16} \|w'_t - w'_{t-1}\|^2 \leq \left(1 - \frac{\mu}{4L}\right) (\|w_t - w^*\|^2 + \frac{1}{16} \|w'_{t-1} - w'_{t-2}\|^2)$$

з чого буде випливати

$$\|w_{t+1} - w^*\|^2 \leq \|w_{t+1} - w^*\|^2 + \frac{1}{16} \|w'_t - w'_{t-1}\|^2 \leq \left(1 - \frac{\mu}{4L}\right)^{t+1} \|w_0 - w^*\|^2$$

за домовленості що $w'_0 = w'_{-1} = w'_{-2}$.

Спочатку доведемо допоміжні твердження.

Лема 1 Нехай $w \in \Omega$ і $w^+ = P_\Omega(w + u)$. Тоді $\forall w' \in \Omega$ виконується

$$\|w^+ - w'\|^2 \leq \|w - w'\|^2 + 2(u, w^+ - w') - \|w^+ - w\|^2$$

Доведення

$$\|w^+ - w'\|^2 = \|(w^+ - w) + (w - w')\|^2 = \|(w^+ - w)\|^2 + 2(w^+ - w, w - w') + \|(w - w')\|^2 = \|(w^+ - w)\|^2 + 2(w^+ - w, w^+ - w') - \|w^+ - w\|^2$$

Оскільки w^+ - проекція $w + u$ на опуклу множину Ω , маємо $(w^+ - (w + u), w^+ - w') \leq 0 \quad \forall w' \in \Omega$, з чого і отримуємо твердження леми.

Лема 2

Якщо F - μ -сильно монотонна, w^* - точка оптимуму, то виконується:

$$\mu (\|w_t - w^*\|^2 - 2\|w'_t - w_t\|^2) \leq 2(F(w'_t), w'_t - w^*)$$

Доведення

$$2\mu \|w'_t - w^*\|^2 \leq 2(F(w^*), w'_t - w^*) + 2\mu \|w'_t - w^*\|^2 \leq 2(F(w'_t), w'_t - w^*)$$

У першому переході ми скористалися оптимальністю точки w^* , у другому - μ -сильно монотонністю. Далі скористаємось нерівністю $\|2w'_t - w_t - w^*\|^2 \geq 0 \Rightarrow 2\|w'_t - w^*\|^2 \geq \|w_t - w^*\|^2 - 2\|w'_t - w_t\|^2$, підставляючи останнє в першу нерівність, отримуємо твердження леми.

Лема 3

Якщо F - L -Ліпшицева, то $\forall w \in \Omega$ виконується

$$2\eta_t(F(w'_t), w'_t - w) \leq \|w_t - w\|^2 - \|w_{t+1} - w\|^2 - \|w'_t - w_t\|^2 + \eta_t^2 L^2 \|w'_{t-1} - w'_t\|^2$$

Доведення

Використавши лему 1 для наборів $(w, u, w^+, w') = (w_t, -\eta_t F(w'_t), w_{t+1}, w)$ та $(w, u, w^+, w') = (w_t, -\eta_t F(w'_{t-1}), w'_t, w_{t+1})$, отримаємо

$$\|w_{t+1} - w\|^2 \leq \|w_t - w\|^2 - 2\eta_t(F(w'_t), w_{t+1} - w) - \|w_{t+1} - w_t\|^2$$

та

$$\|w'_t - w_{t+1}\|^2 \leq \|w_t - w_{t+1}\|^2 - 2\eta_t(F(w'_{t-1}), w'_t - w_{t+1}) - \|w'_t - w_t\|^2$$

Просумувавши ці дві рівності, отримаємо:

$$\|w_{t+1} - w\|^2 \leq$$

$$\|w_t - w\|^2 - 2\eta_t(F(w'_t), w_{t+1} - w) - 2\eta_t(F(w'_{t-1}), w'_t - w_{t+1}) - \|w'_t - w_t\|^2 - \|w'_t - w_{t+1}\|^2 =$$

$$\|w_t - w\|^2 - 2\eta_t(F(w'_t), w'_t - w) - \|w'_t - w_t\|^2 - \|w'_t - w_{t+1}\|^2 - 2\eta_t(F(w'_{t-1}) - F(w'_t), w'_t - w_{t+1})$$

Тепер скористаємося нерівністю $2(a, b) \leq \|a\|^2 + \|b\|^2$.

$$\begin{aligned} \|w_{t+1} - w\|^2 &\leq \|w_t - w\|^2 - 2\eta_t(F(w'_t), w'_t - w) + \eta_t^2 \|F(w'_{t-1}) - F(w'_t)\|^2 + \\ &\quad + \|w'_t - w_{t+1}\|^2 - \|w'_t - w_t\|^2 - \|w'_t - w_{t+1}\|^2 = \\ &= \|w_t - w\|^2 - 2\eta_t(F(w'_t), w'_t - w) + \eta_t^2 \|F(w'_{t-1}) - F(w'_t)\|^2 - \|w'_t - w_t\|^2 \leq \\ &\leq \|w_t - w\|^2 - 2\eta_t(F(w'_t), w'_t - w) + \eta_t^2 L^2 \|w'_{t-1} - w'_t\|^2 - \|w'_t - w_t\|^2 \end{aligned}$$

і після перенесення доданків отримаємо бажану нерівність.

Лема 4

Покладемо $w'_{-2} = w'_{-1} = w'_0$. Тоді для всіх $t \geq 0$ виконується:

$$\|w'_{t-1} - w'_t\|^2 \leq 4\|w_t - w'_t\|^2 + 4\eta_{t-1}^2 L^2 \|w'_{t-1} - w'_{t-2}\|^2 - \|w'_{t-1} - w'_t\|^2$$

Доведення

Скориставшись $\|a + b\|^2 \leq 2(\|a\|^2 + \|b\|^2)$, можемо записати:

$$\|w'_{t-1} - w'_t\|^2 \leq 2\|w_t - w'_t\|^2 + 2\|w_t - w'_{t-1}\|^2$$

Оскільки проекція на опуклу множину - нерозтягуюче відображення, маємо:

$$\begin{aligned} \|w_t - w'_{t-1}\|^2 &\leq \|w_{t-1} - \eta_{t-1} F(w'_{t-1}) - w_{t-1} - \eta_{t-1} F(w'_{t-2})\|^2 = \\ &= \eta_{t-1}^2 \|F(w'_{t-1}) - F(w'_{t-2})\|^2 \leq \\ &\leq \eta_{t-1}^2 L^2 \|w'_{t-1} - w'_{t-2}\|^2 \end{aligned}$$

Поєднуючи ці дві нерівності, отримуємо:

$$\begin{aligned} \|w'_{t-1} - w'_t\|^2 &= 2\|w'_{t-1} - w'_t\|^2 - \|w'_{t-1} - w'_t\|^2 \leq \\ &\leq 4\|w_t - w'_t\|^2 + 4\|w_t - w'_{t-1}\|^2 - \|w'_{t-1} - w'_t\|^2 \leq \\ &\leq 4\|w_t - w'_t\|^2 + 4\eta_{t-1}^2 L^2 \|w'_{t-1} - w'_{t-2}\|^2 - \|w'_{t-1} - w'_t\|^2 \end{aligned}$$

Доведення теореми

Поєднуючи лему 2 та лему 3, отримаємо:

$$\eta_t \mu (\|w_t - w^*\|^2 - 2\|w'_t - w_t\|^2) \leq \|w_t - w^*\|^2 - \|w_{t+1} - w^*\|^2 + \eta_t^2 L^2 \|w'_{t-1} - w'_t\|^2 - \|w'_t - w_t\|^2$$

з чого випливає

$$\|w_{t+1} - w^*\|^2 \leq (1 - \eta_t \mu) \|w_t - w^*\|^2 + \eta_t^2 L^2 \|w'_{t-1} - w'_t\|^2 - (1 - 2\eta_t \mu) \|w'_t - w_t\|^2$$

тепер використаємо лему 4:

$$\begin{aligned} \|w_{t+1} - w^*\|^2 &\leq (1 - \eta_t \mu) \|w_t - w^*\|^2 + \eta_t^2 L^2 (4\eta_{t-1}^2 L^2 \|w'_{t-1} - w'_{t-2}\|^2 - \|w'_{t-1} - w'_t\|^2) - \\ &\quad - (1 - 2\eta_t \mu - 4\eta_t^2 L^2) \|w'_t - w_t\|^2 \end{aligned}$$

Тепер покладемо $\eta_t = \frac{1}{4L} \leq \frac{1}{4\mu}$.

$$\|w_{t+1} - w^*\|^2 \leq \left(1 - \frac{\mu}{4L}\right) \|w_t - w^*\|^2 + \frac{1}{16} \left(\frac{1}{4} \|w'_{t-1} - w'_{t-2}\|^2 - \|w'_{t-1} - w'_t\|^2\right)$$

Використавши нерівність $\frac{\mu}{4L} \leq \frac{1}{4}$, можемо записати

$$\|w_{t+1} - w^*\|^2 + \frac{1}{16} \|w'_{t-1} - w'_t\|^2 \leq \left(1 - \frac{\mu}{4L}\right) (\|w_t - w^*\|^2 + \frac{1}{16} \|w'_{t-1} - w'_{t-2}\|^2)$$

4 Практичні результати

4.1 Метрика якості Inception Score

Для того, щоб аналізувати якість GAN та порівнювати результати для різних оптимізаційних методів, передусім нам потрібно мати деяку числову оцінку якості GAN. Звичайна функція помилки не завжди гарно корелює з якістю згенерованих фотографій, тому запропоновано низку інших метрик якості. Однією з найбільш розповсюджених є Inception Score (скорочено - IS), запропонована у 2016 році у [4]. Ідея Inception Score полягає у наступному. Ми хочемо враховувати

- 1) якість зображень, тобто щоб кожне зображення було схоже на якийсь конкретний існуючий об'єкт
- 2) різноманітність зображень.

Для цього використовується заздалегіть натренерований генератор. У [4] запропоновано використовувати модель Inception v3 (звідси і назва сетрики якості) - глибоку нейронну мережу, що класифікує кольорові зображення на 1000 класів (тобто видає вектор ймовірності приналежності кожному з класів).

Припустимо, що наша модель генерує зображення n класів. З одного боку, щоб зображення були якісними, ми вимагаємо, щоб для кожного зображення у векторі ймовірностей ймовірність одного класу була близькою до 1, інших - до 0. Для того ж, щоб зображення були різноманітними, ми вимагаємо, аби усереднений вектор ймовірності по усім зображенням був схожий на рівномірний розподіл по n класам. Це означає, що якщо виконуються обидві умови, то KL -дивіргенція між розподілами для одного зображення та для усередненого розподілу мають бути великою. В ідеальному випадку, коли розподіл одного зображення має ймовірність 1 для одного класу і ймовірність 0 для всіх інших, а усереднений - ймовірності $1/n$ для кожного з n класів. Тоді відстань Кульбака-Лейблера між ними дорівнює $\ln(n)$. Якщо ж взяти від неї експоненту, то отримаємо n . Отже, алгоритм обрахування Inception Score для генератора наступний:

1. Генеруємо m зображень і для кожного з них визначаємо вектор ймовірностей $(x_1^{(m)}, \dots, x_n^{(m)})$ за допомогою наперед натренерованого класифікатора Inception v3.
2. Знаходимо вектор усереднених ймовірностей $(\hat{x}_1, \dots, \hat{x}_n)$.
3. Для кожного зображення рахуємо величину $IS_i = \exp(KL((x_1^{(m)}, \dots, x_n^{(m)}), (\hat{x}_1, \dots, \hat{x}_n)))$.
4. Усереднююємо результат для всіх зображень: $IS = (IS_1 + \dots + IS_m)/m$.

Для генератора, що генерує "ідеальні" дані з n класів, маємо отримати значення, близьке до n . Для випадкових шумів отримаємо значення, близьке до 1. На практиці ж в якості оцінки IS, яку треба досягнути, беруть не число n , а беруть її як IS для зображень з тренувальних даних.

Надалі для обрахування IS ми генерували 1000 випадкових зображень і вираховували IS по ним. Всі програми навчалися тренувати цифри з набору mnist, на якому Inception Score показує результат якості приблизно 2.0. При реалізації усіх моделей використовувалася DCGAN-архітектура для генератора та дискримінатора. Параметри Adam-оптимізаторів бралися learning rate = 0.0002, beta1 = 0.5, як рекомендується у [5] для архітектур DCGAN. Також зазначимо, що для оптимізатора екстраградієнтний Adam, враховуючи, що він потребує 2 рази обчислювати градієнт для 1 оновлення, ми робили в 2 рази менше оновлень, ніж для звичайного Adam і оптимістичного Adam. Тобто час виконання програм для усіх оптимізаторів приблизно рівний. Всі програми виконувались у середовищі Google Colab із використанням GPU прискорювача.

4.2 Conditional GAN (CGAN)

Код до цієї частини можна знайти за посиланням

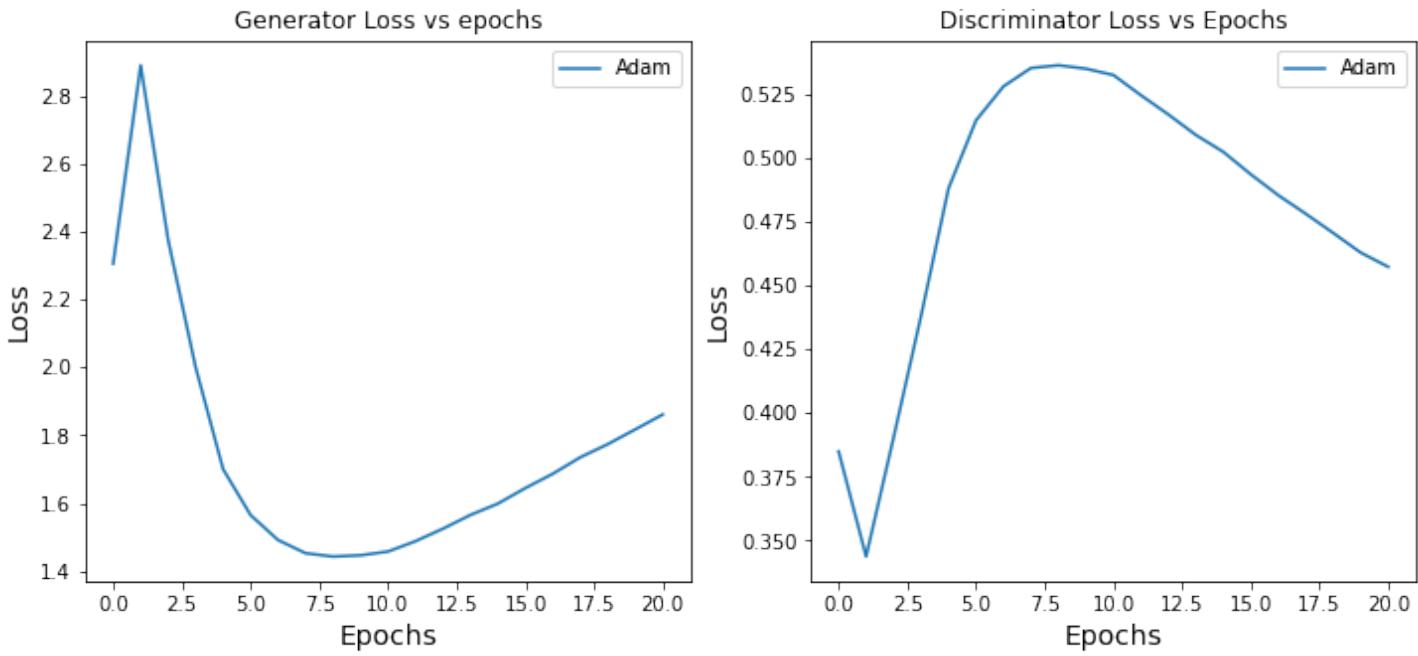
<https://github.com/DenisPushkin/CGAN/blob/master/CGAN.ipynb>

Архітектура CGAN тренувалася на 21 епосі для кожного з оптимізаторів. Також рахувалося усереднене значення змінних CGAN, що усереднювалось починаючи з 12-ої епохи.

