Нейродифференциальные уравнения

Аннотация

Данная работа является обзором статьи «Neural Ordinary Differential Equations» [1], удостоенной награды за лучшую статью на конференции NeurIPS 2018. В обзоре будет описано, что такое нейродифференциальное уравнение, как его обучать, какие преимущества имеет данная модель по сравнению с обычными нейронными сетями и где может применяться на практике. Также будут приведены результаты авторских и оригинальных экспериментов.

1 Введение: от ResNet к дифференциальным уравнениям

Известно, что одной из наиболее успешных и прорывных идей в области глубинного обучения за последние несколько лет является создание архитектуры остаточных сетей (Residual Networks, ResNets) [2]. В 2015 году исследователи из Microsoft предложили довольно простую модификацию обычных нейросетевых архитектур: добавление так называемых skip-connections, которые позволяют проходить входному сигналу через сеть, минуя нелинейные преобразования слоёв.

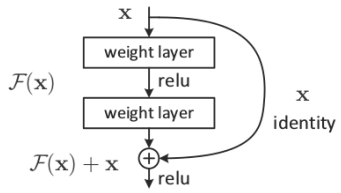

Рис. 1: Блок со skip-connection

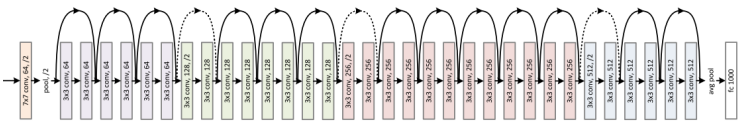

Рис. 2: Архитектура ResNet-34

Более формально, если $z(t) \in \mathbb{R}^D$ — вход слоя нейросети с номером t, $f_t(z, \theta) : \mathbb{R}^D \to \mathbb{R}^D$ — преобразование слоя t, то выход данного слоя $z(t+1)$ рассчитывается следующим образом:

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

Несмотря на всю простоту идеи, данная архитектура позволила успешно обучать очень глубокие нейронные сети. Одной из основных проблем обучения нейросетевых архитектур с большим количеством слоев является проблема затухающих

градиентов: ситуация, когда в ходе обучения градиент практически не доходит от выхода к входу, поскольку становится очень близок к 0 в ходе алгоритма обратного распространения ошибки. Добавление skip-connections в какой-то степени позволяет избавиться от данной проблемы, поскольку у градиента появляется возможность проходить не только через нелинейное преобразование слоя, но также и через тождественное преобразование.

Немного отвлечёмся от нейронных сетей и рассмотрим обыкновенное дифференциальное уравнение первого порядка с начальным условием:

$$\begin{cases} \dfrac{\mathrm{d}z(t)}{\mathrm{d}t} = f(z(t), t, \theta) \\ z(0) = x, t \in [0, T] \end{cases} \tag{2}$$

Необходимо найти значение функции $z(t)$ в момент времени $t_1 \in [0, T]$:

$$z(t_1) = z(0) + \int_0^{t_1} f(z(t), t, \theta) dt \tag{3}$$

Вычисление $z(t_1)$ требует интегрирования производной $z(t)$ по времени и не всегда может быть произведено аналитически.

Существуют различные численные методы решения данного дифференциального уравнения, простейшим из которых является метод Эйлера. В данном методе отрезок $[0, T]$ разбивается равномерной сеткой с длиной шага Δt. Значение приближенного решения в следующем узле задаётся небольшим шагом в сторону градиента функции $z(t)$ из текущего узла. Более формально, приближенное решение определяется по формуле:

$$z(t + \Delta t) = z(t) + \Delta t f(z(t), t, \theta) \tag{4}$$

Отметим крайнюю схожесть выражения (4) в методе Эйлера с выражением (1) для вычисления выхода блока со skip-connection в архитектуре ResNet. В частности, если в методе Эйлера положить $\Delta t = 1$, то упомянутые выражения и вовсе совпадают. Таким образом, между остаточными сетями и дифференциальными уравнениями типа (2) может быть проведена прямая аналогия: вычисление выхода блока остаточной сети может рассматриваться как одна итерация метода Эйлера для решения дифференциального уравнения.

