МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
имени М. В. ЛОМОНОСОВА
Факультет вычислительной математики и кибернетики

А.С. Крылов

ВВЕДЕНИЕ В МАТЕМАТИЧЕСКИЕ МЕТОДЫ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

Учебное пособие

Москва, 2025

Введение

Данное пособие содержит часть материала спецкурса «Математические методы обработки изображений», читаемого на третьем курсе бакалавриата факультета вычислительной математики и кибернетики МГУ имени М.В. Ломоносова. Методы обработки изображений являются очень быстро развивающейся областью исследований. Однако, наряду с необходимостью разработки и адаптации все новых подходов и методов, включая в том числе и методы глубокого обучения, это требует от студентов освоения ряда базовых концепций математических методов обработки изображений. В пособии особое внимание уделено вопросам практического введения в использование как одномерного и двухмерного преобразований Фурье, так и интегральных преобразований с ядрами Фурье. Особое внимание уделено вопросу оптимальной одновременной локализации информации в пространственной и частотной областях: соотношению неопределенности; функциям Габора, на которых достигается минимум соотношения неопределенности; функциям Эрмита, являющимся собственными функциями преобразования Фурье. Рассмотрены также простейшие линейные и нелинейные методы повышения разрешения изображений, включая и регуляризирующие методы, и теорема Котельникова-Шеннона как теоретическое средство оценки возможности повышения разрешения изображений. Изложение базово ведется для случая функций непрерывного аргумента, переход к дискретным функциям проводится в основном как средство иллюстрации практического применения математических методов.

В расширенное издание книги добавлена 6 глава, кратно описываю-

щая новые возможности компьютерной обработки и анализа изображений, основанные на использовании гибридных методов, объединяющих нейросетевые и классические математические методы. Она частично отражает материалы спецкурса «Нейросетевые методы численного решения дифференциальных уравнений», поддерживаемого фондом «Интеллект» в рамках академической программы по искуственному интеллекту факультета ВМК МГУ имени М.В. Ломоносова. Глава, в отличие от остального содержания книги, содержит список литературы, позволяющий более серьезно углубиться в эту, развивающуюся в данное время, тематику.

Хотелось бы выразить благодарность за многолетнее сотрудничество в преподавательской и исследовательской деятельности к.ф.-м.н., со-руководителям студенческого учебного спецсеминара «Обработка изображений и компьютерное моделирование» факультета ВМК МГУ Насонову А.В., Сорокину Д.В., Хвостикову А.В., Павельевой Е.А., Пчелинцеву Я.А. и аспирантам Пенкину М.А., Жебрику Л.Л.

Глава 6

Классические математические методы обработки изображений и глубокое обучение

Часто при решении практических задач, с которыми уже не один год боролись ученые-прикладники, возникает следующая ситуация: берется набор данных (например, снимков с медицинского прибора с имеющимися результатами диагностики врачами), на этих данных обучается какая-нибудь нейронная сеть (трансформер, ...). И на данных, слабо отличающихся от тренировочных, эта сеть очень успешно решает задачу. Намного лучше, чем это получалось у "классических" алгоритмов без машинного обучения. Но в реальности устойчивости у нейросетевого решения нет. Для аналогичных данных сеть дает неправильные ответы ("шаг-влево, шаг-вправо" – и все становится очень плохо). Более того нейронные сети можно заставить ошибаться очень маленькими изменениями в данных. Этого нет у "классических" алгоритмов – они более устойчивы (во многом – из-за малопараметричности используемых моделей). В связи с этим очень перспективна идея объединить нейронные сети, позволяющие делать тонкую настройку на специфику конкретных данных, с алгоритмами, созданными ранее и основанными на математических моделях, например, удовлетворяющих законам сохранения. Отметим, что есть несомненная необходимость использования математиче-

ских моделей для повышения устойчивости нейросетевых методов. Однако и классические математические модели имеют ряд серьезных недостатков, преодолеть которые тоже можно на основе гибридных методов, основанных на нейросетевых методах и использующих, в то же время, математические модели.

Рассмотрим кратко основные из этих недостатков классических математических моделей, возникающих на практике:

- Человеческое восприятие не основано на метриках. Это можно проиллюстрировать примерами на Рис. 6.1-6.5.