Adam

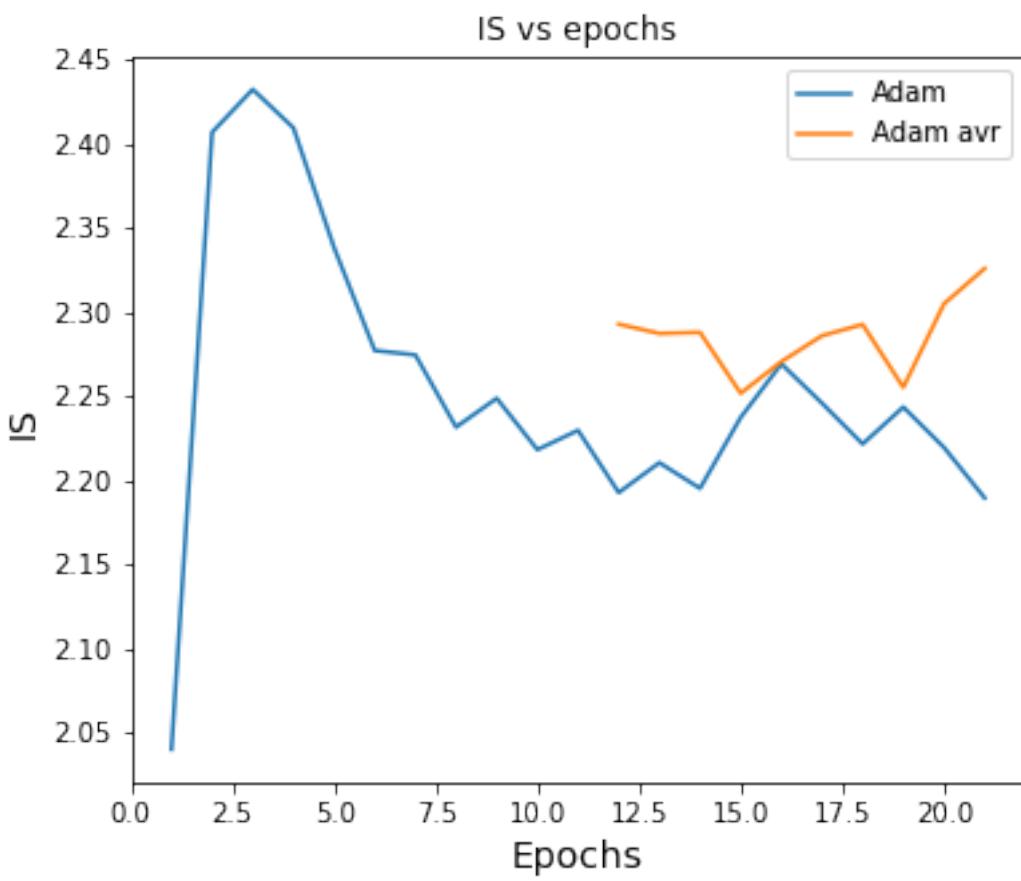
Використовуючи стандартний оптимізатор Adam, були досягнуті наступні результати. Всі наступні графіки беруться як усереднення відповідних характеристик по трьом запускам програми.

Графік помилок для



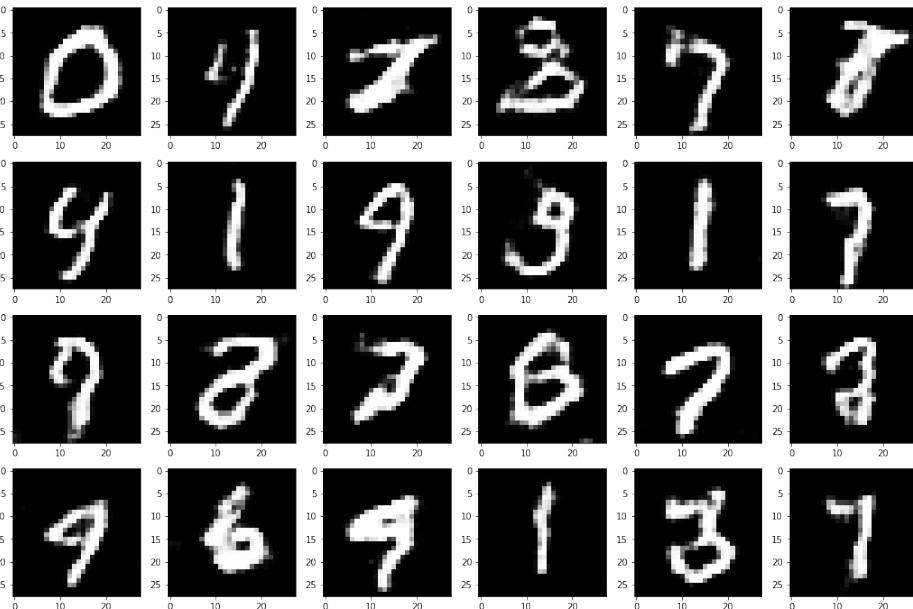
Бачимо, що впродовж перих 2-3 епох дискримінатор значно переважає генератор. Це пояснюється тим, що задача дискримінатора - на перших порах це бінарона класифікація між зображенням цифр і шумами, - простіша, ніж задача генератора - генерування зображення 28×28 . Тому дискримінатор значно швидше навчається до деякого прийнятного стану, ніж генератор. Далі навпаки, генератор навчається генерувати більш-менш правдоподібні зображення, а дискримінатор, який до цього отримував від генератора на вихід шуми, починає занадто довіряти генератору. Зазвичай вже під час цієї фази генератор навчається генерувати більш-менш правдоподібні зображення. Втім, приблизно після 10-ої епохи дискримінатор навчається помічати більш глибинні ознаки і починає поспутово зменшувати свою функцію помилки (знову таки, відіграє свою роль те, що задача дискримінатора простіша). Але зростання функції помилки генератора не означає, що він генерує гірші фотографії: ми побачимо, що Inception Score для генератора продовжить рости. Якщо продовжувати тренування, то зазвичай дискримінатор так і продовжуватиме потроху покращувати свої позиції, доки функції помилок не перестануть змінюватися майже зовсім.

Описана історія тренування, яку за графіками функцій помилок можна умовно поділити на 3 фази, є притаманною для успішного тренування GAN (крім Wasserstein GAN, які використовують іншу функцію помилки). Часто у тих випадках, коли тренування відбувається по іншому сценарію, є сенс перезапустити тренування або взагалі змінити модель. Наприклад, якщо друга фаза (коли генератор переважає дискримінатор) довго не закінчується, це є ознакою затухання градієнтів у обох мережах.

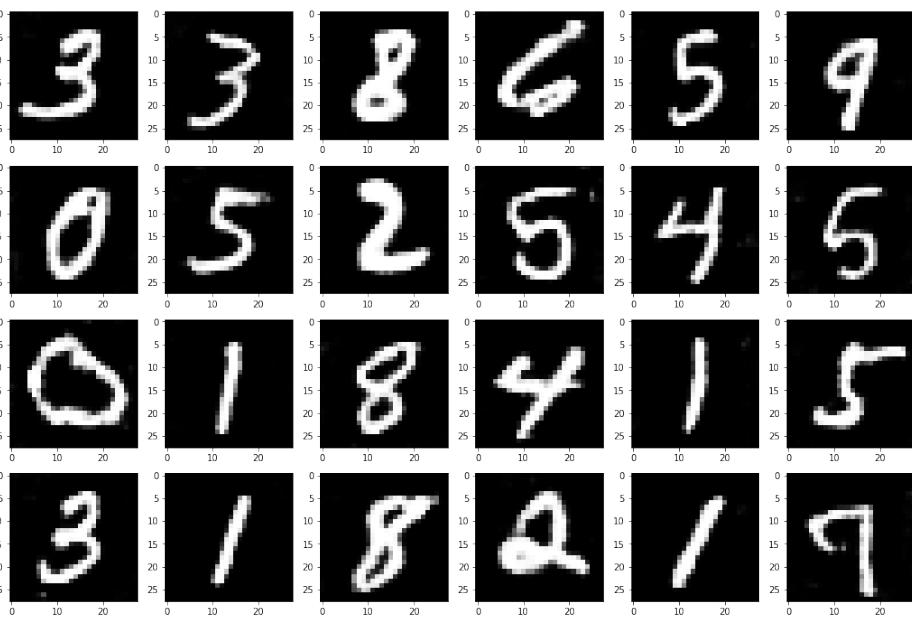


Бачимо один несподіваний ефект: найбільше значення Inception Score набуває після 2-4 епох. Порівняємо приклади згенерованих зображень після 3-ої епохи, коли значення IS було найбільше, та після останньої 21-ої.

Після 3-ої:



Після 21-ої:



Бачимо, що насправді і цьому випадку Inception Score нас обманює, після 21-ої епохи згенеровані фотографії набагато більше нагадують цифри. Важливо розуміти, що насправді серед 1000 класів, на які класифікує модель Inception v3, немає цифр, коректність оцінки опирається на припущення, що якісне зображення повинне нагадувати 1 із 1000 існуючих класів у цій моделі (або принаймні малу кількість схожих класів). Насправді ж, хоча добре відомо, що модель Inception Score може обманювати, на практиці перевірено, що вона в переважній більшості випадків дає більш-менш коректну оцінку генератору на 2-ій половині тренування. До того ж, Inception Score для самих зображень із mnist приблизно дорівнює 2.0. Тому якщо IS значно вищий за цю величину, це означає, що скоріш за все генеруються не зовсім зображення із розподілу цифр mnist.

Розглядаючи графік IS на другій половині епох бачимо, що IS-метрика при усереднених вагах моделі дає стабільно кращий результат, ніж при вагах на останній ітерації (в середньому на 2.86%).

Осолівістю CGAN є те, що ми можемо самі задавати мітки класу, який хочемо згенерувати. Для демонстрації результатів було послідовно згенеровано по 18 прикладів кожної з цифр.

Результати генерування після 21 епох, використовуючи ваги на останній ітерації:

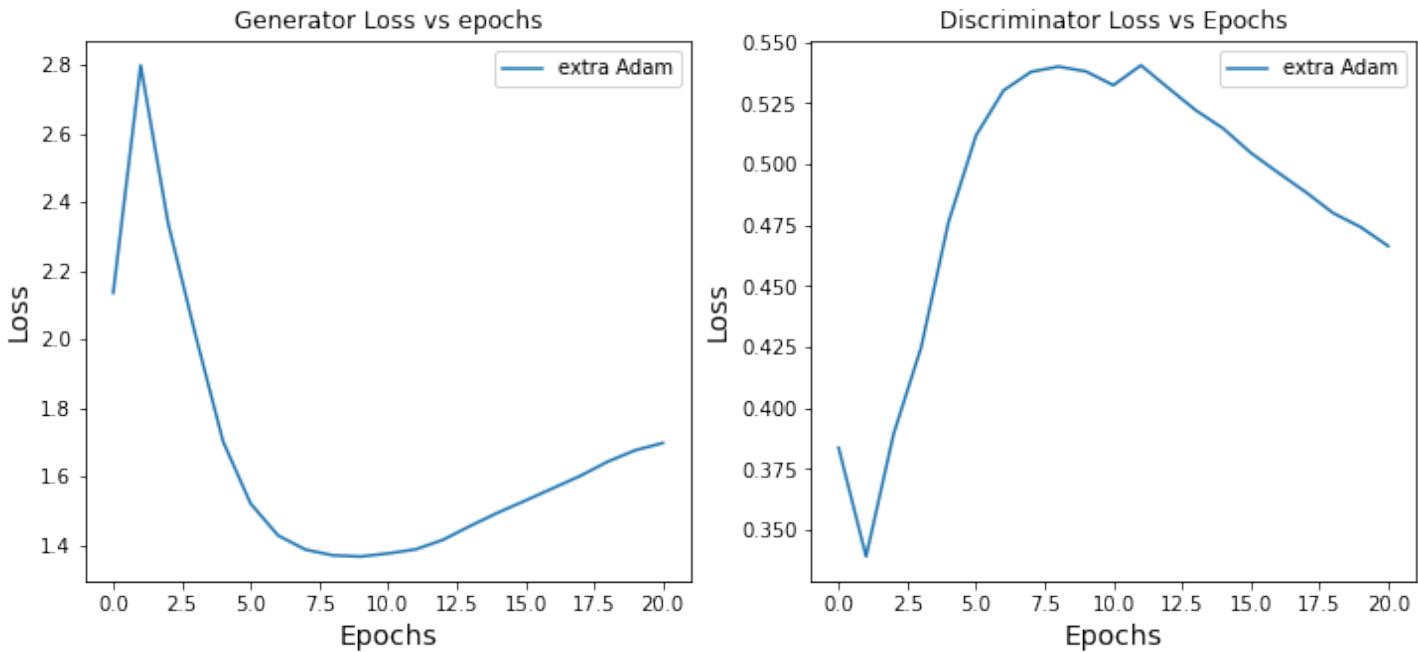
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2
3	3	3	3	3	3
3	3	3	3	3	3
3	3	3	3	3	3
4	4	4	4	4	4
4	4	4	4	4	4
4	4	4	4	4	4
5	5	5	5	5	5
5	5	5	5	5	5
5	5	5	5	5	5
6	6	6	6	6	6
6	6	6	6	6	6
6	6	6	6	6	6
7	7	7	7	7	7
7	7	7	7	7	7
7	7	7	7	7	7
8	8	8	8	8	8
8	8	8	8	8	8
8	8	8	8	8	8
9	9	9	9	9	9
9	9	9	9	9	9
9	9	9	9	9	9

Результати генерування після 21 епох, використовуючи асережнені ваги по другій половині ітерацій (з 12-ої по 21-у):