2 Нейродифференциальные уравнения

Известно, что метод Эйлера имеет ряд существенных недостатков. Так, например, данный метод имеет большую погрешность, вычислительно неустойчив и накапливает ошибку на каждом шаге. Однако в настоящее время существуют более сложные численные методы решения дифференциальных уравнений типа (2), во многом лишённые этих недостатков [3] [4] [5].

Таким образом, возникает простая идея: вместо численной схемы (4) рассматривать непосредственно само дифференциальное уравнение (2) и обучать его параметры. В качестве функции $f(z(t), t, \theta)$, определяющую динамику функции $z(t)$, положим нейронную сеть. Стоит отметить, что, в отличие от обычных нейронных сетей, параметры θ являются общими для всех моментов времени. Математически это означает, что $\frac{\mathrm{d}\theta(t)}{\mathrm{d}t} = 0$.

Имея начальное условие $z(0) = x$, где x — входные данные модели (например, изображение, которое нужно классифицировать), определим значение в конечный момент времени $y = z(T)$ в качестве выхода модели нейродифференциального уравнения. Это значение может быть вычислено с помощью какого-либо численного метода решения дифференциальных уравнений (не обязательно Эйлера!).

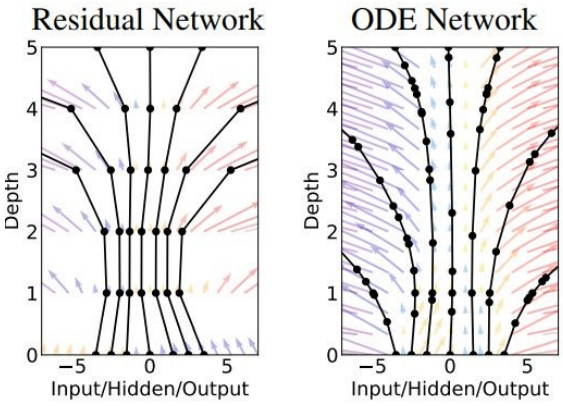

Рис. 3: Сравнение модели ResNet и нейродифференциального уравнения

Рис. 3 демонстрирует принципиальные отличия модели ResNet от модели нейродифференциального уравнения. Как известно, остаточная сеть определяет дискретную последовательность преобразований. Напротив, как можно видеть на рисунке справа, нейродифференциальное уравнение определяет непрерывное во времени преобразование входов модели к выходам. Точкам на обоих рисунках соответствуют моменты времени $t \in [0, T]$, соответствующие вычислениям промежуточных значений $z(t)$.

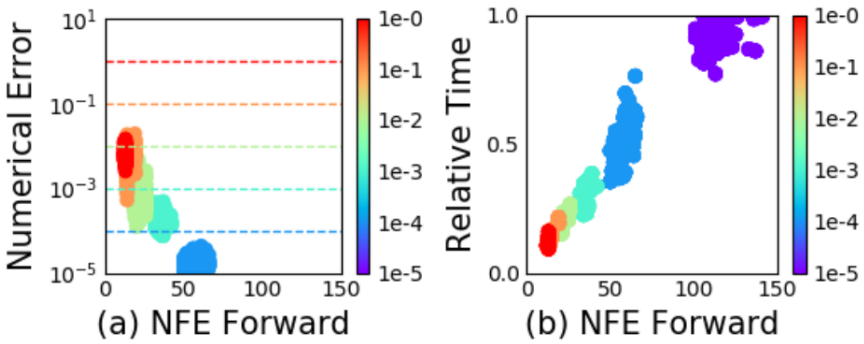

Рис. 4: (a) Численная ошибка уменьшается с увеличением количества промежуточных вычислений; (b) Время работы растёт с увеличением количества промежуточных вычислений.
(NFE - number of function evaluations)

Модель нейродифференциального уравнения имеет ряд преимуществ по сравнению с обычными нейронными сетями:

- **Эффективное по памяти обучение**. В разделе 3 будет показано, как можно обучать модель нейродифференциального уравнения, не пробрасывая градиенты через все операции численного метода. Отсутствие необходимости хранить промежуточные значения, вычисленные в ходе прямого прохода, позволяет обучать модель с независящими от «глубины» затратами по памяти. Под «глубиной» понимается количество вычислений функции в промежуточных точках.