Рис. 6.1: A

Рис. 6.2: B

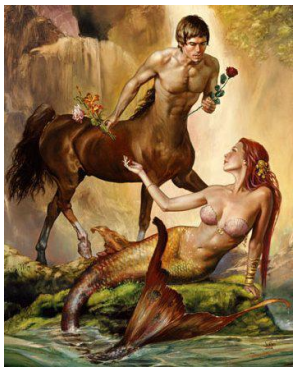

Рис. 6.3: C

Рис. 6.4: A

Рис. 6.5: B

Можно увидеть, что если оценивать “непохожесть” лошади (A, рис.

6.1) и мужчины (B, рис. 6.2) или рыбы (A, рис. 6.4) и девушки (B, рис. 6.5), то они велики. Если же мы захотим построить метрику ρ оценки похожести, то добавление дополнительного объекта - кентавра в первом случае и русалки – во втором, показывает невыполнение ключевого свойства математической метрики – неравенства треугольника.

Т.е. неравенство $\rho(A,C) + \rho(B,C) \geqslant \rho(A,B)$ не выполняется!

Данный пример может показаться надуманным, но на самом деле, он отражает реальную ситуацию. Например, при проведении диагностики пациента врачом по медицинскому снимку он при анализе изображения внутренне оперирует некоторыми характерными изображениями, ранее встречавшимися на практике и имевшими связь с дополнительной информацией о пациенте – результатами анализов, снимками полученными с помощью других медицинских модальностей, характером протекания заболевания и т.д. Таким образом, сравнение проводится отнюдь не только со снимками здоровых пациентов.

В связи с этим именно использование гибридным методом данных, размеченных человеком, позволяет получать решения, соответствующие человеческому восприятию.

- Классические математические модели всегда являются приближением реального процесса или явления, и они имеют серьезные ограничения для описания реальных данных. Очевидно, что математические модели не могут учитывать абсолютно всех факторов явления, то есть носят качественный характер. Необходимая точность вычисленного результата существенно повышается, если гибридный метод опирается и на реальные размеченные данные.

- Существует серьезная разница между решением прямой и обратной задач для математических моделей. Данная проблема кажется ме-

нее очевидной при обосновании необходимости гибридного подхода. При этом надо еще учитывать, что обратные задачи часто являются некорректно–поставленными. Однако для таких задач использование обучения на данных очень перспективно. Гибридные методы в каком-то смысле, являются, на существенно более высоком уровне развития технологий, отсылкой к грубым, но работавшим на практике подходам, когда человек при решении задачи классификации просматривал альбом изображений образцов и находил наиболее соответствующий, по его мнению, исследуемому.

Менее принципиальными, но важными с практической точки зрения являются следующие проблемы:

- Простые математические модели слишком грубые, в то время как сложные модели требуют слишком много вычислительных ресурсов.
- Даже сложные математические модели имеют относительно небольшое количество параметров оптимизации, и даже в этой ситуации часто очень сложно найти «оптимальное» решение (глобальный экстремум соответствующей вариационной задачи). Часто это тоже связано с тем, что модель не полностью соответствует экспериментальным данным. Но, тем не менее, исследователь упорно ищет это «оптимальное» решение, чего никогда не делает нейронная сеть.

Краткой иллюстрации некоторых основных, в настоящее время, направлений по созданию таких гибридных методов и посвящена данная глава.

Последние несколько лет характеризуются очень быстрым развитием нейросетевых методов, использующих, в том числе, математические модели, построенные для описания анализируемого процесса или объекта. Примером являются физически информированные нейронные сети (PINN, Physics-Informed Neural Networks), которые позволяют строить нейросетевое решение для нелинейных уравнений в частных производ-

ных. При этом, при обучении, в функции потерь учитывается требование удовлетворения дифференциального уравнения на сетке и его начальных и краевых условий.

Более перспективным по мнению многих авторов является новый подход, основанный на использовании так называемых нейронных операторов. Наиболее часто используемыми на практике являются нейронный оператор DeepONet [1] и нейронный оператор Фурье FNO [2]. Нейронные операторы основываются на использовании теоремы об аппроксимации перцептроном произвольного непрерывного нелинейного оператора с любой заданной точностью. Отметим, что обычно нейронные операторы применяются для моделирования операторов, основанных на использовании дифференциальных уравнений. Однако на практике они могут быть эффективны и для решения широкого круга задач, для которых в предыдущие годы были предложены нелинейные операторы (алгоритмы), решающие задачу, например, в компьютерном зрении, анализе изображений и т. д. К этому же подходу можно отнести и метод построения нейронных операторов, возникший в 2024 году и основанный на использовании теоремы Колмогорова-Арнольда о представлении, утверждающей, что каждая многомерная непрерывная функция может быть представлена в виде суперпозиции непрерывных функций одной переменной.