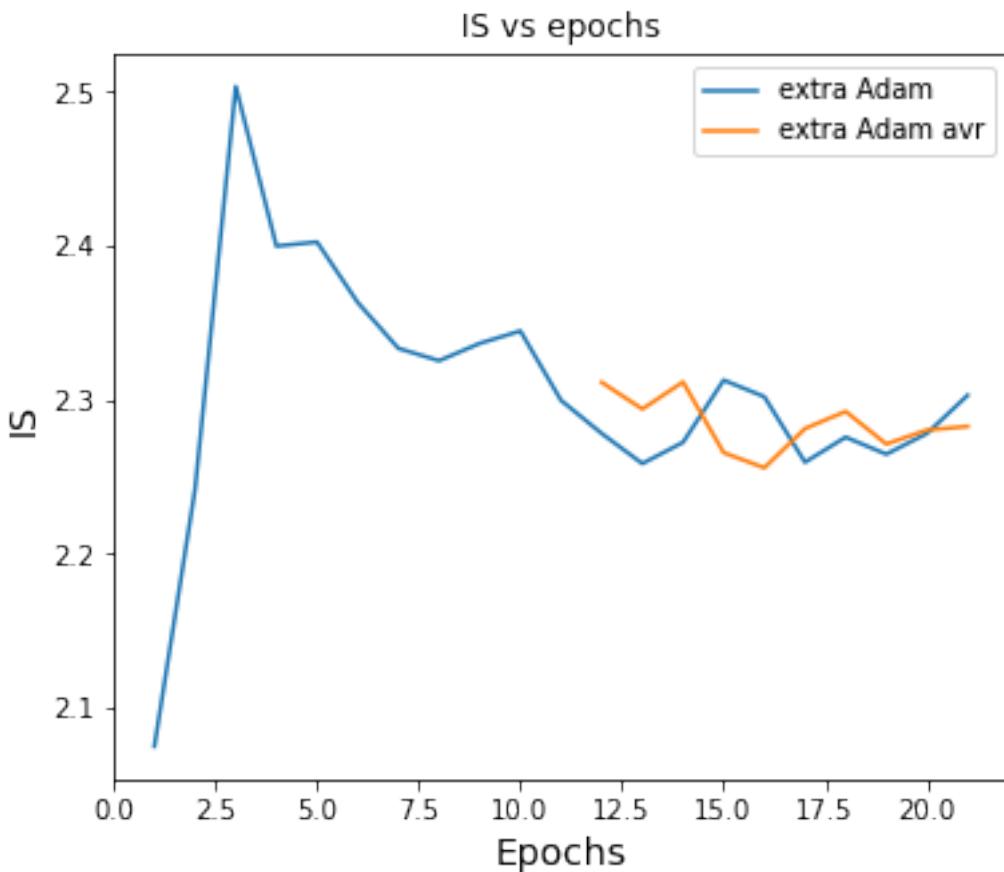
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2
3	3	3	3	3	3
3	3	3	3	3	3
3	3	3	3	3	3
4	4	4	4	4	4
4	4	4	4	4	4
4	4	4	4	4	4
5	5	5	5	5	5
5	5	5	5	5	5
5	5	5	5	5	5
6	6	6	6	6	6
6	6	6	6	6	6
6	6	6	6	6	6
7	7	7	7	7	7
7	7	7	7	7	7
7	7	7	7	7	7
8	8	8	8	8	8
8	8	8	8	8	8
8	8	8	8	8	8
9	9	9	9	9	9
9	9	9	9	9	9
9	9	9	9	9	9

extra Adam

Наступним був використаний оптимізаційний метод Adam, побудований на основі екстраградієнтного методу.

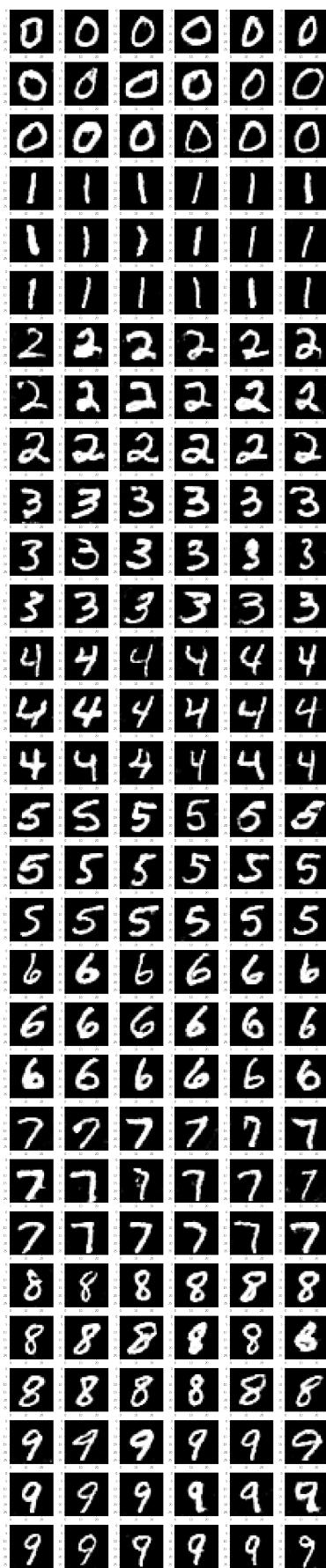


Бачимо, що графіки функцій помилки поводять себе аналогічним чином.



Але на відміну від звичайного методу Adam, при екстраградієнтному Adam немає виграшу від використання усереднених ваг (на трьох запусках програми середній виграш усереднення, що рахувався для кожної з епох 12-21, на яких використовувалося усереднення, складає лише 0.19%). Порівняння різних оптимізаторів між собою проведемо у кінці.

Приклади згенерованих зображень після 21 епохи без використання усереднення:

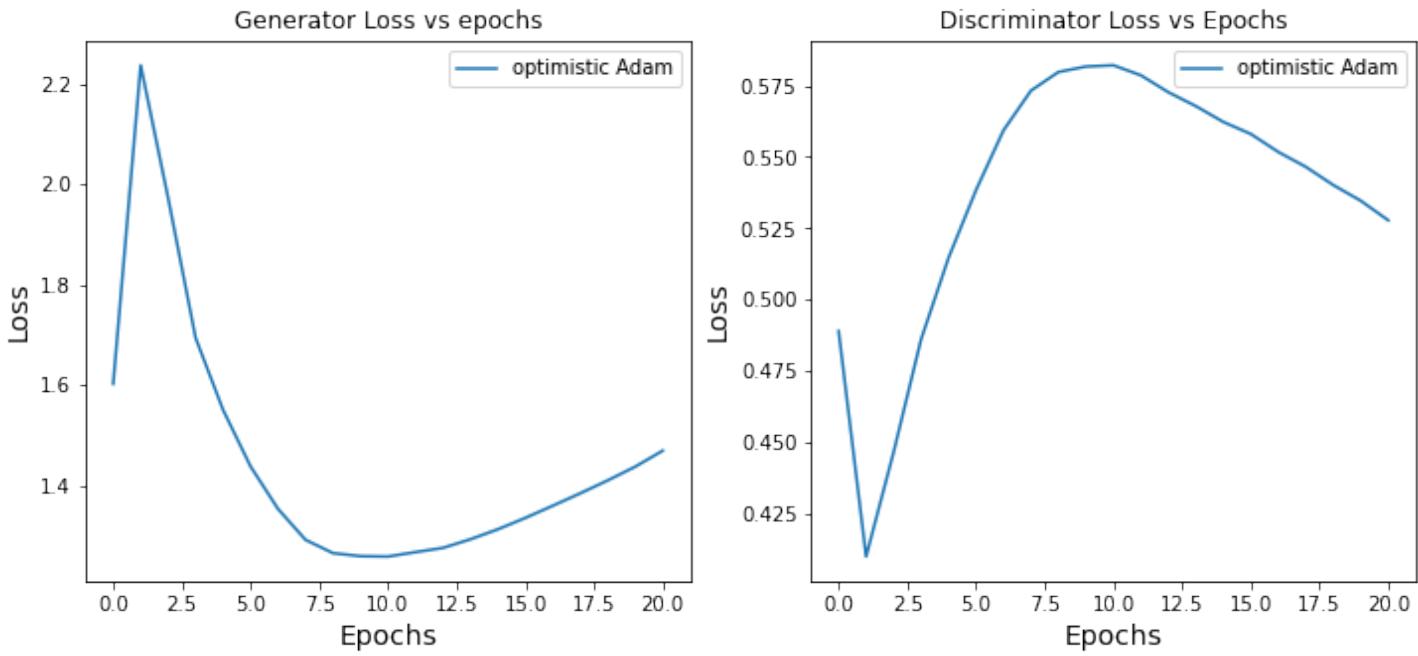


Приклади згенерованих зображень після 21 епохи з використання усереднення:

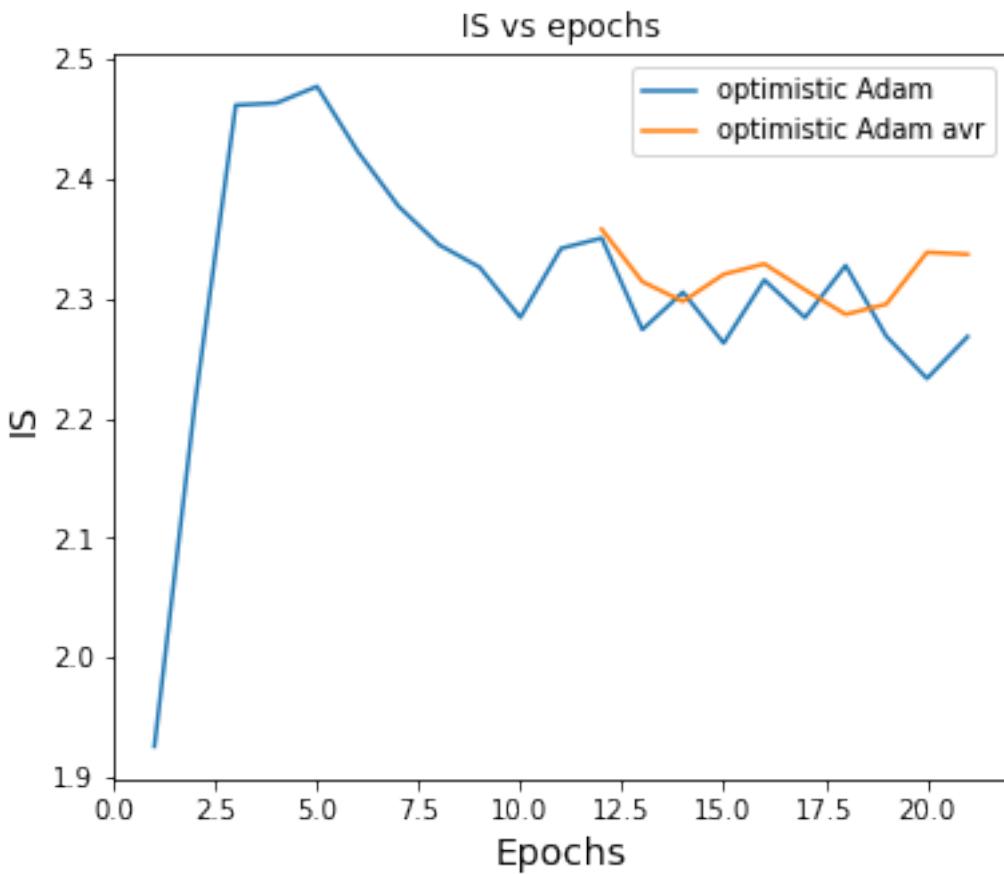
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2
3	3	3	3	3	3
3	3	3	3	3	3
3	3	3	3	3	3
4	4	4	4	4	4
4	4	4	4	4	4
4	4	4	4	4	4
5	5	5	5	5	5
5	5	5	5	5	5
5	5	5	5	5	5
6	6	6	6	6	6
6	6	6	6	6	6
6	6	6	6	6	6
7	7	7	7	7	7
7	7	7	7	7	7
7	7	7	7	7	7
8	8	8	8	8	8
8	8	8	8	8	8
8	8	8	8	8	8
9	9	9	9	9	9
9	9	9	9	9	9
9	9	9	9	9	9

optimistic Adam

При використанні оптимізаційного методу оптимістичний Adam, були досягнуті наступні результати.

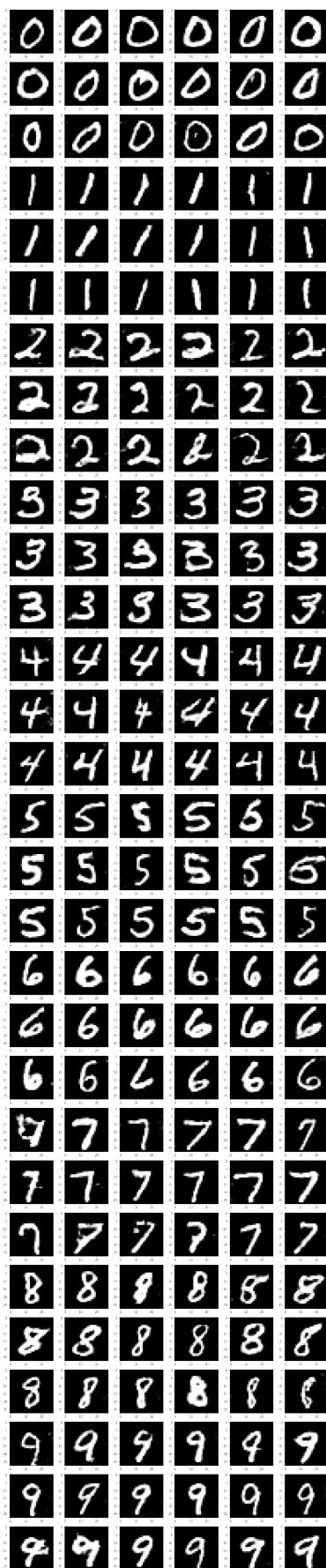


Знову графіки функцій помилки поводять себе аналогічно.



Бачимо, що використання усереднення дає більш стабільні результати. В середньому усереднення дає на 1.31% кращі результати.