- **Адаптивное время работы**. Современные численные методы решения дифференциальных уравнений позволяют явно указывать желаемую точность численного решения, подбирая требуемую дискретность сетки самостоятельно (Рис. 4 a). Естественно, с измельчением сетки возрастает и время работы метода (Рис. 4 b). Таким образом, существует явный способ контролировать trade-off между численной точностью решения и вычислительными затратами. Данное свойство метода позволяет обучать модель с высокой точностью, а во время тестирования намеренно понижать её, тем самым сокращая объём требуемых вычислительных ресурсов. Это может быть полезно, например, для запуска модели на устройстве с низкой вычислительной мощностью (например, телефоне).

- **Непрерывные нормализационные потоки**. Идея нейродифференциальных уравнений, примененная к модели нормализационных потоков [6], породила новый класс моделей, который авторы статьи назвали «Непрерывные нормализационные потоки». Данная модель имеет ряд преимуществ по сравнению со своим дискретным аналогом, о которых будет подробнее рассказано в разделе 5.

3 Обучение модели нейродифференциального уравнения: непрерывный аналог алгоритма обратного распространения ошибки

Каким же образом можно обучать модель нейродифференциального уравнения? Как обычно, обучение модели соответствует минимизации функции потерь L по параметрам θ:

$$L(y) = L(z(T)) = L\Big(z(0) + \int_0^T f(z(t), t, \theta)dt\Big) = L(ODESolve(z(0), f, 0, T, \theta)) \to \min_{\theta}, \tag{5}$$

где $ODESolve$ — численный метод решения дифференциальных уравнений. Для возможности обучения модели стандартными градиентными методами необходимо уметь вычислять градиент функции потерь по параметрам $\frac{\partial L}{\partial \theta}$.

Теоретически градиент может быть вычислен «в лоб» с помощью алгоритма обратного распространения ошибки, поскольку обычно операции численного метода $ODESolve$ дифференцируемы. Тем не менее, у данного подхода есть ряд существенных недостатков. Во-первых, это высокие затраты по памяти, обусловленные необходимостью хранить промежуточные результаты вычислений численного метода. Во-вторых, данный подход влечёт за собой дополнительные численные ошибки, поскольку подразумевает дифференцирование аппроксимации, а не аппроксимацию «честного» градиента.

Альтернативный подход вычисления искомого градиента, впервые рассмотренный в работах советского математика Л. В. Понтрягина [7], основывается на понятии сопряженной функции. Данный метод не требует информации о внутреннем

устройстве применяемого численного метода, трактуя его как чёрный ящик. Подход вычисляет градиенты с помощью ещё одного дифференциального уравнения, решая его обратно во времени. Необходимо отметить, что для простоты нотации все выкладки будут проводиться для одномерного случая, однако они легко обобщаются и на случай многих переменных.

Итак, зададим сопряженную функцию $a(t)$, положив по определению $a(t) = \frac{\partial L}{\partial z(t)}$. Данная функция имеет следующую интуитивную интерпретацию: насколько изменится значение функции потерь L, если немного изменить значение z в момент времени t. Изменив значение z в момент времени t, мы тем самым изменим траекторию, поэтому изменится и конечная точка $z(T)$, а, следовательно, и значение функции потерь $L(z(T), \theta)$.

Покажем, что динамика сопряженной функции задаётся следующим дифференциальным уравнением с граничным условием:

$$\begin{cases} \dfrac{\partial a(t)}{\partial t} = -a(t)\dfrac{\partial f(z(t), t, \theta)}{\partial z(t)} \\ a(T) = \dfrac{\partial L}{\partial z(T)}, t \in [0, T] \end{cases} \tag{6}$$

Доказав, что имеет место (6), у нас появится возможность находить значение сопряженной функции $a(t)$ в любой точке отрезка $[0, T]$ с помощью численных методов.