Мы рассмотрим эти направления для задач, описываемых моделями, связанными с применением дифференциальных уравнений (см., например, обзоры [3–5]). Необходимо также отметить, что концепция гибридных физически информированных нейронных сетей используется также и в трансформерах [6, 7] и мамбе [8].

6.1 Современные методы машинного обучения для решения дифференциальных уравнений

Рассмотрим следующее нелинейное ОДУ/УЧП:

$$\begin{cases} \mathfrak{F}_\lambda[u](x) = f(x), & x \in \Omega \\ \mathfrak{B}_\lambda[u](x) = b(x), & x \in \partial\Omega \end{cases} \tag{6.1}$$

где x — пространственно-временная координата, u — решение, λ — параметр математической модели, а $\mathfrak{F}$ и $\mathfrak{B}$ — общие нелинейные дифференциальный и граничный операторы соответственно. В литературе [9] обычно упоминаются два способа решения (6.1) с помощью современных методов машинного обучения.

Один из них — аппроксимация решения u с помощью параметризованной модели u_θ (где θ - параметры). Модель строит физически информированную функцию посредством автоматического дифференцирования, и ищет θ таким образом, чтобы минимизировать функцию потерь [10,11]. Представительной моделью являются физически информированные нейронные сети (PINN) [10]. Иногда эту модель называют нейронным дифференциальным уравнением.

Другой способ — изучение оператора решения, который отображает f, b и/или λ в u с помощью нейронной сети. Представительными моделями являются глубокие операторные сети (DeepONet) [1] и нейронные операторы Фурье (FNO) [2]. Их называют нейронными операторами. Главное различие между нейронными дифференциальными уравнениями и нейронными операторами заключается в том, что первые нацелены на решение одного конкретного ОДУ/УЧП, в котором обучение нейронной сети дает приближенное решение, отображающее точку в точку, тогда как вторые нацелены на решение семейства ОДУ/УЧП, в котором нейронная сеть отображает функции в функции.

6.1.1 Физически информированные нейронные сети

Метод PINN [10] решает проблему, включающую (6.1), путем моделирования искомого решения с помощью нейронной сети, обозначенной как u_θ, а затем моделирования f и b с помощью $\mathfrak{F}_\lambda[u_\theta]$ и $\mathfrak{B}_\lambda[u_\theta]$ посредством автоматического дифференцирования соответственно. Дифференциальное уравнение явно кодируется путем построения физически информированной функции потерь следующим образом:

$$\begin{aligned}\mathfrak{L}(\theta) \;=\; & \frac{w_u}{N_u}\sum_{i=1}^{N_u}\|\alpha_i(u_\theta(x_i^u)-u_i)\|^2 \\ & +\frac{w_f}{N_f}\sum_{j=1}^{N_f}\|\beta_j(\mathfrak{F}_\lambda[u_\theta](x_j^f)-f_j)\|^2 \\ & +\frac{w_b}{N_b}\sum_{l=1}^{N_b}\|\gamma_l(\mathfrak{B}_\lambda[u_\theta](x_l^b)-b_l)\|^2,\end{aligned} \tag{6.2}$$

где w_u, w_f, w_b — веса доверия для различных слагаемых в функции потерь, $\|\cdot\|$ — норма ℓ_2 для конечномерных векторов, $\{x_i^u, u_i\}_{i=1}^{N_u}$, $\{x_j^f, f_j\}_{j=1}^{N_f}$, $\{x_l^b, b_l\}_{l=1}^{N_b}$, — данные для u, f, b и α_i, β_j и γ_l — локальные веса (например, веса внимания на основе анализа точечных ошибок), которые уравновешивают вклад потерь точек обучения i, j и l соответственно.

Когда параметр λ известен, говорят, что решается прямая задача [10], и, соответственно, в функции потерь пропадает первое слагаемое. Иначе, если параметр λ неизвестен, то $N_u > 0$ и говорят, что решается обратная задача [12].

Одной из неотъемлемых проблем обучения нейронных сетей является то, что остатки в конкретных точках (то есть, точечные ошибки) могут быть упущены при вычислении кумулятивной функции потерь (т. е. суммирования или усреднения значений остатков). Для решения этой проблемы в нескольких исследованиях предлагалось масштабировать точечные члены функции потерь с использованием локальных множителей

[13, 14]. Локальные множители, такие как веса остаточного внимания [14] или самоадаптивные веса [13], показали замечательную эффективность в физически информированных нейронных сетях и других задачах контролируемого обучения. Эти веса уравновешивают вклад конкретных точек обучения в каждом члене потерь, вызывая остаточную однородность.