Приклади згенерованих зображень після 21 епохи без використання усереднення:



Приклади згенерованих зображень після 21 епохи з використання усереднення:

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
2	2	2	2	2	2
2	2	2	2	2	2
2	2	2	2	2	2
3	3	3	3	3	3
3	3	3	3	3	3
3	3	3	3	3	3
4	4	4	4	4	4
4	4	4	4	4	4
4	4	4	4	4	4
5	5	5	5	5	5
5	5	5	5	5	5
5	5	5	5	5	5
6	6	6	6	6	6
6	6	6	6	6	6
6	6	6	6	6	6
7	7	7	7	7	7
7	7	7	7	7	7
7	7	7	7	7	7
8	8	8	8	8	8
8	8	8	8	8	8
8	8	8	8	8	8
9	9	9	9	9	9
9	9	9	9	9	9
9	9	9	9	9	9

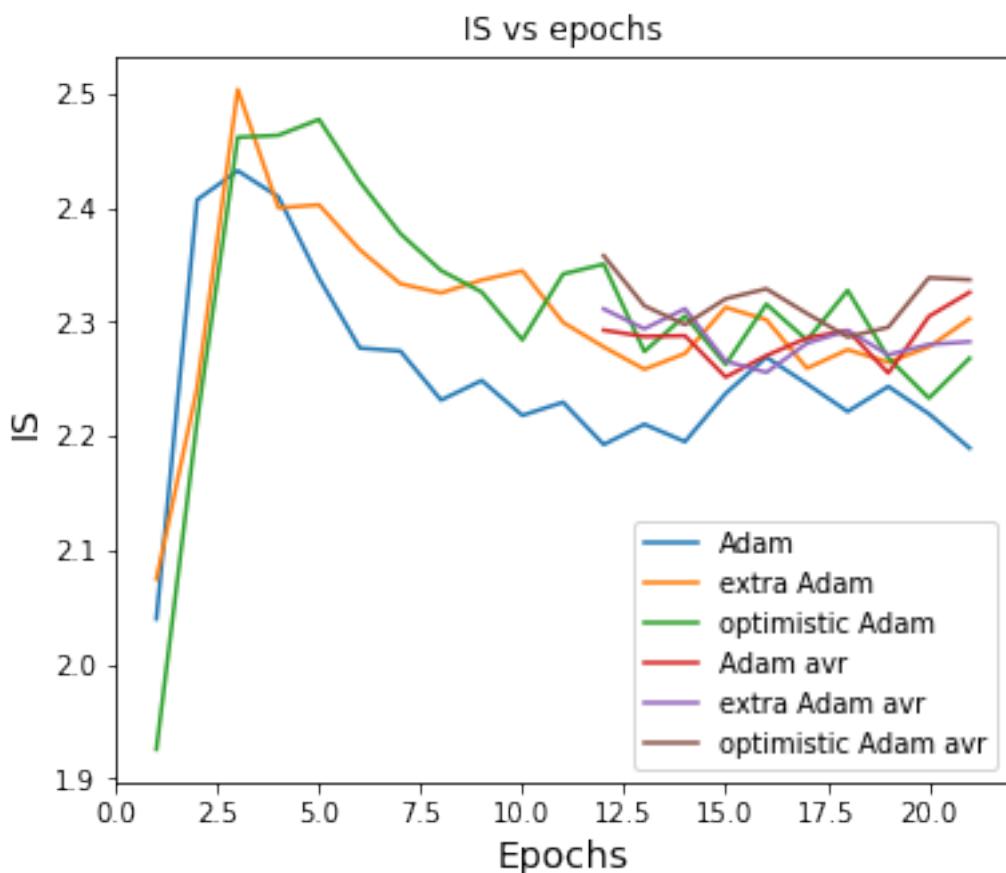
Порівняння результатів

Для аналізу результатів, ми усереднили IS-метрику для усіх оптимізаторів по останнім 10 епохам. Спочатку порівняємо виграло від усереднення для усіх оптимізаторів.

	Adam	extra Adam	optimistic Adam
IS без усереднення	2.22	2.28	2.29
IS з усередненням	2.29	2.28	2.32
виграш від усереднення, %	2.86%	0.19%	1.31%

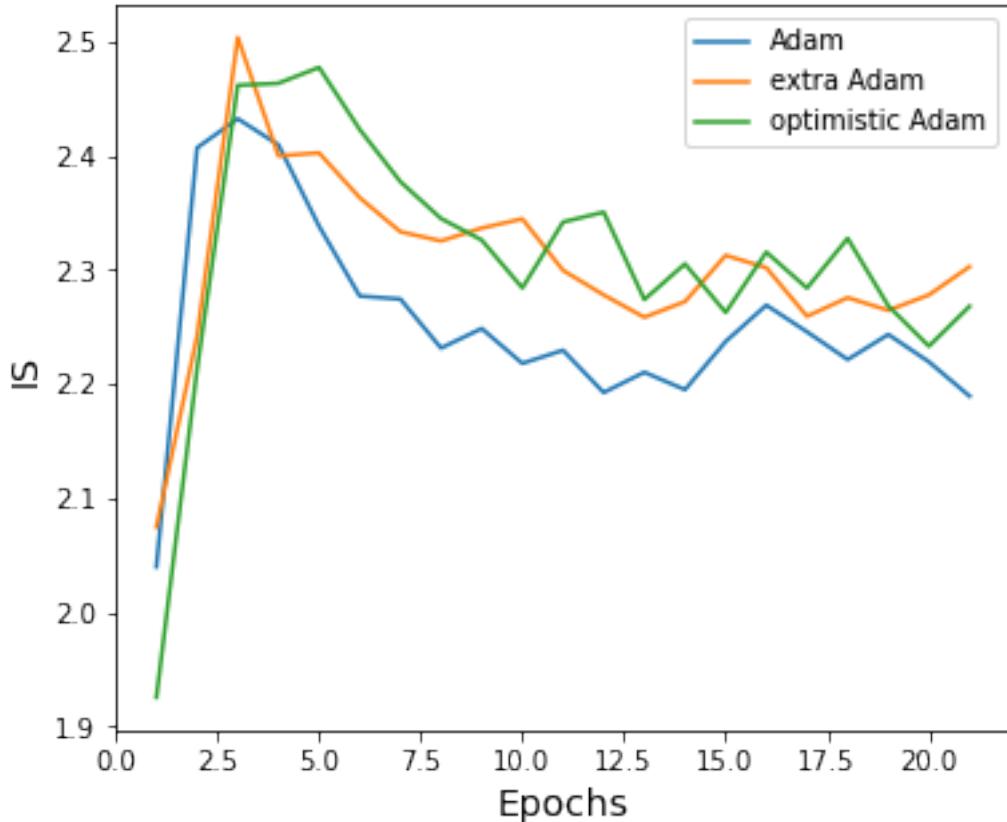
Бачимо, що суттєвими є виграші від усереднення лише для Adam та оптимістичний Adam.

Тепер порівняємо результати різних оптимізаторів та їх усереднень між собою (нагадаємо, що усереднення починалися робитися з 12-ої епохи, починаючи з уього місця і будуються графіки для усереднень). Графіки IS-метрики виглядають наступним чином:

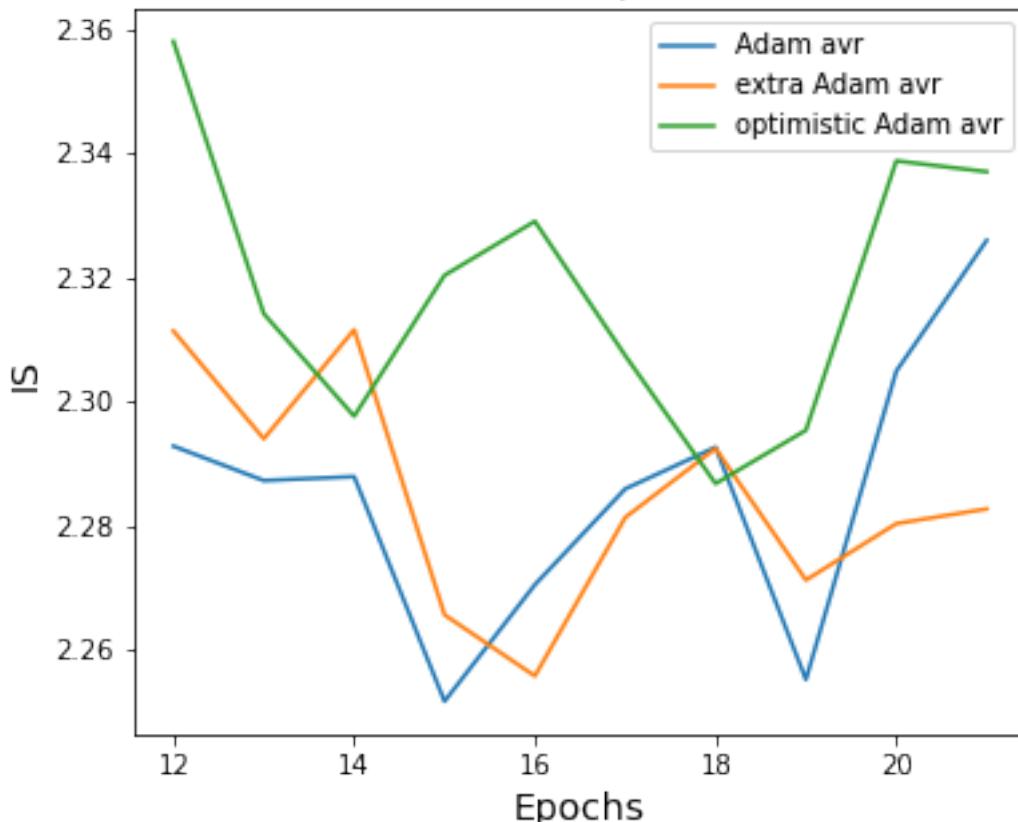


Для більшої наочності виведемо окремо графіки без усереднень та графіки з усередненням.

IS vs epochs



IS avr vs epochs



Найменше значення IS-метрики дає звичайний Adam. Прийнявши його якість за 1, виведемо умовні якості для оптимізаторів.

	результат IS	відносний результат IS
Adam	2.22	1.000
Adam з усер.	2.29	1.028
extra Adam	2.28	1.026
extra Adam з усер.	2.28	1.028
optimistic Adam	2.29	1.030
optimistic Adam з усер.	2.32	1.043

Бачимо, що для цієї архітектури GAN використання екстраградієнтного Adam та оптимістичного Adam підвищує метрику якості на 2.8-3.0%. Якщо ж робити усереднення, для Adam та екстраградієнтний Adam якість підвищується приблизно на 2.8%, а для оптимістичного Adam - на 4.3%, що є найкращим результатом. Підвищення якості результату через усереднення вказує на те, що ітерації методів коливаються навколо оптимального значення. Відсутність суттєвого виграншу від усереднення для екстраградієнтного методу вказує на те, що для нього коливання найменші.

4.3 DCGAN

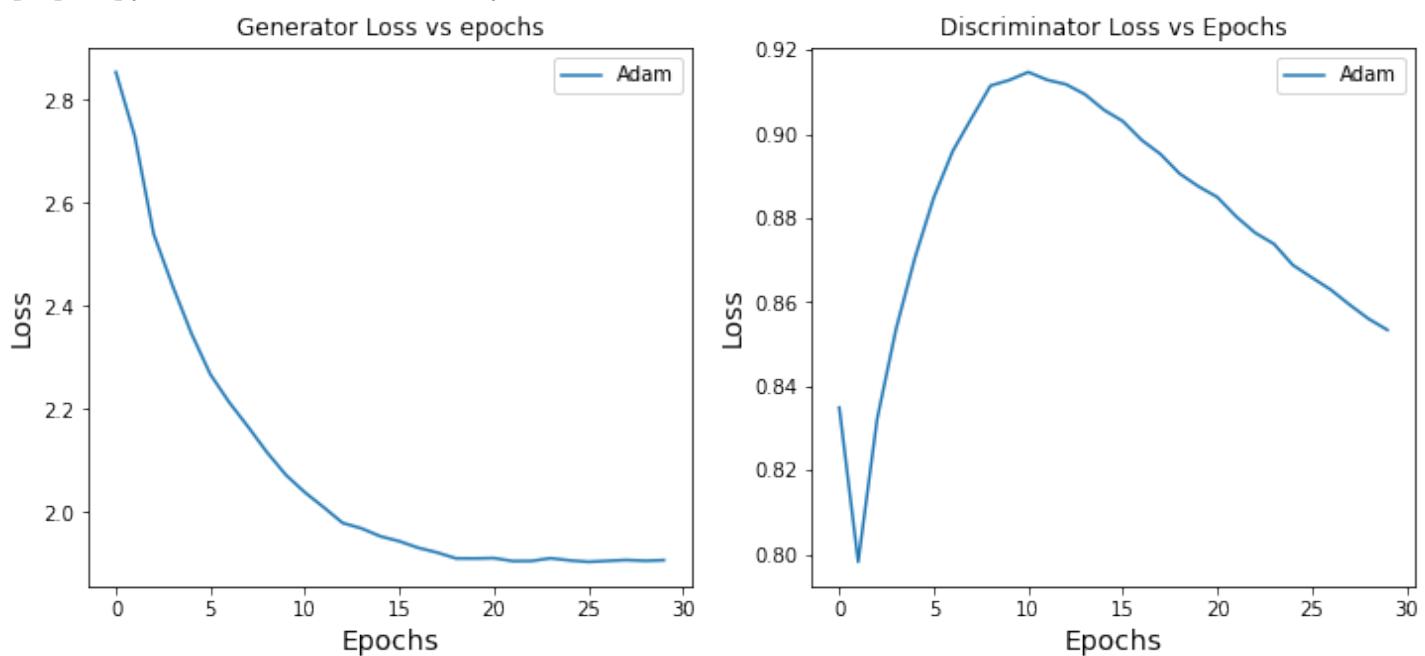
Код до цієї частини можна знайти за посиланням

<https://github.com/DenisPushkin/DCGAN/blob/master/DCGAN.ipynb>

Перейдемо до розгляду іншої архітектури - Deep Convolutional GAN, або просто DCGAN. Ця архітектура навчається повільніше за CGAN, тому ми тренували на 30 епохах, а також рахували усереднене значення вагів на останніх 15 епохах. Як і для CGAN, для кожного оптимізаційного методу ми запускали програму 3 рази і всі характеристики брали як середнє у цих трьох запусках.

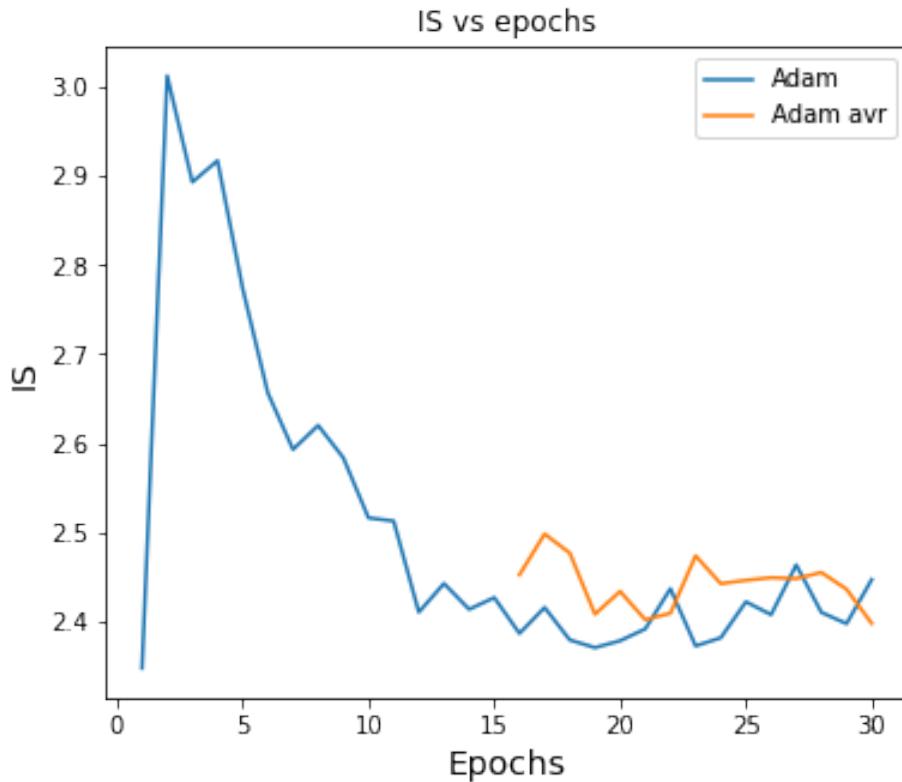
Adam

Графік функцій помилок має наступний вигляд



Цей графік вказує на те, що, можливо, є сенс продовжити навчання після 30 епох (скоріш за все, далі функція помилки генератора почне потроху зростати). Втім, якість зображень від цього зміниться не суттєво, тому ми обмежились 30 епохами.

Графік IS-метрики:



Бачимо, що усереднення трохи покращує якість (в середньому на 1.59%, де усереднюються по кожній з останніх 15 епох, де рахувалось усереднення).

