Neural ODE studying

Ilya Lopatin
lopatin.ia@phystech.edu

Daniil Merkulov
daniil.merkulov@skoltech.ru

Project Proposal

Проект посвящен изучению принципов работы нейросетевой архитектуры Neural ODE, представленной в работе [1]. Планируется проведение различных численных экспериментов с данной архитектурой и последующие объяснение и интерпретация полученных результатов, изучение возможных приложений Neural ODE, таких как: восстановление функции динамики по зашумленным наблюдениям, непрерывные нормализующие потоки, генеративные функции для моделирования временных рядов.

1 Описание

1.1 Идея нейросети как обыкновенного дифференциального уравнения

В этом разделе изложена основная идея построения архитектуры Neural ODE, которая впервые была представлена в работе [1].

Опишем как архитектура *Residual neural network* может быть связана или интерпретирована в терминах обыкновенных дифференциальных уравнений (более подробно см. [1, 2, 3]). Рассмотрим общий вид (1) *skip-connection* слоя в остаточной нейронной сети:

$$z(t+1) = z(t) + f_t(z(t), \theta), \tag{1}$$

где $t \in \mathbb{N}_0$ – дискретный параметр, соответствующий номеру слоя, f_t – передаточная функция узла с номером t ($f : \mathbb{R}^d \to \mathbb{R}^d$, где d – размерность z), θ – множество, «обучаемых» параметров нейронной сети.Наряду с этим, рассмотрим задачу Коши (2) для обыкновенного дифференциального уравнения:

$$\begin{cases} \dfrac{dz(t)}{dt} = f(z(t), t, \theta) \\ t \in [t_1; t_2] \\ z(t_1) = x \end{cases} \tag{2}$$

и явный метод Эйлера (3) численного решения (2):

$$z(t + \Delta t) = z(t) + \Delta t f(z(t), t, \theta), \; z(t_1) = x, \tag{3}$$

где Δt – параметр численной схемы, соответствующий длине шага сетки численного решения. Заметим, что при $\Delta t = 1$ (1) и (3) полностью совпадают. Основная идея состоит в том, чтобы интерпретировать нейронную сеть как задачу Коши (2). Оказывается, что при таком подходе процедуры прямого и обратного распространения можно представить в виде численного решения задачи Коши для ОДУ.

1.2 Прямое и обратное распространение

Прямое распространение представляет численное решение (2), отметим, что решение необязательно проводить именно методом Эйлера, здесь применимы методы более высокого порядка и устойчивости, например методы Рунге-Кутты (для справки по численным методам см., например, [4]). Вообще, здесь можно отметить, что, если рассматривать т.н. *dense neural network*, сигнал в которых на k-том слое явно зависит не только от сигнала на $k-1$–слое, но и предыдущих, иначе говоря в формулу (1) явно входят $z(t-1), z(t-2), \ldots, z(t-k)$, то аналогия между прохождением сигнала в таких нейросетях и маршевыми методами Рунге-Кутты становится еще сильнее.

Для алгоритма обратного распространения в работе [1] вводится следующая функция (*adjoint function*[1]):

$$a(t) = \frac{\partial L(y)}{\partial z(t)},$$

где $L(y)$ – функция потерь (*loss–function*), а y – выход нейросети (в нашем случае $y = z(t_2)$).

В [1] показано, что $a(t)$ есть решение следующей задачи Коши (4) (заметим, что условие задано на правом крае, а не на левом, как обычно):

$$\begin{cases} \dfrac{da(t)}{dt} = -a^T(t)\dfrac{\partial f}{z(t)} \\ a(t_2) = \dfrac{\partial L}{\partial y} \end{cases} \tag{4}$$

А градиент по обучаемым параметрам выражается как:

$$\nabla_\theta L = \int_{t_1}^{t_2} a^T(t)\frac{\partial f}{\partial \theta}\left(z(t), t, \theta\right) dt, \tag{5}$$

где подынтегральное выражение представляет Ниже представлен алгоритм из [1], подсчета всех необходимых градиентов.