По правилу дифференцирования сложной функции для $\varepsilon > 0$ справедливо

$$\frac{\partial L}{\partial z(t)} = \frac{\partial L}{\partial z(t+\varepsilon)}\frac{\partial z(t+\varepsilon)}{\partial z(t)} + \frac{\partial L}{\partial \theta(t+\varepsilon)}\frac{\partial \theta(t+\varepsilon)}{\partial z(t)} \tag{7}$$

Поскольку $\theta(t+\varepsilon) = \theta(t) = \theta$ (параметры θ общие для всех моментов времени), то $\frac{\partial \theta(t+\varepsilon)}{\partial z(t)} = \frac{\partial \theta}{\partial z(t)} = 0$ и, следовательно,

$$\frac{\partial L}{\partial z(t)} = \frac{\partial L}{\partial z(t+\varepsilon)}\frac{\partial z(t+\varepsilon)}{\partial z(t)} \text{ или } a(t) = a(t+\varepsilon)\frac{\partial z(t+\varepsilon)}{\partial z(t)}. \tag{8}$$

Раскладывая $z(t+\varepsilon)$ в ряд Тейлора в точке t, имеем

$$\frac{\partial}{\partial z(t)} z(t+\varepsilon) = \frac{\partial}{\partial z(t)}(z(t) + \varepsilon f(z(t), t, \theta) + \mathcal{O}(\varepsilon^2)) = 1 + \varepsilon\frac{\partial}{\partial z(t)} f(z(t), t, \theta) + \mathcal{O}(\varepsilon^2). \tag{9}$$

Подставляя (9) в (8), находим

$$a(t) = a(t+\varepsilon)(1 + \varepsilon\frac{\partial}{\partial z(t)} f(z(t), t, \theta) + \mathcal{O}(\varepsilon^2)). \tag{10}$$

Тогда по определению производной

$$\begin{aligned}\frac{\partial a(t)}{\partial t} &= \lim_{\varepsilon \to 0+} \frac{a(t+\varepsilon) - a(t)}{\varepsilon} \overset{(10)}{=} \lim_{\varepsilon \to 0+} \frac{-\varepsilon a(t+\varepsilon) \frac{\partial}{\partial z(t)} f(z(t), t, \theta) + \mathcal{O}(\varepsilon^2)}{\varepsilon} = \\ &= -a(t) \frac{\partial}{\partial z(t)} f(z(t), t, \theta).\end{aligned} \tag{11}$$

Также известно граничное условие в конечный момент времени $a(T) = \frac{\partial L}{\partial z(T)}$. Таким образом, доказана справедливость (6).

Проведём некоторые аналогии с дискретным случаем для упрощения понимания приведённых выкладок:

Дискретный случай	Непрерывный случай
$z(t+1) - z(t) = f_t(z(t), \theta)$	$\frac{\mathrm{d}z(t)}{\mathrm{d}t} = f(z(t), t, \theta)$
$a(t) = a(t+1) \frac{\partial z(t+1)}{\partial z(t)}$	$a(t) = a(t+\varepsilon) \frac{\partial z(t+\varepsilon)}{\partial z(t)}$
$a(t+1) - a(t) = -a(t+1) \frac{\partial f_t(z(t), \theta)}{\partial z(t)}$	$\frac{\partial a(t)}{\partial t} = -a(t) \frac{\partial f(z(t), t, \theta)}{\partial z(t)}$

Таким образом, с помощью системы (6) и какого-либо численного метода можно находить значение $\frac{\partial L}{\partial z(t)}$ в момент времени $t \in [0, T]$.

Схожими рассуждениями можно показать, что динамика градиента $\frac{\partial L}{\partial \theta}$ задаётся следующим дифференциальным уравнением:

$$\begin{cases} \dfrac{\partial}{\partial t} \dfrac{\partial L}{\partial \theta(t)} = -a(t) \dfrac{\partial f(z(t), t, \theta)}{\partial \theta} \\ \dfrac{\partial L}{\partial \theta(T)} = 0, t \in [0, T] \end{cases} \tag{12}$$

Заметим, что частные производные $\frac{\partial}{\partial z(t)} f(z(t), t, \theta)$ и $\frac{\partial}{\partial \theta} f(z(t), t, \theta)$, определяющие динамики функций $a(t)$ и $\frac{\partial L}{\partial \theta(t)}$ соответственно, могут быть вычислены с помощью стандартного алгоритма обратного распространения ошибки.