Приклади згенерованих зображень після 30 епох без використання усереднення:

9	1	0	9	6	3
0	6	1	7	1	1
8	6	1	9	5	2
4	4	0	8	0	5
6	0	8	8	4	1
6	9	1	6	0	1
0	0	0	0	8	9
0	9	4	2	9	1
6	7	8	4	6	4
0	2	1	9	8	5
9	0	8	4	9	4
4	9	1	0	9	3
0	0	1	0	0	4
1	0	4	9	1	6
1	6	0	6	6	7
6	0	0	6	9	5
4	5	4	1	0	6
5	0	1	7	0	6
0	3	8	1	0	0
1	3	0	4	4	6
4	6	4	0	4	6
7	5	0	4	2	3
8	6	8	0	0	6
1	9	5	0	8	9
0	9	5	0	8	2
0	6	9	5	6	8
4	4	9	0	1	7
6	5	9	1	1	5
8	9	6	8	6	9
1	0	8	1	2	1

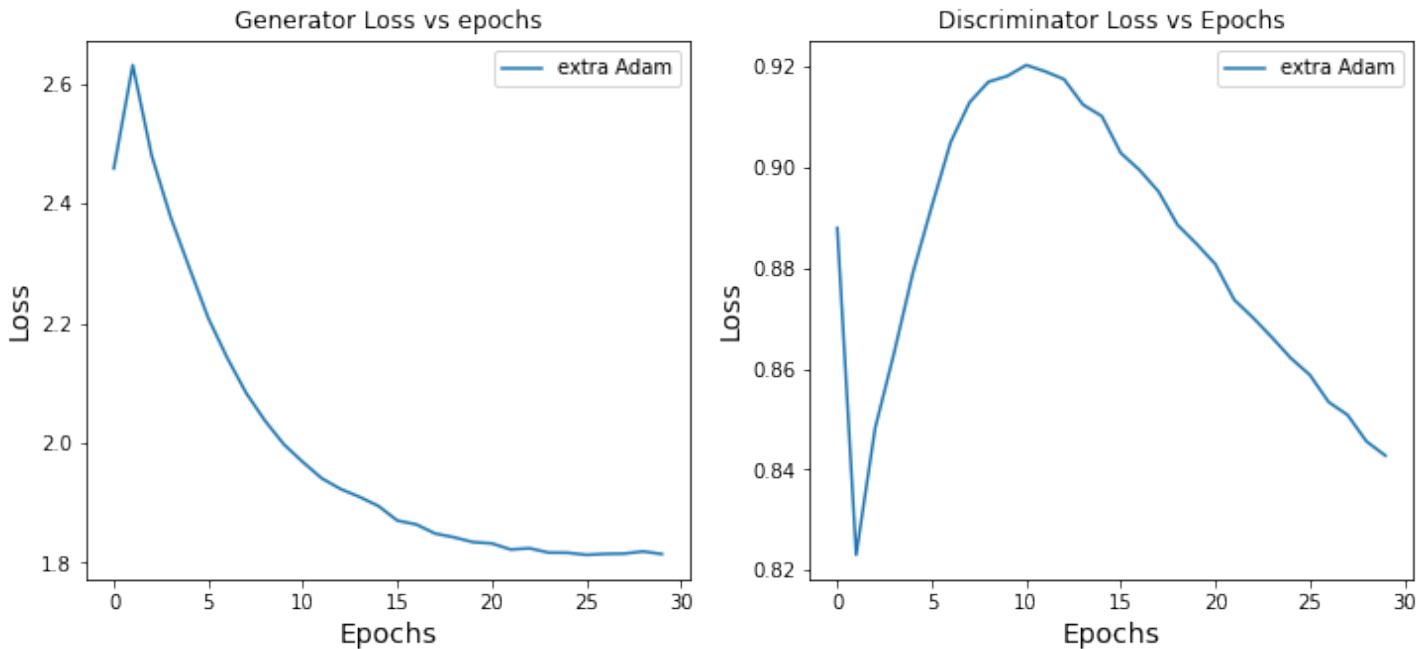
Приклади згенерованих зображень після 30 епох з використання усереднення:

1	3	4	4	4	4
1	5	1	8	2	0
6	9	6	8	9	1
6	1	7	1	0	5
5	6	9	0	7	0
3	1	9	1	3	4
0	5	0	0	3	4
0	1	4	9	4	1
0	6	6	7	3	1
9	6	0	3	0	0
0	0	8	6	1	4
0	0	3	9	9	5
6	0	1	9	9	6
7	6	3	6	8	5
9	5	1	1	8	8
0	0	0	1	0	8
6	9	4	0	4	6
0	8	9	9	4	4
1	0	4	5	4	9
1	6	0	6	6	9
1	6	5	1	3	0
7	5	9	6	1	8
3	1	0	8	5	4
8	8	9	0	8	0
6	6	1	7	9	9
8	6	1	6	9	1
9	6	1	9	0	5
7	9	6	0	9	7
5	0	6	0	4	1
6	7	1	6	5	6

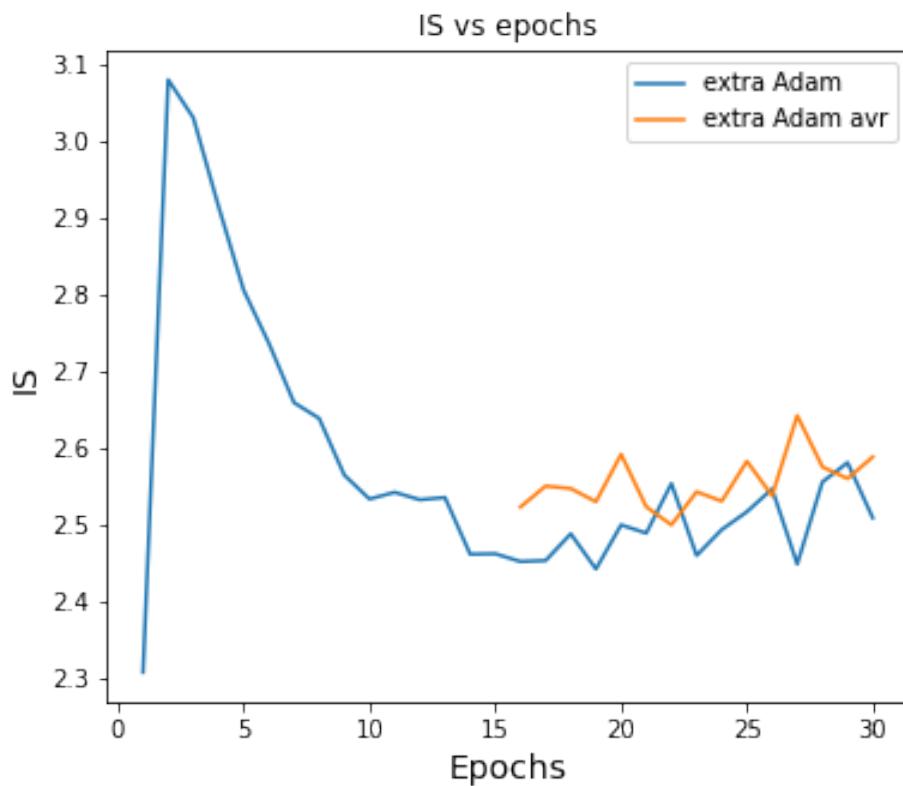
Нагадаємо, що без використання умовних моделей GAN ми не можемо задавати, які самі цифри будуть згенеровані. Вадою моделі DCGAN є те, що якщо тренувати модель занадто довго, то модель буде генерувати "прості" цифри, такі як 0, 1, 7, значно частіше, ніж складні для генерування. Це ще одна причина, чому ми обмежелись тренуванням DCGAN на 30 епохах, де вже починає простежуватись даний ефект.

естраградієнтний Adam

Графік функцій помилок аналогічний:



Графік IS-метрики:



Бачимо, що навідміну від архітектури CGAN, у цьому випадку усереднення при оптимізаторі естраградієнтний Adam відсутньо покращує метрику якості зображень (в середньому на 2.26%).

Приклади згенерованих зображень після 30 епох без використання усереднення:

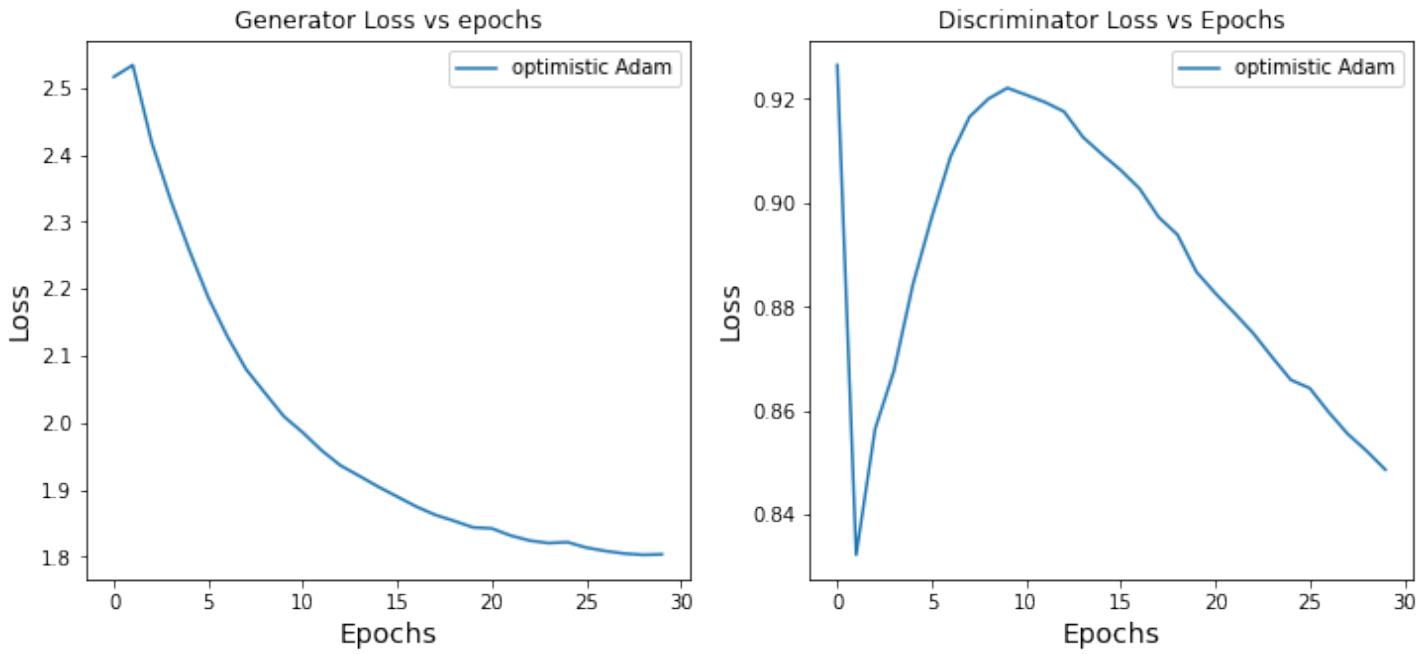
2	6	6	1	6	6
4	4	1	7	2	6
6	0	5	6	1	6
5	2	8	6	1	4
4	9	1	6	4	1
1	4	6	1	6	5
9	4	1	1	8	6
1	1	6	4	0	6
6	6	0	4	5	6
7	7	6	6	6	3
1	4	1	5	6	2
7	1	4	4	0	0
6	7	1	2	0	1
5	1	6	4	5	7
0	1	6	7	3	0
6	9	4	0	1	0
5	5	6	4	6	6
4	0	6	0	4	8
1	6	1	6	1	4
3	9	0	1	6	8
4	6	5	5	0	6
9	1	9	1	6	8
6	6	4	6	8	9
3	2	6	4	7	6
6	6	6	8	0	1
9	2	2	9	6	6
6	9	6	6	6	1
6	6	4	1	6	9
4	5	6	3	3	9
0	0	0	6	9	1

Приклади згенерованих зображень після 30 епох з використанням усереднення:

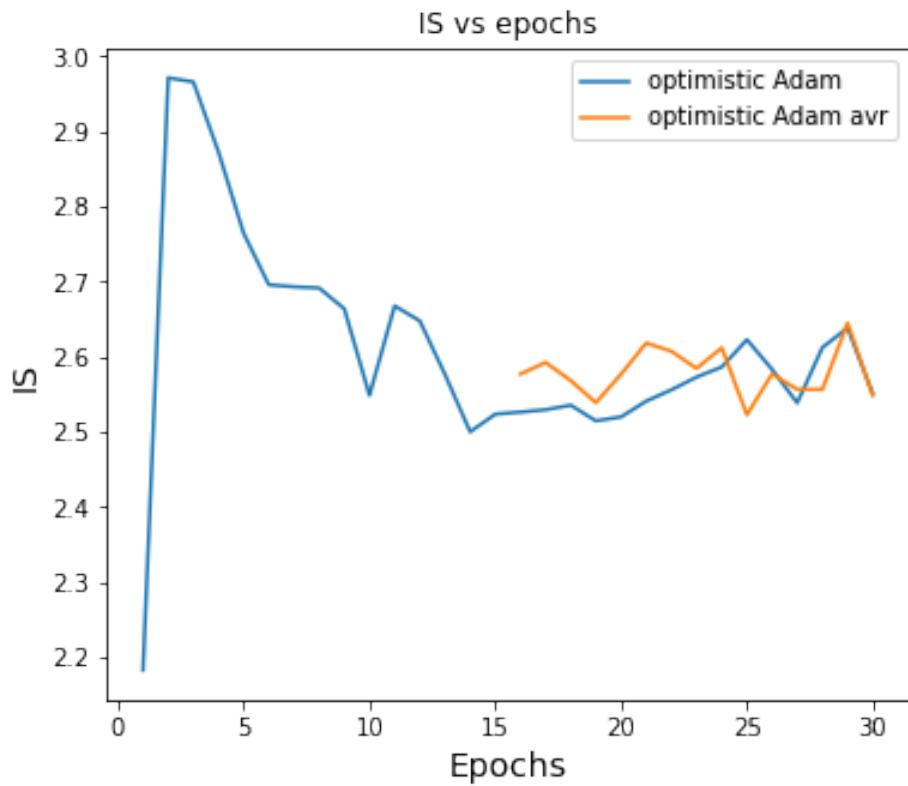
6	5	5	5	6	6
4	8	9	4	3	2
1	1	0	8	0	6
9	6	0	6	9	2
6	5	1	3	6	6
9	9	1	0	4	5
9	6	7	8	0	1
9	6	9	6	3	9
9	6	7	9	7	7
4	9	9	0	5	0
0	9	6	2	5	4
3	6	6	1	6	3
3	3	4	3	0	9
0	5	6	6	1	9
7	3	6	6	8	1
6	9	9	6	9	7
7	0	9	6	6	1
4	8	1	7	9	8
1	6	5	0	7	4
9	2	8	6	5	1
3	1	1	4	0	8
8	8	1	1	6	6
6	6	6	6	1	5
0	9	9	4	1	1
5	0	6	7	5	3
5	6	5	6	8	8
4	6	4	1	7	6
9	3	2	3	7	4
0	4	9	1	0	6
0	1	5	8	7	0

оптимістичний Adam

Графік функцій помилок має вигляд, аналогічний для Adam:



Графік IS-метрики:



Бачимо, що суттєвого виграншу від усереднення немає, він складає лише приблизно 0.68%.
Приклади згенерованих зображень після 30 епох без використання усереднення:

b	2	0	1	1	6
0	1	1	9	9	1
1	6	1	9	9	1
5	8	2	0	1	4
1	9	7	3	6	0
4	1	6	1	2	9
2	1	0	6	5	4
1	7	9	9	1	6
0	7	6	7	4	7
9	0	1	9	7	8
2	9	6	0	0	5
2	1	1	7	6	1
1	2	9	4	6	4
7	9	8	6	8	4
2	9	0	2	3	9
7	4	0	2	1	6
1	6	9	7	8	1
0	3	1	8	6	1
1	6	3	1	0	1
6	0	9	4	9	6
1	1	1	1	1	0
6	9	7	6	1	6
4	0	0	7	6	6
4	9	9	4	5	9
3	6	4	7	9	0
4	9	0	4	4	1
2	5	6	7	0	1
6	0	0	1	1	1
6	6	1	9	7	7
9	9	9	1	6	6

Приклади згенерованих зображень після 30 епох з використанням усереднення:

0	2	0	7	9	6
2	3	8	4	6	6
8	7	3	3	9	6
0	1	6	8	6	1
6	1	3	1	6	6
3	9	1	8	6	2
7	9	6	4	0	3
0	6	1	1	7	0
1	9	0	1	8	6
7	8	4	7	4	0
6	9	5	2	5	1
1	6	1	1	3	1
0	7	0	1	7	0
0	1	5	1	1	6
3	4	8	9	8	9
6	1	6	0	4	1
7	6	7	4	0	1
2	9	6	6	2	7
6	2	7	6	1	1
1	9	6	1	7	7
1	0	0	1	8	8
0	7	0	4	8	2
0	2	5	6	6	6
4	6	9	3	6	5
5	6	5	3	8	1
8	5	0	0	7	0
3	9	7	1	6	6
2	3	3	1	5	3
3	0	8	8	9	6
3	3	8	3	2	2

Порівняння результатів

Як і минолога разу, для аналізу результатів ми усереднили IS-метрику для усіх оптимізаторів по другій половині епох (16-30). Спочатку порівняємо виграш від усереднення для усіх оптимізаторів.