Algorithm 1 Reverse-mode derivative of an ODE initial value problem

Input: dynamics parameters θ, start time t_0, stop time t_1, final state $\mathbf{z}(t_1)$, loss gradient $\partial L / \partial \mathbf{z}(t_1)$

$s_0 = [\mathbf{z}(t_1), \frac{\partial L}{\partial \mathbf{z}(t_1)}, \mathbf{0}_{|\theta|}]$ ▷ Define initial augmented state

def aug_dynamics($[\mathbf{z}(t), \mathbf{a}(t), \cdot], t, \theta$): ▷ Define dynamics on augmented state

  return $[f(\mathbf{z}(t), t, \theta), -\mathbf{a}(t)^\intercal \frac{\partial f}{\partial \mathbf{z}}, -\mathbf{a}(t)^\intercal \frac{\partial f}{\partial \theta}]$ ▷ Compute vector-Jacobian products

$[\mathbf{z}(t_0), \frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}] = \text{ODESolve}(s_0, \text{aug\_dynamics}, t_1, t_0, \theta)$ ▷ Solve reverse-time ODE

return $\frac{\partial L}{\partial \mathbf{z}(t_0)}, \frac{\partial L}{\partial \theta}$ ▷ Return gradients

где f_{aug} – расширенная или *аугментированная* функция динамики:

$$\frac{d}{dt}\begin{bmatrix} z \\ \theta \\ t \end{bmatrix}(t) = f_{aug}\left([z, \theta, t]\right) = \begin{bmatrix} f\left([z, \theta, t]\right) \\ 0 \\ 1 \end{bmatrix}$$

1.3 Нормализующие потоки

В работе [6] показано, что для генеративно-состязательных нейросетей (*GAN*) фактически неизбежен т.н. *Mode Collapse* – особенность обучения GAN-ов – нейросеть обучается генерировать изображение или иной выход на основе только лишь части своей обучающей выборки. Один из возможных способов избежать Mode Collapse это подсчитывать плотность распределения, порожденного генератором GAN и регуляризация каким-либо способом, если плотность становится сильно неравномерной, подробное изложение см. в [7]. Однако этот способ имеет следующую вычислительную трудность: рассмотрим моделирование плотности сложного распределения в высокоразмерном пространстве как прохождение распределения $z(0) \sim p_0(z_0)$ через нелинейное преобразование – нейросеть с T слоями: $z(T) \sim p_T(z(T))$, тогда есть возможность посчитать плотность трансформированного распределения по формуле (6)

$$\log p_T\left(z(T)\right) = \log p_0\left(z(0)\right) - \sum_{t=1}^{T}\left|\frac{df_t}{df_{t-1}}\right|, \tag{6}$$

где $z(T) = f(z(0), \theta)$ – прямое распространение в нейронной сети, а $|df_t/df_{t-1}|$ – якобиан между соответствующими слоями, вычисление которого есть куб операций от размерности пространства $\Theta(d^3)$, что делает формулу (6) неприменимой на практике в общем случае для высокоразмерных пространств.

[1] вообще говоря, понятие сопряженного решения для ОДУ известно достаточно давно и было введено Л.С. Понтрягиным, см. [5]

Здесь оказывается полезным следующие утверждение:

Пусть $z(t)$ – конечная непрерывная случайная величина с вероятностью $p(z(t))$, зависящей от времени. Пусть $dz/dt = f(z(t), t)$ – дифференциальное уравнение, описывающее непрерывное во времени преобразование $z(t)$. Предполагая, что f удовлетворяет условию Липшица по аргументу z и непрерывна по аргументу t, имеем следующее соотношение (7):

$$\frac{\partial \log p\left(z(t)\right)}{\partial t} = -\operatorname{tr}\left(\frac{\partial f}{\partial z(t)}\right) \tag{7}$$