Итак, опишем финальный алгоритм обучения модели нейродифференциального уравнения. Для обучения модели необходимо уметь вычислять значение градиента функции потерь в начальный момент времени $\frac{\partial L}{\partial \theta(0)}$. Динамика градиента $\frac{\partial L}{\partial \theta(t)}$ зависит от $a(t)$ и $z(t)$. Таким образом, для возможности применения численных методов для решения (12), необходимо уметь вычислять значения $a(t)$ и $z(t)$ в произвольных

точках отрезка $[0, T]$. Значение $z(t)$ может быть получено с помощью численного решения исходного уравнения (2) назад во времени, стартуя из точки $z(T)$. Зная значение $z(t)$, становится возможным найти значение $a(t)$ с помощью численного решения (6). Наконец, зная $a(t)$ и $z(t)$, может быть вычислено значение $\frac{\partial L}{\partial \theta(t)}$ в следующей точке, численно решая (12).

Таким образом, градиент функции потерь по настраиваемым параметрам может быть вычислен с помощью численного решения (2), (6) и (12) назад во времени. Все интегралы для нахождения z, a и $\frac{\partial L}{\partial \theta}$ могут быть вычислены единственным вызовом численного метода $ODESolve$. Алгоритм 1 показывает, как сконструировать начальное состояние и необходимые динамики для вычисления искомых градиентов за 1 вызов $ODESolve$.

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

4 Эксперименты на задаче классификации рукописных цифр

В данном разделе будут приведены результаты авторских и оригинальных экспериментов на датасете MNIST [8].

	Test Error	# Params	Memory	Time
1-Layer MLP†	1.60%	0.24 M	-	-
ResNet	0.41%	0.60 M	$\mathcal{O}(L)$	$\mathcal{O}(L)$
RK-Net	0.47%	0.22 M	$\mathcal{O}(\tilde{L})$	$\mathcal{O}(\tilde{L})$
ODE-Net	0.42%	0.22 M	$\boldsymbol{\mathcal{O}(1)}$	$\mathcal{O}(\tilde{L})$

Рис. 5: Результаты авторов статьи на MNIST. L - количество слоев, $\widetilde{L}$ - количество промежуточных вычислений метода $ODESolve$.

Авторами статьи исследовалась применимость модели нейродифференциального уравнения для решения задачи обучения с учителем. В качестве «традиционной» нейросетевой модели использовалась небольшая остаточная сеть, состоящая из 6 стандартных блоков со skip-connection (1). Исследованию подверглась модель ODE-Net, полученная из предыду-

щей архитектуры заменой блоков со skip-connection на блок нейродифференциального уравнения. Обучение данной модели проводилось с помощью метода, изложенного в разделе 3. Также были проведены эксперименты с аналогичной моделью, но обученной с помощью алгоритма обратного распространения ошибки через операции численного метода Рунге-Кутта (в таблице обозначена RK-Net). Результаты эксперимента приведены в таблице 5.

Модели ODE-Net и RK-Net продемонстрировали примерно такие же значения ошибки на тестовой выборке, что и архитектура ResNet. При этом стоит отметить, что, по сравнению с ResNet, у моделей ODE-Net и RK-Net почти втрое меньше обучаемых параметров. Данные результаты являются экспериментальным подтверждением применимости модели нейродифференциального уравнения на практике.

Также было проведено оригинальное исследование возможности модели ODE-Net явным образом контролировать trade-off между точностью и вычислительными затратами. Для этого самостоятельно была обучена модель ODE-Net (поэтому значения ошибки на тестовой выборке немного отличаются от Рис. 5). Далее считались предсказания на тестовой выборке данной модели с различными значениями допустимой абсолютной ошибки tol численного метода. Обучение модели проводилось со значением $tol = 10^{-3}$. В данном эксперименте вычислялись значения ошибки на тестовой выборке, среднее время предсказания для одного батча (размер батча был взят равным 1000), а также количество вычислений в промежуточных точках (NFE). Результаты эксперимента приведены в таблице 1.