	Adam	extra Adam	optimistic Adam
IS без усереднення	2.40	2.50	2.56
IS з усередненням	2.44	2.56	2.58
виграш від усереднення, %	1.59%	2.26%	0.68%

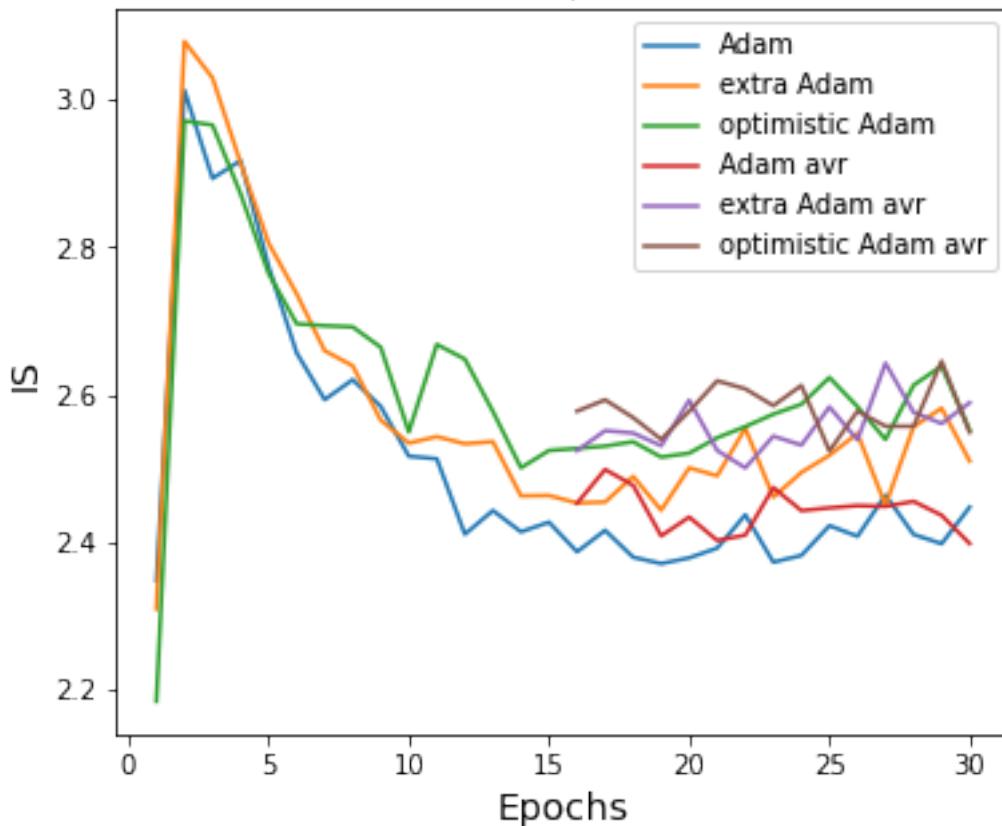
Бачимо, що на цей раз уже оптимістичний Adam має маргінальне покращення від усереднення, а покращення для звичайного Adam та екстраградієнтного Adam суттєві. Можна припустити, що оптимістичний Adam має найменші коливання, коли наближується до розв'язку. Втім, якби коливань взагалі не було, то слід було би очікувати меншу якість згенерованих зображень при усередненні. Також бачимо, що як і минулого разу, найменшу IS-метрику має звичайний Adam. Прийнявши його якість за 1, обчислимо відносні якості усіх методів.

	результат IS	відносний результат IS
Adam	2.40	1.000
Adam з усер.	2.44	1.016
extra Adam	2.50	1.040
extra Adam з усер.	2.56	1.063
optimistic Adam	2.56	1.066
optimistic Adam з усер.	2.58	1.073

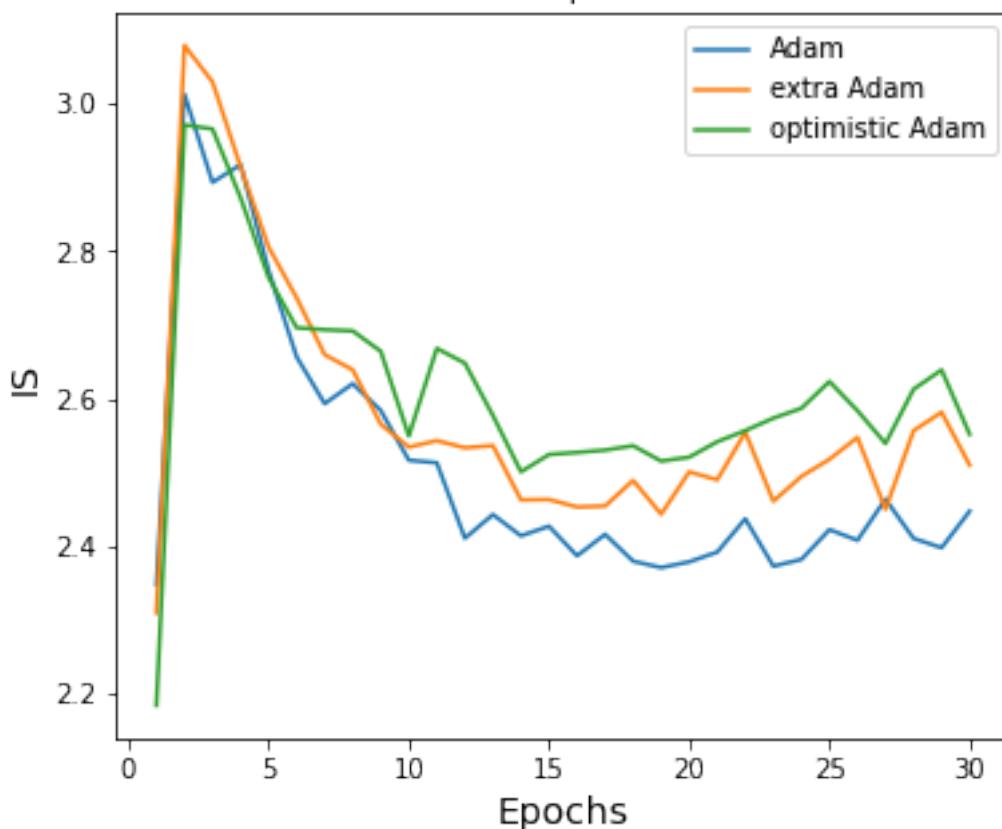
Як і минулого разу, найкращий результат показав метод оптимістичного Adam з усередненням, дуже близькі до нього результати у оптимістичного Adam без усереднення та екстраградієнтного Adam з усередненням. Звичайний же Adam, як і його варіант з усередненням, дають відчутно гірші результати за усі інші методи.

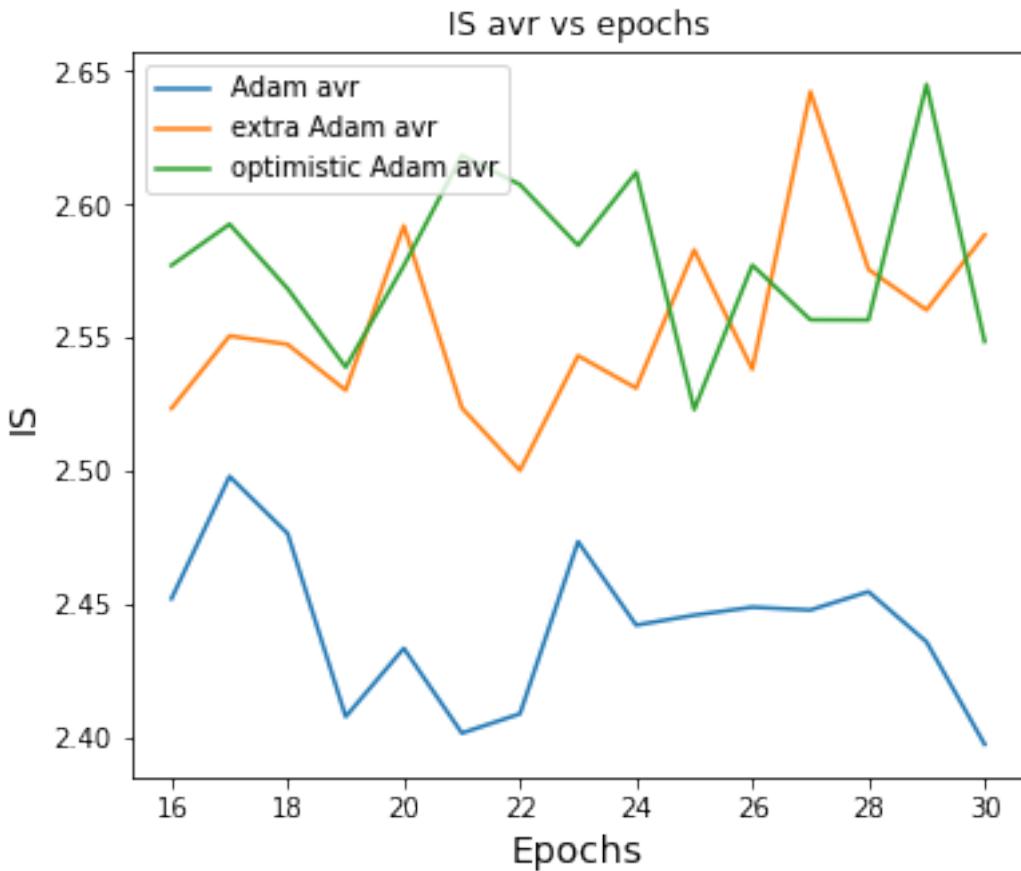
Наочанок, наведемо графіки IS-метрики для усіх оптимізаторів.

IS vs epochs



IS vs epochs





4.4 WGAN зі штрафом на градієнт

Наступною розглянемо архітектуру Wasserstein GAN зі штрафом на градієнт. Так як вона використовує зовсім інші функції помилки, можна очікувати, що і результати для наших оптимізаторів можуть відрізнятися від попередніх.

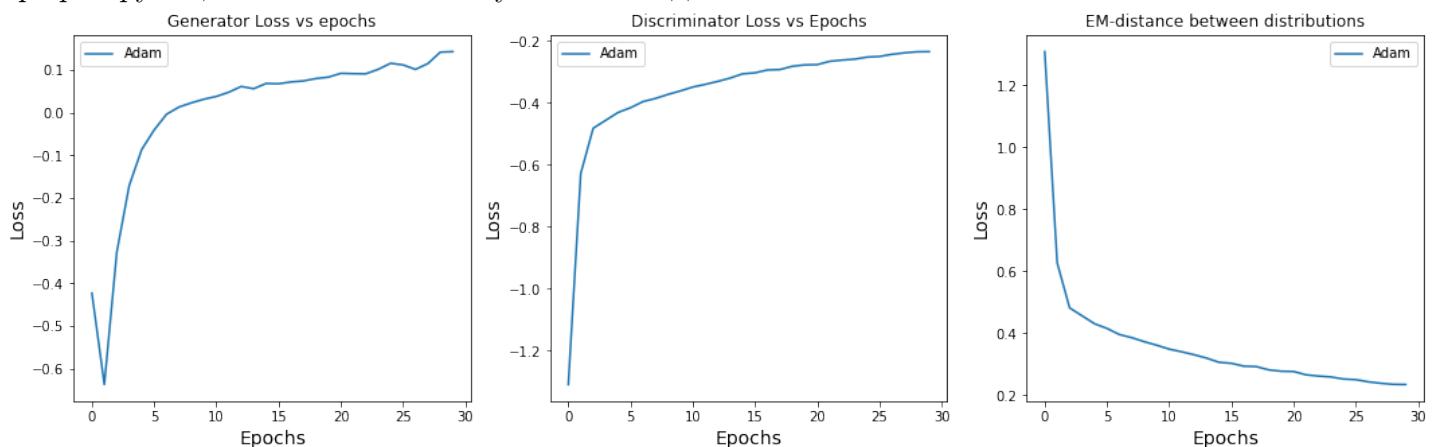
Як і для минулих моделей, для кожного оптимізатора програма була запущена по 3 рази і всі характеристики виводились як середні по 3-ом запускам. На відміну від минулих моделей, де генератор та дискримінатор тренувалися однакову кількість разів, у цій моделі на одне оновлення генератора робимо 5 оновлень дискримінатора, як рекомендується у [2].