Несмотря на привлекательность подхода, использование аппарата физически информированных нейронных сетей не всегда эффективно. Так, например, в работе [15], где сравнивалось применение метода конечных элементов (FEM) и PINN для численного решения эллиптического уравнения Пуассона в 1D, 2D и 3D, параболического уравнения Аллена-Кана в 1D и гиперболического полулинейного уравнения Шредингера в 1D и 2D, были сделаны следующие выводы:

- С учетом времени решения и точности PINN не могут превзойти FEM. Решения FEM, как правило, быстрее, обеспечивая ту же или даже более высокую точность.
- PINN хороши для обобщения на более высокие измерения, где классические методы (такие как FEM) являются непомерно дорогими: - например, у PINN нет прироста вычислительной стоимости для уравнения Пуассона при переходе из 2D в 3D.
- Для определенных классов УЧП, для которых применимы классические методы, PINN не могут превзойти их.

Однако практическое использование PINN содержит много нюансов, и существует целый ряд приемов повышения производительности и точности. Описание некоторых из этих приемов приведено в [16]. Отражением большой активности в применении PINN в практической деятельности является, например, работа [17], где анализируются более чем 80 работ в области применения PINN для анализа медицинских изображений.

Аппарат PINN быстро развивается и число примеров его использования очень быстро растет. И большого прогресса удается добиться, когда имеющиеся экспериментальные данные для практической задачи не очень точно описываются используемым дифференциальным уравнением. В этом случае добавление в функцию потерь (6.2) еще и невязки получающегося описания экспериментальных данных позволяет рассматривать нейросетевой метод как эффективное расширение классического.

6.1.2 Нейронные операторы

Нейронные операторы (Neural Operators, NO) решают (6.1) путем аппроксимации оператора решения, обозначаемого как G_θ, с использованием нейронных сетей на основе данных [1, 2, 18]. В отличие от нейронных дифференциальных уравнений, в которых нейронная сеть отображает точку в точку, т. е. пространственно-временную координату в значение функции, вычисленной в этой координате, NO осуществляют отображения из функций в функции, например, отображение из исходной правой части f в искомое решение u, и могут использоваться как быстрые решатели для прямых задач и физические кодировщики [2, 19] для обратных задач. Обозначим входную функцию как v, а выходную функцию как u, и тогда функцию потерь можно записать следующим образом:

$$\mathfrak{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{N_u^i}\sum_{j=1}^{N_u^i}\left\|G_\theta(\mathbf{v}_i)(x_i^j) - u_i^j\right\|^2, \tag{6.3}$$

где $\{\mathbf{v}_i, \{x_i^j, u_i^j\}_{j=1}^{N_u^i}\}_{i=1}^{N}$ — данные для обучения NO. Здесь N обозначает количество парных данных для входной функции v и выходной функции u, $\mathbf{v}$ обозначает дискретное представление v, и $N_u^i, i = 1, \ldots, N$ — количество измерений для i-х данных для u, которое обозначается как $\{x_i^j, u_i^j\}$, $j = 1, \ldots, N_u^i$.

Нейронный оператор DeepONet

Нейронный оператор DeepONet был предложен в работе [1]. В дальнейшем был предложен ряд его модификаций (см., например, [18]). Опишем базовую архитектуру сети, описанную в оригинальном варианте нейронного оператора DeepONet [1].

Построение оператора основывается на теореме универсального приближения нелинейных операторов, предложенной в работе [20]:

Теорема. Предположим, что σ — непрерывная неполиномиальная функция, X — банахово пространство, $K_1 \subset X$, $K_2 \subset \mathbb{R}^d$ — два компакта в X и $\mathbb{R}^d$ соответственно, V — компакт в $C(K_1)$, G — нелинейный непрерывный оператор, отображающий V в $C(K_2)$. Тогда для любого $\varepsilon > 0$ существуют положительные целые числа n, p и m, константы $c_i^k, \xi_{ij}^k, \theta_i^k, \zeta_k \in \mathbb{R}$, $w_k \in \mathbb{R}^n$, $x_j \in K_1$, $i = 1, \ldots, n$, $k = 1, \ldots, p$, и $j = 1, \ldots, m$ такие, что

$$\left| G(u)(y) - \sum_{k=1}^{p} \underbrace{\sum_{i=1}^{n} c_i^k \sigma \left(\sum_{j=1}^{m} \xi_{ij}^k u(x_j) + \theta_i^k \right)}_{\text{branch}} \underbrace{\sigma(w_k \cdot y + \zeta_k)}_{\text{trunk}} \right| < \varepsilon, \qquad (6.4)$$

выполняется для всех $u \in V$ и $y \in K_2$. Здесь $C(K)$ — банахово пространство всех непрерывных функций, определенных на K с нормой $\|f\|_{C(K)} = \max_{x \in K} |f(x)|$.