1.4 Преимущества и приложения

Отметим несколько преимуществ данной архитектуры. Варьируя шаг численного решения ОДУ, мы как бы меняем число слоев в нейросети, т.е. архитектура Neureal ODE дает возможность фактически в режиме реального времени производить трейд-ин между точностью и временем работы. Кроме того, заметим, что ввиду метода подсчета градиента нету необходимости хранить все результаты прохождения в скрытых слоях, т.е. процедура градиентного спуска в данной архитектуре константна по памяти относительно количества слоев или мелкости сетки численного решения. Более подробно: численно решается (4), если решение происходит методом Рунге-Кутты, то необходимо помнить только фиксированное число предыдущих значений, полученные численные значения можно сразу передавать в (5) для подсчета $\nabla_\theta L$.

Поговорим о возможных приложениях:

- Замена Res-блока на блок NeuralODE с целью указанных выше преимуществах.
- Восстановление функции динамики. Пусть у нас есть какая-то динамическая система с законом типа $z'(t) = f(z(t), t)$, где f – неизвестная функция динамики и есть серия наблюдений точек траектории. Тогда можно попытаться восстановить эту функцию динамики, надлежащем образом обучив NeuralODE.
- Непрерывные Нормализующие Потоки. Семплирование плотности распределения какого-то сложного процесса может помочь например в регуляризации обучения нейронных сетей архитектуры GAN.

2 Problem

Итак, мы имеем достаточно новую архитектуру (работа [1] вышла 2018 году) нейронной сети. Целью этого проекта является экспериментальное изучение следующих вопросов:

- Действительно ли NeuralODE имеет все указанные выше преимущества?
- Как NeuralODE ведет себя в плане точности распознавания / времени обучения, требуемой памяти по сравнению со стандартными остаточными нейросетями?
- Как точно NeuralODE может восстановить неизвестные динамики по серии наблюдений и сколько временных ресурсов и памяти на это нужно?
- Как точно NeuralODE может семплировать плотность распределения сложных процессов? Какие могут быть ограничения на целевую плотность?

Отметим, что, теоретически можно указать «слабые места» данной архитектуры. Что если скрытая целевая функция динамики плохо обусловленна возможно ли в таких случаях корректно обучить нейросеть? При выводе формулы (7) берется бесконечно малое приращение аргументов, мы же, хоть и можем варьировать шаг решения, не можем делать его «слишком малым» ввиду, например, ресурсных ограничений, применима ли формула (7) для реальных вычислений, получим ли мы действительно выигрыш или лучше вернуться к подходу, основанному на формуле (6)?

3 Outcomes

В качестве выхода планируется репозиторий на github с авторской реализацией NeuralODE из статьи [1], плюс результаты экспериментов, проведенных с этой архитектурой. Быть может, будет сделан отчет в формате статьи / доклада с описанием и интерпретированием полученных результатов.

Эксперименты планируется проводить для ответов на поставленные вопросы в разделе «Problem». Ниже ориентировочный список основных сценариев:

- Запуск и сравнение результатов (точность / время / память) с другими известными нейросетями. Например, задача классификации с обучающей выборкой MNIST.
- Задание несколько различных функций динамик и отслеживание, как Neural ODE будет обучаться/восстанавливать их по множеству точек.
- Проверка (желательно даже визуализация процесса), как Neural ODE будет моделировать заранее заданную плотность распределения.

4 Литературный обзор

Работа [1] есть основная в моем проекте, в ней была впервые построена архитектура NeuralODE. Работы [2], [3] являются фундаментом для работы [1], в них были описаны первые аналогии между ResNet и дифференциальными уравнениями.Хочется отметить [5] в этой работе группа советских математиков во главе с Л.С. Понтрягиным впервые ввела технику сопряженных функций, вывод формулы (5) идейно вдохновлен этой работой. Работа [4] содержит описания существующих численных метод решения ОДУ, в частности особо интересные для этой работы методы Рунге-Кутты. В работе [7] был предложен метод моделирования сложной плотности с помощью нормализующих потоков. В работе [6] подробно изложена проблема Mode Collapse для генеративно-состязательных нейронных сетей. В работе [8] рассматриваются методы компьютерного подсчета выражений типа (4),(5) содержащие пространственные якобианы больших размерностей.