Код до цієї частини можна знайти за посиланням:

<https://github.com/DenisPushkin/WGANGP/blob/master/WGANGP.ipynb>

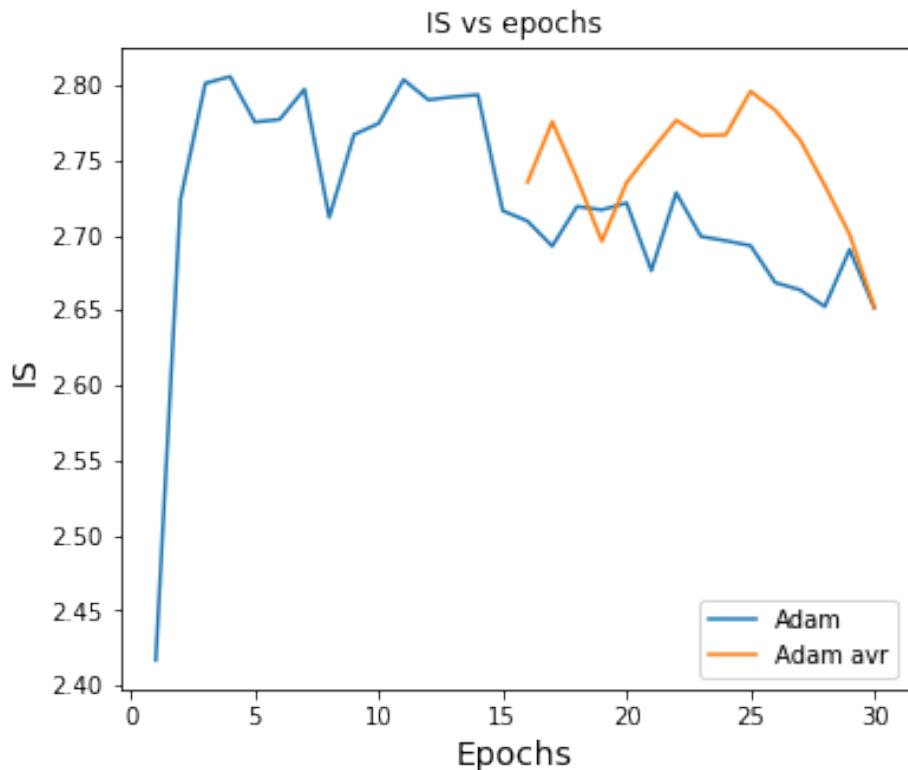
Adam

Графік функцій помилок має наступний вигляд



Нагадаємо, що для моделі WGAN функція помилки дискримінатора, взята зі знаком мінус, наближає EM-відстань (вона ж Wasserstein-1 відстань) між справжнім на згенерованим розподілом. Графік цієї відстані зображений поряд з графіками функцій помилок.

Графік IS-метрики:



Використання усереднення покращує оцінку якості в середньому на 1.98%.

Приклади згенерованих зображень після 30 епох без використання усереднення:

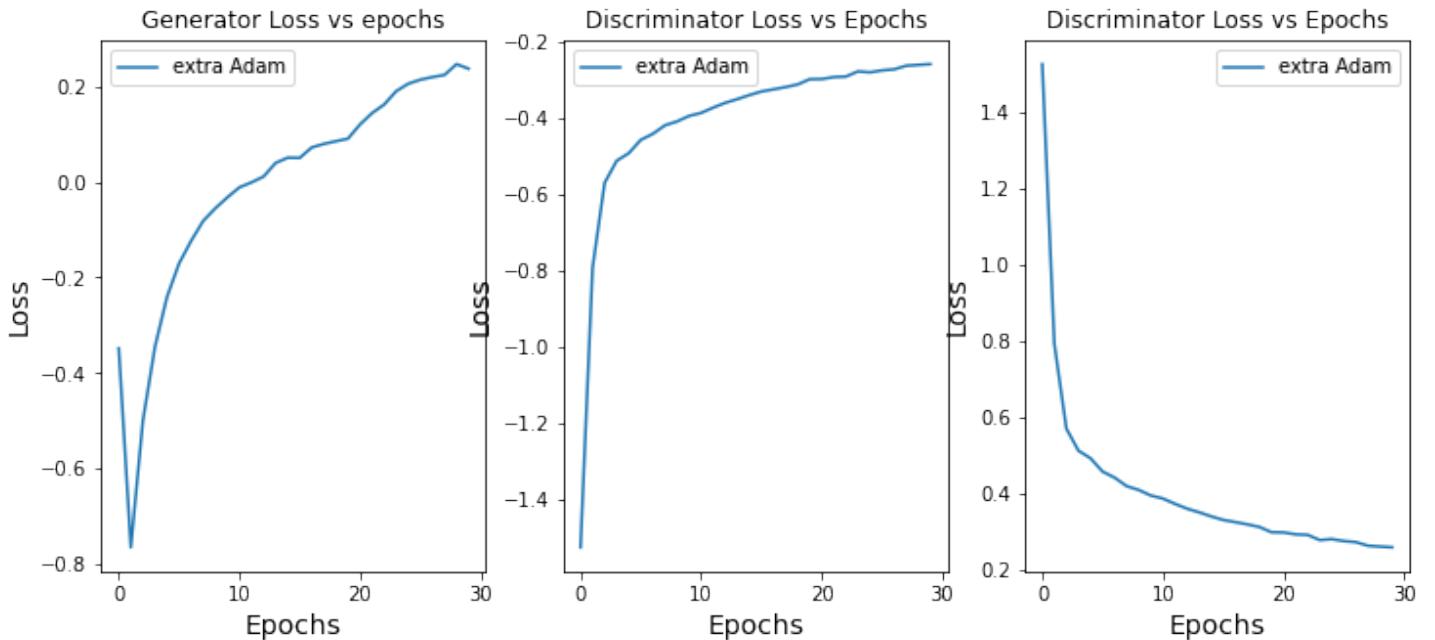
7	5	4	9	4	6
9	8	8	0	6	0
4	5	4	9	8	3
5	8	7	3	8	2
4	0	6	3	4	2
4	5	2	8	7	4
7	9	4	7	2	0
0	9	7	3	0	8
4	8	6	9	7	3
8	0	3	2	2	0
8	9	6	1	7	9
4	8	3	8	7	0
7	7	4	0	9	5
8	6	9	8	0	3
8	0	2	1	2	6
6	6	1	1	1	3
9	8	3	5	6	2
1	9	0	8	8	7
0	5	0	9	5	4
5	0	0	1	8	2
8	9	3	7	0	0
5	9	6	6	2	9
1	8	9	2	6	7
7	0	8	9	9	2
9	9	3	8	1	7
6	6	1	2	4	6
9	8	4	4	6	4
2	9	5	3	3	2
6	3	2	5	0	9
7	6	4	9	4	1

Приклади згенерованих зображень після 30 епох з використання усереднення:

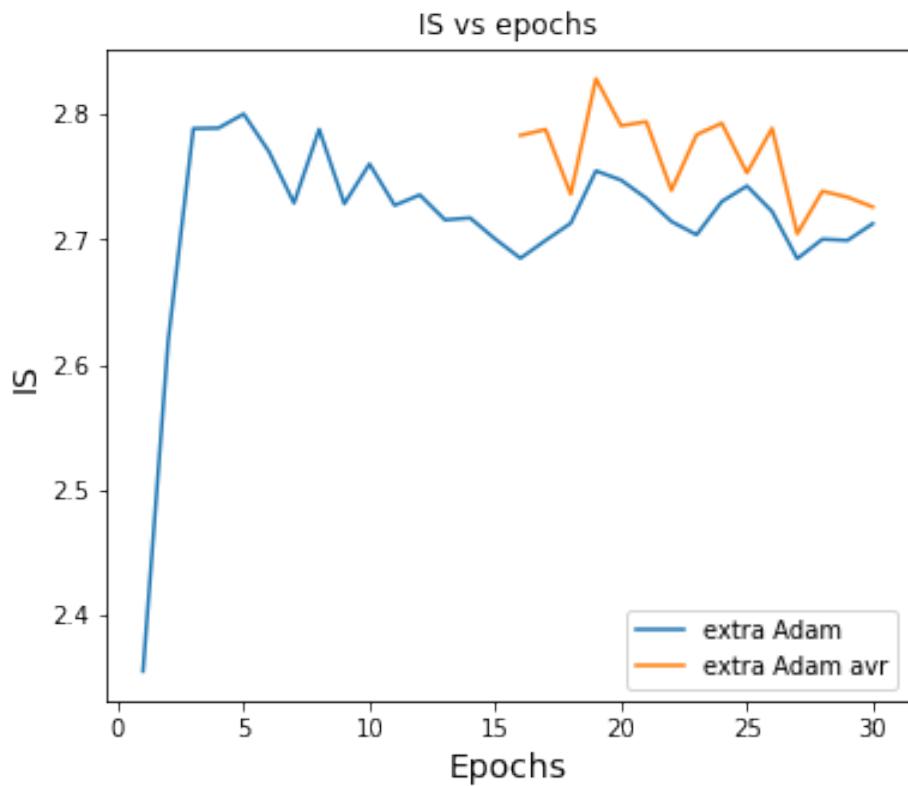
3	3	6	8	9	0
2	6	6	3	0	1
6	8	5	6	8	0
1	3	3	4	6	3
7	7	9	1	2	5
6	0	3	0	6	0
1	4	2	6	4	0
9	8	9	3	1	9
2	3	5	8	8	3
3	3	7	0	7	1
8	5	8	8	1	9
0	1	3	2	1	0
2	4	4	8	5	0
3	5	4	2	7	6
9	1	5	5	8	1
1	1	0	1	0	0
3	6	4	5	5	9
8	4	2	1	1	2
1	8	6	3	1	8
1	0	5	3	3	8
0	7	2	0	6	6
0	8	0	6	1	7
5	3	9	2	1	3
0	9	1	1	5	8
5	1	4	8	9	9
8	2	7	6	8	2
6	3	2	7	7	9
1	8	9	5	0	1
1	5	9	7	9	8
2	8	4	1	1	3

естраградієнтний Adam

Графік функцій помилок аналогічний:



Графік IS-метрики:



Використання усереднення покращує оцінку якості в середньому на 1.81%.

Приклади згенерованих зображень після 30 епох без використання усереднення:

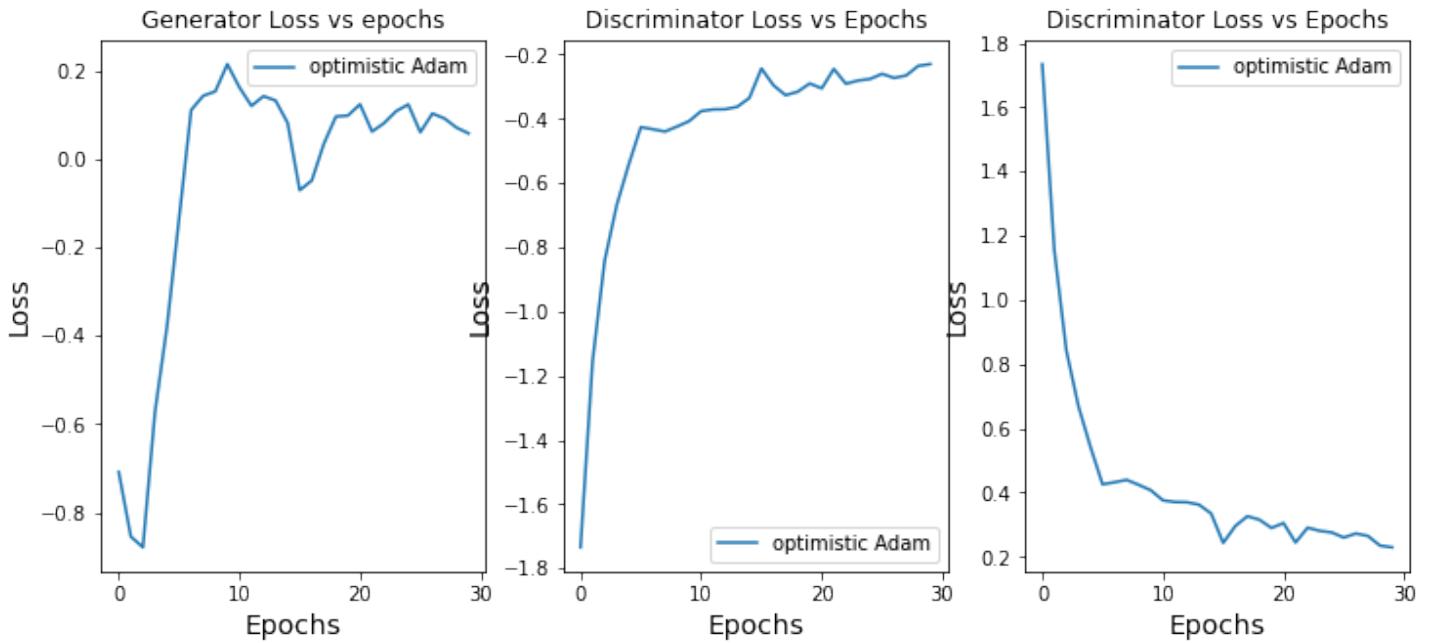
0	9	2	4	1	4
8	0	0	0	7	0
1	8	5	8	1	7
7	5	8	4	5	4
0	9	6	3	2	9
4	4	6	5	7	0
6	1	2	8	9	4
4	8	0	0	7	6
7	3	8	0	8	6
7	1	6	7	4	1
2	4	0	2	3	0
5	5	5	6	3	8
5	3	9	1	2	8
2	1	4	4	0	3
1	1	6	1	8	2
1	9	3	6	4	9
4	9	6	5	1	3
6	2	2	6	0	4
9	9	5	6	2	4
4	1	0	9	5	6
7	8	0	0	6	1
0	1	8	7	0	3
1	7	4	6	1	2
4	0	5	1	0	8
0	2	7	9	1	5
5	5	3	4	4	9
6	8	3	3	3	2
9	6	1	7	3	4
6	2	4	6	5	2
0	0	3	5	8	1

Приклади згенерованих зображень після 30 епох з використання усереднення:

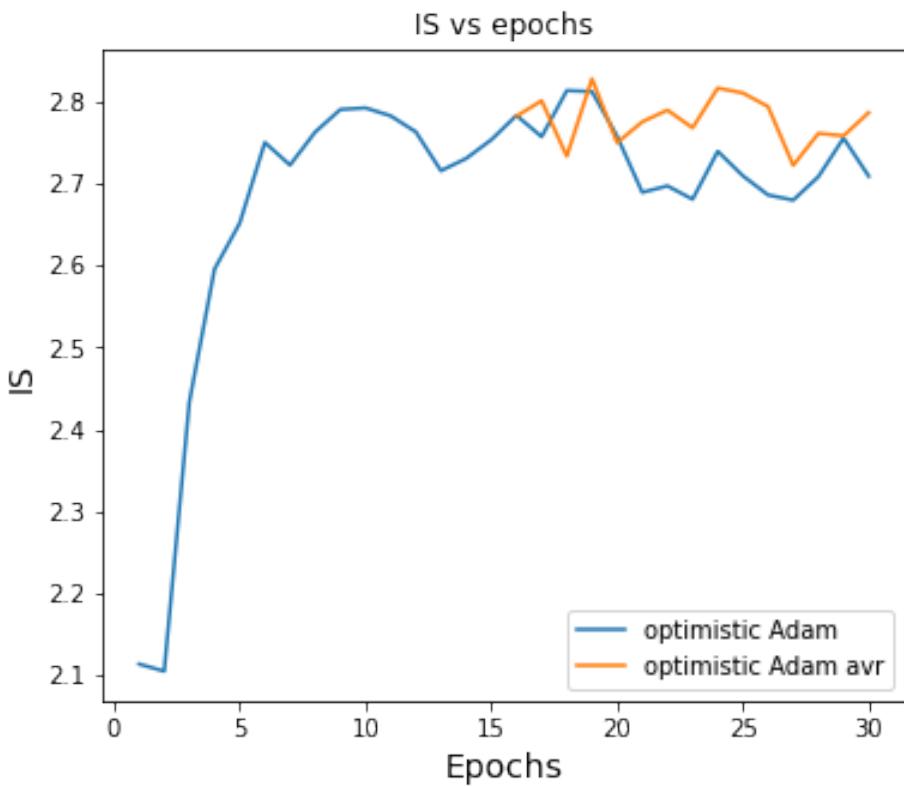
3	7	9	0	2	8
5	6	6	2	3	4
4	1	4	9	6	0
5	4	9	3	2	2
5	0	1	1	9	3
1	2	3	5	5	3
8	8	4	1	2	2
1	6	2	8	8	7
5	9	3	0	4	7
9	4	6	5	3	2
7	1	6	3	5	9
2	3	0	4	4	2
8	0	8	8	5	9
9	8	0	7	2	4
1	1	0	3	1	9
0	0	9	5	1	8
1	0	1	9	5	6
3	0	4	7	4	8
9	1	8	3	2	8
7	3	7	9	3	1
8	6	7	2	2	9
0	0	0	8	9	6
7	1	0	8	4	7
0	6	7	2	3	8
0	6	7	8	0	7
5	7	1	6	6	4
9	1	7	0	2	1
5	9	6	4	3	0
4	0	7	7	7	1
3	0	3	2	6	9

оптимістичний Adam

Графік функцій помилок має аналогічний вигляд:



Графік IS-метрики:



Використання усереднення покращує оцінку якості в середньому на 1.73%.