Архитектура нейронной сети, построенной на основе уравнения (6.4), и смысл гиперпараметров n, p и m иллюстрируются на рис. 6.6.

Архитектура включает trunk (магистральную) сеть, которая принимает y в качестве входа и на выходе дает выходы $[t_1, t_2, \ldots, t_p]^T \in \mathbb{R}^p$. В дополнение к магистральной сети есть p branch (веточных) сетей, и каждая из них принимает $[u(x_1), u(x_2), \ldots, u(x_m)]^T$ в качестве входа и выдает на выходе скаляр $b_k \in \mathbb{R}$ для $k = 1, 2, \ldots, p$. Затем эти выходы

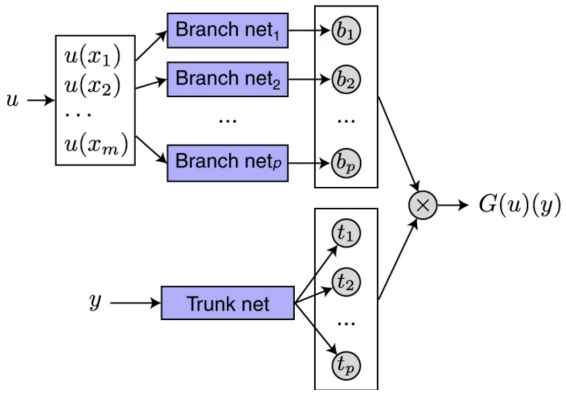

Рис. 6.6: Архитектура сети DeepONet [1]

объединяются по аналогии с уравнением (6.4):

$$G(u)(y) \approx \sum_{k=1}^{p} \underbrace{b_k(u(x_1), u(x_2), \ldots, u(x_m))}_{\text{branch}} \underbrace{t_k(y)}_{\text{trunk}}.$$

В отличие от теоремы (6.4) для повышения эффективностив сети добавляется смещение $b_0 \in \mathbb{R}$ на последнем этапе: $G(u)(y) \approx \sum_{k=1}^{p} b_k t_k + b_0$. Также часто вместо p веточных сетей используется одна веточная сеть, дающая на выходе вектор $[b_1, b_2, \ldots, b_p]^T \in \mathbb{R}^p$.

Единственным требуемым условием на расположение датчиков данных (sensors) является то, что их местоположения $\{x_1, x_2, \ldots, x_m\}$ одинаковы (необязательно на равномерной сетке) для всех входных функций u, в то время как никакие ограничения на выходные местоположения y не накладываются (рис. (6.7)). Однако даже это ограничение можно снять, например, интерполируя u на общий набор местоположений датчиков или проецируя u на набор базисных функций, а затем используя коэффициенты в качестве представления u.

Необходимо обратить внимание, что условия теоремы универсального приближения нелинейных операторов требуют компактности множеств K_1 и K_2. Данное условие является достаточно серьезным и, соответственно, предъявляет требования к постановке решаемой нейронным оператором задачи. В то же время, при выполнении данного условия можно

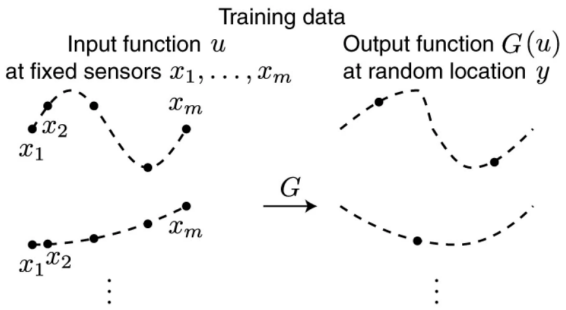

Рис. 6.7: Обучающие данные для сети DeepONet [1]

получать устойчивость нейросетевого решения не только для прямой, но и обратной задачи.