5 Метрики качества

В качестве метрик качества можно рассмотреть содержания планируемого репозитория: количество, широта и результаты проведенных экспериментов по применению Neural ODE. Под широтой подразумевается:

- планируется провести несколько серий экспериментов для проверки различных способов применения Neural ODE (см. раздел «Преимущества и приложения»), в идеале планируется проверить все указанные применения с различными типами входных данных.
- на каждый вопрос, поставленный в разделе Problem, можно будет дать ответ, подкрепленный соответствующими тестами.

Качество проделанной работы можно будет оценить по результат экспериментов, как точно они согласуются с теоретическими предсказаниями и насколько хорошо их можно объяснить.

6 Примерный план

- Сначала, конечно, необходимо ознакомиться с теоретическим аспектом работы NeuralODE: прочитать оригинальную статью и, быть может, пару других постов/обзорных статей, благо материала много. Т.к. теоретический материал мне знаком в достаточной степени хорошо я выделю на это сравнительно малый срок: до 22 апреля (т.е. до *draft 1*). Особое внимание я уделю нормализующим потокам т.к. на данный момент это самая сложная для меня теоретическая тема в данном проекте.
- Потом планирую установить и запустить оригинальный репозиторий и разобрать вложенные в него примеры. В этом пункте я подразумеваю просто "технический"запуск кодов, поэтому выделю на это время до 25 апреля.
- До 29 апреля (т.е. draft 2) планируется провести первые собственные эксперименты, чтобы понять "что к чему". Планируется начать с экспериментов по восстановлению скрытых функций динамик.
- К 6 мая планируется расширить число экспериментов, провести серию экспериментов в сфере нормализующих потоков. Быть может, реализовать некоторую часть репозитория самостоятельно, это может быть например реализация «с нуля» какого-то одно приложения, на данный момент, самым простым кажется восстановление функции динамики.

- После 6 мая, по согласованию с семинаристом планируется расширение проекта в какую-то из сторон, указанных в пункте выше.

Список литературы

[1] Ricky T., Q. Chen, Yulia Rubanova, Jesse Bettencourt. *Neural Ordinary Differential Equations.* University of Toronto, Vector Institute, 2018.

[2] Brandon Amos and J Zico Kolter. *OptNet: Differentiable optimization as a layer in neural networks.* International Conference on Machine Learning, pages 136–175, 2017.

[3] Bo Chang, Lili Meng, Eldad Haber, Lars Ruthotto, David Begert, and Elliot Holtham. *Reversible architectures for arbitrarily deep residual neural networks.* arXiv preprint arXiv:1709.03698.

[4] Ascher, Uri M.; Petzold, Linda R. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations.* Society for Industrial and Applied Mathematics, 1998.

[5] Lev Semenovich Pontryagin, EF Mishchenko, VG Boltyanskii, and RV Gamkrelidze. *The mathematical theory of optimal processes*, 1962.

[6] Hoang Thanh-Tung, Truyen Tran. *On Catastrophic Forgetting and Mode Collapse in Generative Adversarial Networks.* Applied Artificial Intelligence Institute Deakin University, 2018

[7] Danilo Jimenez Rezende and Shakir Mohamed. *Variational inference with normalizing flows.* arXiv preprint arXiv:1505.05770, 2015.

[8] Joel A E Andersson and Joris Gillis and Greg Horn and James B Rawlings and Moritz Diehl. *CasADi – A software framework for nonlinear optimization and optimal control.* Mathematical Programming Computation, 11(1):1–36, 2019.