Приклади згенерованих зображень після 30 епох без використання усереднення:

9	0	6	9	1	2
2	2	9	1	7	2
9	3	3	9	2	1
3	3	3	0	3	3
7	1	9	9	1	2
7	5	3	5	4	7
8	7	1	1	4	5
7	4	2	8	7	6
2	1	4	8	8	7
5	0	6	8	3	1
6	4	7	0	7	6
1	8	7	0	9	5
7	9	9	3	3	9
1	0	3	1	6	2
1	3	5	8	2	5
6	2	3	7	5	4
5	4	3	7	7	5
0	7	6	1	7	6
5	0	0	3	3	7
3	4	6	9	4	6
8	3	9	2	1	1
3	0	5	9	2	3
7	4	9	1	4	5
3	9	9	1	7	2
8	2	9	3	7	8
2	7	9	5	6	3
1	2	8	9	5	9
1	4	1	2	9	4
9	7	7	2	9	4
8	8	9	7	9	9

Приклади згенерованих зображень після 30 епох з використанням усереднення:

3	9	3	4	8	5
1	1	4	8	9	1
1	8	2	0	2	9
1	6	0	3	1	8
1	5	6	1	7	0
7	1	5	6	1	4
5	7	8	9	8	6
0	9	0	6	7	6
6	9	9	1	8	3
2	2	4	3	7	3
7	9	9	8	9	3
1	7	2	1	1	6
4	3	1	9	1	8
9	6	4	9	1	3
1	7	8	3	5	1
5	9	9	8	1	1
1	5	8	3	4	2
5	1	7	4	0	6
0	7	3	9	8	7
1	6	0	9	4	6
8	8	7	8	5	6
4	8	9	3	8	1
8	2	1	3	2	3
1	4	5	1	7	6
1	9	4	6	0	1
1	8	0	1	0	1
1	3	6	3	3	9
8	7	8	6	9	6
5	7	7	1	3	0
7	7	1	9	2	5

Порівняння результатів

Як і раніше, для аналізу результатів ми усереднили IS-метрику для усіх оптимізаторів по другій половині епох (16-30). Спочатку порівняємо виграш від усереднення для усіх оптимізаторів.

	Adam	extra Adam	optimistic Adam
IS без усереднення	2.69	2.72	2.73
IS з усередненням	2.75	2.77	2.78
виграш від усереднення, %	1.98%	1.81%	1.73%

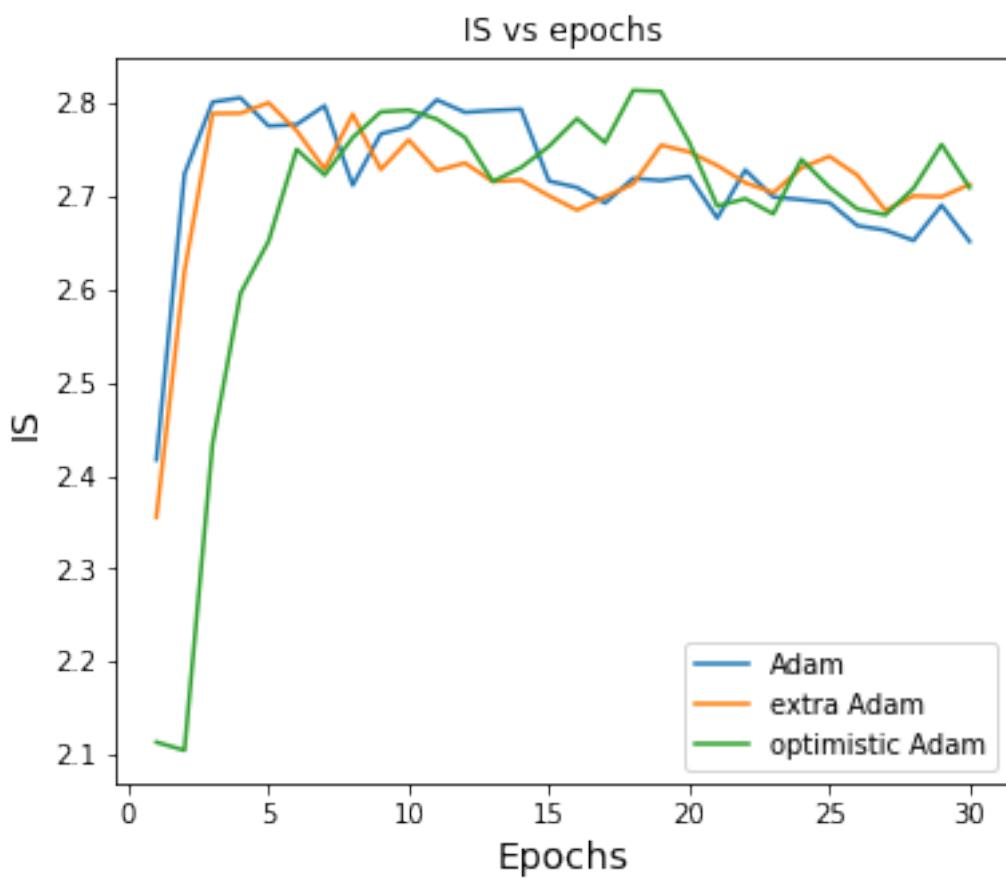
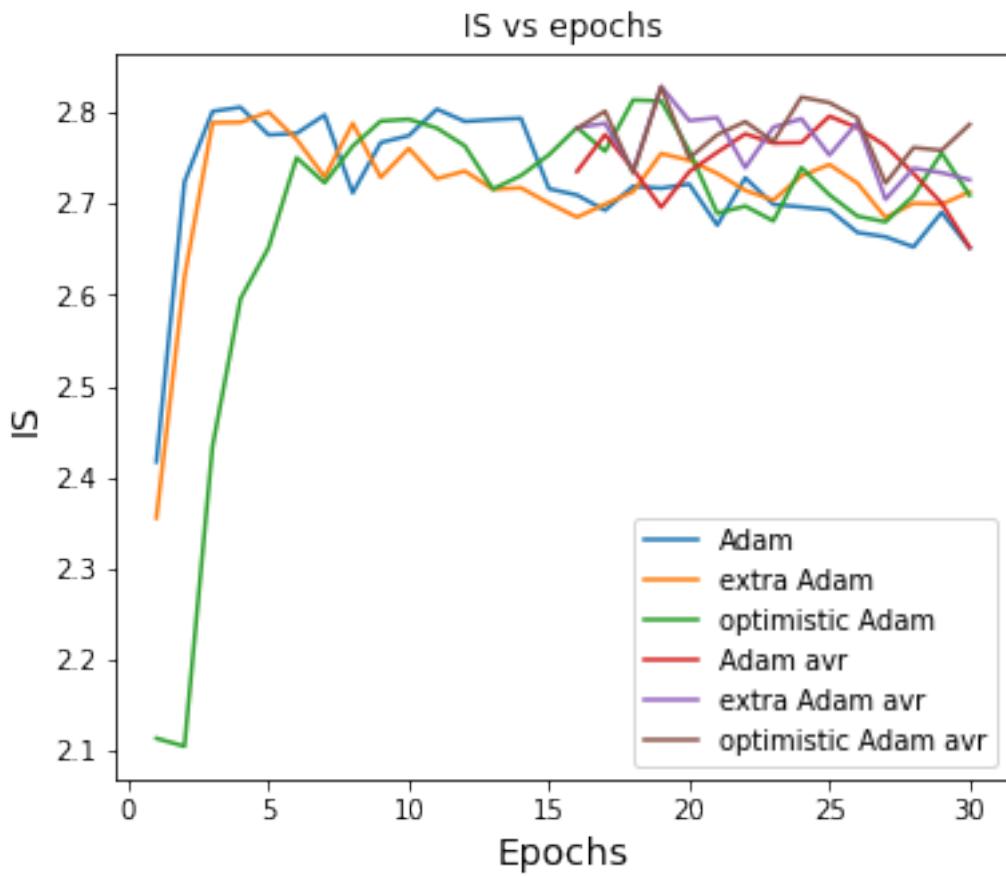
Бачимо, що хоча знаву звичайний оптимізатор Adam показав найгірші результати, тепер різниця між оптимізаторами набагато менш суттєва. Одне з можливих пояснень цього факту полягає у тому, що у минулих моделях у нас не було завдання тренувати дискримінатор майже до оптимуму: ми оновлювали генератор та дискримінатор почергово, що більше нагадувало гру між двома гравцями. У можелі ж WGAN ми робимо 5 оновлень дискримінатора перед кожним оновленням генератора, намагаючись, щоб функція помилки генератора якнайкраще наблизяла справжню ЕМ-відстань між розподілами. Тобто цей процес навчання більше схожий на звичайну мінімізацію генератором функцію помилки, а не на гру між двома гравцями, при якій перевага екстраградієнтних методів особливо відчутна.

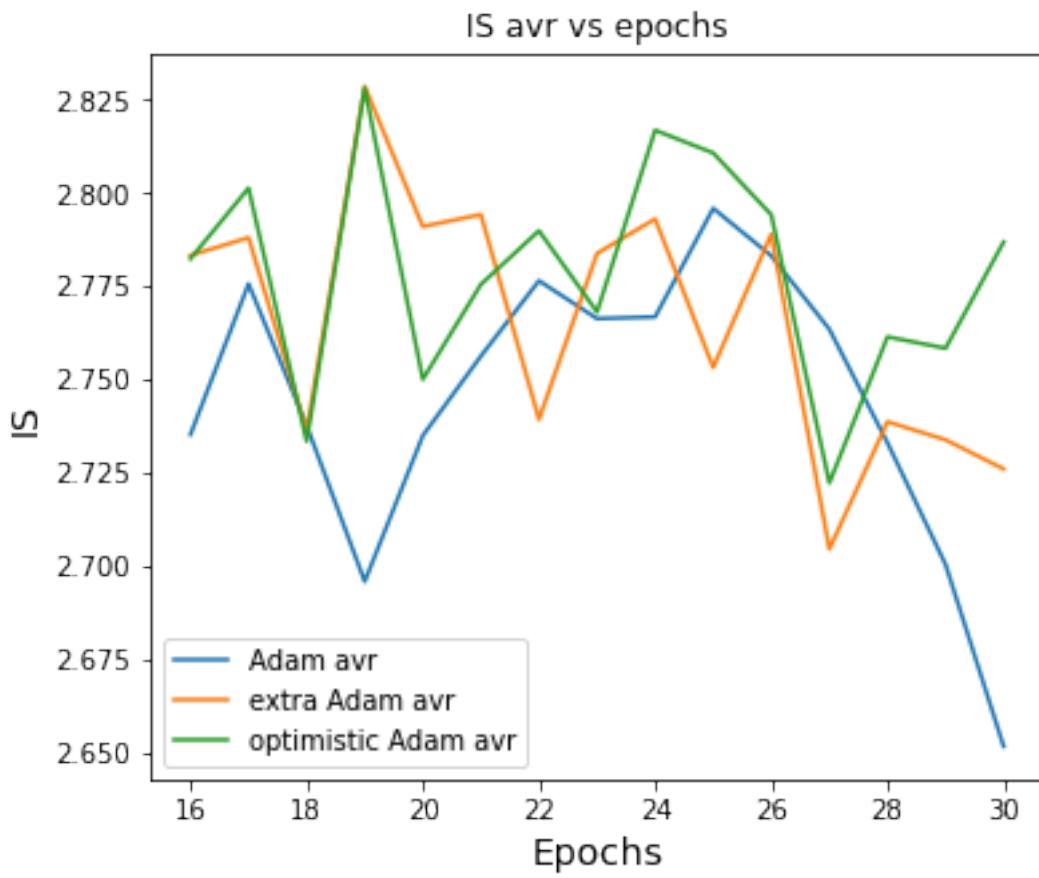
Також зазначимо, що у цьому випадку більш важливим є використання усереднення, а не вибір оптимізатора.

	результат IS	відносний результат IS
Adam	2.69	1.000
Adam з усер.	2.74	1.020
extra Adam	2.72	1.009
extra Adam з усер.	2.77	1.027
optimistic Adam	2.73	1.014
optimistic Adam з усер.	2.78	1.032

Втім, бачимо, що як і у минулих моделях, найгірший результат дає звичайний Adam, а найкращий - усереднений оптимістичний Adam.

Наочанок, наведемо графіки IS-метрики для усіх оптимізаторів.





5 Висновки

Було зроблено порівняння найбільш уживаного для нейронних мереж адаптивного оптимізатора Adam та його варіацій оптимістичний Adam та екстраградієнтний Adam для трьох моделей: умовний GAN (CGAN), DCGAN та WGAN зі штрафом на градієнт. Помічено, що для усіх цих моделей використання екстраградієнтних методів покращує якість генерованих зображень, а використання усереднених значень змінних по другій половині навчання майже завжди ще більше покращує якість зображень.

6 Список використаної літератури

- [1] Ian J. Goodfellow, Jean Pouget-Abadiey, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozairz, Aaron Courville, Yoshua Bengio. Generative Adversarial Nets (2014). arXiv:1406.2661
- [2] Martin Arjovsky, Soumith Chintala, and Leon Bottou. Wasserstein GAN (2017). arXiv:1701.07875v3
- [3] Cedric Villani. Optimal Transport: Old and New. Grundlehren der mathematischen Wissenschaften. Springer, Berlin, 2009.
- [4] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville. Improved Training of Wasserstein GANs (2017). arXiv:1704.00028v3
- [5] Alec Radford, Luke Metz, Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks (2016). arXiv:1511.06434v2.
- [6] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, Jose M. Álvarez. Invertible Conditional GANs for image editing (2016). arXiv:1611.06355v1
- [7] Ляшко С. І., Семенов В. В., Клюшин Д. А. Спеціальні питання оптимізації. Київ: ВПЦ "Київський університет". 2015. 183 с.