	Ошибка на тесте	Среднее время	NFE
$tol = 10^{-4}$	0.39%	4.60c	320
$tol = 10^{-3}$	0.39%	3.76c	260
$tol = 10^{-2}$	0.40%	2.93c	200
$tol = 10^{-1}$	0.39%	2.13c	140
$tol = 1$	0.44%	2.13c	140

Таблица 1: Сравнение результатов модели с различными значениями абсолютной ошибки

На удивление, результаты эксперимента свидетельствуют о том, что значение допустимой абсолютной ошибки численного метода может быть увеличено в 100 раз (по сравнению с той, что использовалась при обучении), при этом ничуть не потеряв в точности предсказаний. Отметим, что таким образом удалось сократить время, требуемое для предсказания, более, чем в 1.5 раза. Таким образом, адаптивное время работы действительно является весомым преимуществом данной модели по сравнению с обычными нейросетями.

5 Непрерывные нормализационные потоки

Для понимания дальнейшего изложения необходимо иметь представление о модели нормализационных потоков [6] и задаче, которую они решают. Кратко опишем данную модель.

Пусть наблюдаемые объекты принадлежат некоторому пространству X и подчиняются некоторому распределению: $x \in X, x \sim p_{data}(x)$. Важной прикладной задачей является эффективное сэмплирование из данного распределения $p_{data}(x)$, а также вычисление значения плотности на сгенерированном объекте. Идея нормализационных потоков состоит в получении «сложного» модельного распределения $p_X(x)$, «близкого» к $p_{data}(x)$, из априорного распределения $p_Z(z)$ с помощью биективного отображения $f : X \to Z$ (Рис. 6).

Имея наблюдаемый объект $x \in X$, априорное распределение $p_Z(z)$ и биективное отображение $f : X \to Z$, плотность модели $p_X(x)$ может быть вычислена с помощью формулы замены переменных:

$$p_X(x) = p_Z\big(f(x)\big) \left| \det \left(\frac{\partial f(x)}{\partial x} \right) \right|, \tag{13}$$

$$\log\left(p_X(x)\right) = \log\Big(p_Z\big(f(x)\big)\Big) + \log\left(\left|\det\left(\frac{\partial f(x)}{\partial x}\right)\right|\right), \tag{14}$$

где $\frac{\partial f(x)}{\partial x}$ — матрица Якоби отображения f. Объекты из распределения $p_X(x)$ генерируются следующим образом: сначала генерируется объект из априорного распределения $z \in Z, z \sim p_Z(z)$, затем с помощью обратного преобразования вычисляется $f^{-1}(z) = x \sim p_X(x)$. Именно поэтому важно, чтобы преобразование f было обратимым.

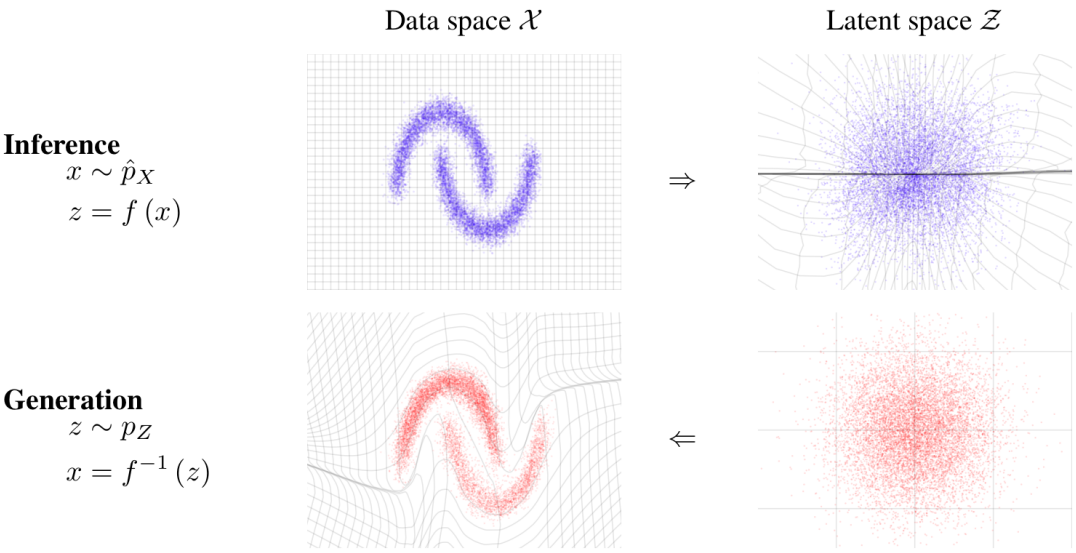

Рис. 6: Нормализационный поток

Основная проблема данной модели состоит в том, что вычисление якобиана $\frac{\partial f(x)}{\partial x}$ в общем случае крайне вычислительно затратно (имеет кубическую сложность от размерности пространства). Однако, выбирая отображение f специальным образом, можно добиться того, что якобиан будет считаться за линейное время. Примером такого отображения является так называемый planar flow [6], в котором преобразование слоя t выглядит следующим образом:

$$z(t+1) = z(t) + uh(w^T z(t) + b), \tag{15}$$

$$\log p(z(t+1)) = \log p(z(t)) - \log \left| 1 + u^T \frac{\partial h}{\partial z} \right| \tag{16}$$

где $h : Z_t \to Z_t$ — некоторое дифференцируемое нелинейное преобразование. Схожесть уравнения (15) с уравнением (4) в схеме Эйлера опять наталкивает на идею о переходе от дискретного к непрерывному во времени преобразованию.

Итак, рассмотрим модель непрерывных нормализационных потоков. Пусть $z(t)$ принадлежит распределению $p(z(t))$, которое зависит от времени. Пусть непрерывное во времени преобразование $z(t)$ задаётся следующим дифференциальным уравнением $\frac{\mathrm{d}z}{\mathrm{d}t} = f(z(t), t)$. Предполагая, что функция f Липшиц-непрерывна по z и непрерывна по t, имеет место следующая формула замены переменных:

$$\frac{\partial \log p(z(t))}{\partial t} = -tr\left(\frac{\partial f}{\partial z(t)}\right) \tag{17}$$

Удивительным является тот факт, что при переходе от дискретного к непрерывному во времени преобразованию формула замены переменных стала менее вычислительно затратной. Действительно, теперь вместо вычисления определителя матрицы Якоби в (14) необходимо вычислять след (17), что может быть сделано за линейное относительно размерности пространства время.

В качестве примера рассмотрим непрерывный аналог planar flow (15):

$$\frac{\mathrm{d}z}{\mathrm{d}t} = uh(w^T z(t) + b), \; \frac{\partial \log p(z(t))}{\partial t} = -u^T \frac{\partial h}{\partial z(t)} \tag{18}$$

Имея априорное распределение $p(z(0))$, мы можем сэмплировать из $p(z(T))$ и вычислять значение плотности с помощью численного решения данных дифференциальных уравнений.

Непрерывные нормализирующие потоки имеют огромное преимущество по сравнению со своим дискретным аналогом: в качестве преобразования f может использоваться любая непрерывная функция, поскольку, опять же, отсутствует необходимость вычисления определителя матрицы Якоби. Таким образом, данные модели являются более выразительными.

Динамика в нормализационных потоках может определяться суммой нескольких функций, что повышает экспрессивность модели. Поскольку след — линейная операция, имеем:

$$\frac{\mathrm{d}z}{\mathrm{d}t} = \sum_{n=1}^{M} f_n(z(t)), \; \frac{\partial \log p(z(t))}{\partial t} = -\sum_{n=1}^{M} tr\left(\frac{\partial f_n}{\partial z(t)}\right) \tag{19}$$

Таким образом, в непрерывных нормализационных потоках при росте модели в «ширину» вычислительная сложность растет линейно с ростом M, что является очень хорошим свойством. Для сравнения, в обычных нормализационных потоках вычислительная сложность подобных моделей имеет порядок $\mathcal{O}(M^3)$.

Результаты некоторых экспериментов авторов статьи с данной моделью приведены на Рис. 7. Визуально непрерывные нормализационные потоки с «шириной» M демонстрируют результаты не хуже, чем дискретные нормализационные потоки с глубиной $K = M$, при этом достигая меньших значений функции потерь.

Также, в отличии от дискретных потоков, в непрерывных нормализационных потоках для любой непрерывной f обратное преобразование может быть вычислено

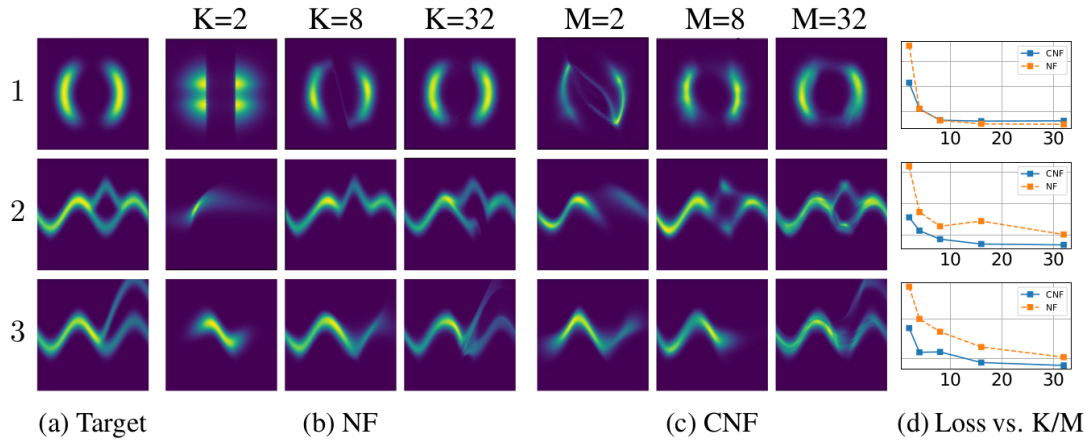

Рис. 7: Сравнение дискретных и непрерывных нормализующих потоков. K - глубина дискретного потока, M - «ширина» непрерывного потока (19).

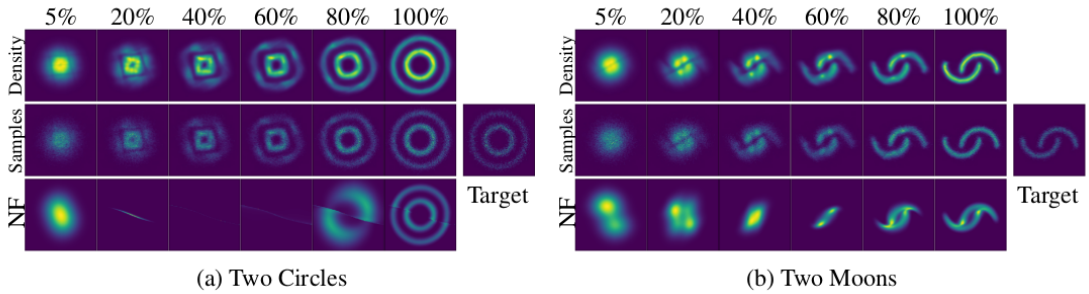

Рис. 8: Процесс преобразования в непрерывных нормализационных потоках.

примерно за такое же время, что и прямое. Это позволяет обучать данную модель напрямую с помощью метода максимального правдоподобия:

$$\mathbb{E}_{p_{data}(x)} \log p_X(x|\theta) \to \max_{\theta} \tag{20}$$

где $p_{data}(x)$ — реальное распределение данных, $p_X(x|\theta)$ — плотность модели непрерывного нормализационного потока. Результаты эксперимента с такой процедурой обучения приведены на Рис. 8. Можно отметить, что данная модель успешно справляется с восстановлением плотности в рассмотренных модельных задачах.

6 Заключение

Нейродифференциальные уравнения являются очень интересным классом моделей, который имеет свои несомненные достоинства. Первые эксперименты подтвер-

ждают применимость данных моделей на практике и демонстрируют сопоставимые с традиционными моделями результаты. Однако, поскольку данная область является достаточно новой и малоизученной, на текущий момент не было проведено полномасштабных экспериментов. Несмотря на то, что данное направление для исследований является очень перспективным, станет ли модель нейродифференциальных уравнений такой же популярной и широко используемой, как модель ResNet — большой вопрос.

Список литературы

[1] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential equations. *CoRR*, abs/1806.07366, 2018.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[3] C. Runge. Uber die numerische auflosung von differentialgleichungen. *Mathematische Annalen*, 1895.

[4] W. Kutta. Beitrag zur naherungsweisen integration totaler differentialgleichungen. *Zeitschrift fur Mathematik und Physik*, 1901.

[5] E. Hairer, S.P. Norsett, and G. Wanner. Solving ordinary differential equations - nonstiff problems. *Springer*, 1987.

[6] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. 2015.

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

[8] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.