Нейронный оператор FNO

Нейронный опрератор Фурье FNO (Fourier Neural Operator) был предложен в работе [2]. Теоретически он может рассматриваться как частный случай оператора DeepONet [18], хотя его архитектура мотивирована несколько иными соображениями.

Для заданного УЧП, зависящего от параметра a

$$\begin{cases} (\mathfrak{L}_a u)(x) = f(x), & x \in \Omega, \\ u(x) = 0, & x \in \partial\Omega, \end{cases}$$

при выполнении определенных условий, решение может быть получено с помощью интегрального оператора с функцией Грина:

$$u^*(x) = \int\limits_{\Omega} G_a(x, y) f(y) dy.$$

Это решение может быть аппроксимированно нейронной сетью с параметром θ

$$\hat{u}(x) = \int\limits_{\Omega} K_\theta(x, y, a(x), a(y)) f(y) dy.$$

Предполагая, что K_θ является сверткой, можно эффективно вычислять этот интеграл в частотном пространстве преобразования Фурье,

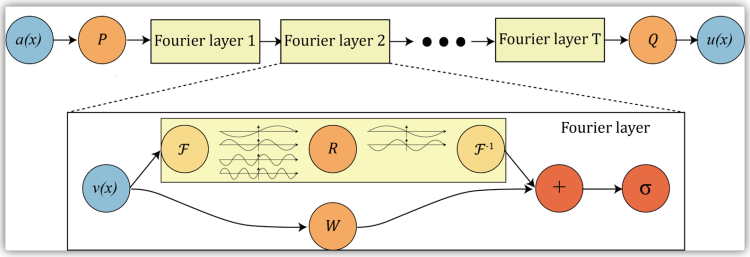

Рис. 6.8: Архитектура сети FNO [2]

существенно уменьшая сложность вычислений.

Последовательность вычислений для архитектуры FNO (рис. (6.8)):
Верхняя линия переходов

1. Вход $a(x)$ поднять в признаковое пространство большей размерности с помощью нейронной сети P.

2. Последовательно применить слои интегральных операторов и функций активации.

3. Спроецировать обратно в пространство целевой размерности с помощью нейронной сети Q и вывести u.

Нижняя линия переходов (архитектура слоя Фурье с входом v):
Сверху:

применить преобразование Фурье F;

линейное преобразование R к нижним модам Фурье и отфильтровать высшие моды;

применить обратное преобразование Фурье F^{-1}.

Снизу:

применить локальное линейное преобразование W.

Таким образом, последовательное применение Фурье–слоев осуществляется итеративным процессом

$$v_{t+1}(x) = \sigma(W v_t(x) + F^{-1}(R \cdot F(v_t))(x)),$$

где F и F^{-1} - прямое и обратное преобразование Фурье; W - матрица линейного преобразования в пространственной области; R - ядро в ча-

стотной области, σ - нелинейная функция активации.

6.1.3 Модели представления

Многослойный перцептрон (MLP)

Выход у многослойного перцептрона (Multi-Layer Perceptron, MLP) можно описать следующей вложенной формулой, где σ обозначает функцию активации, $W^{(l)}$ и $b^{(l)}$ — веса и смещения l-го слоя соответственно:

$$y(x) = \sigma(W^{(L)}\sigma(W^{(L-1)} \dots \sigma(W^{(1)}\mathbf{x} + b^{(1)}) \cdots + b^{(L-1)}) + b^{(L)}).$$

В этой формуле $\mathbf{x} = (x_1, x_2, \dots)$ представляет входной вектор, а L — количество слоев. Выход каждого слоя служит входом для следующего слоя, достигая конечного выхода y. Эта структура, в сочетании с достаточным количеством нейронов и правильным выбором функции активации, позволяет MLP аппроксимировать практически любую непрерывную функцию на компактных подмножествах $\mathbb{R}^n$, как утверждает теорема об универсальной аппроксимации [21]. Эта теорема подкрепляет способность нейронных сетей моделировать сложные нелинейные отношения. Именно комбинация физически информированного машинного обучения и многослойных перцептронов чаще всего называется нейронными сетями с учетом физики (PINN) (хотя сети, используемые в PINN, могут быть различными, например, сверточными).

Сети Колмогорова-Арнольда (KAN)

Сети Колмогорова-Арнольда (KAN) — это новый тип нейронных сетей, созданный на основе теоремы Колмогорова-Арнольда о представлении. Эта теорема утверждает, что любая многомерная непрерывная функция $f(x) = f(x_1, x_2, \dots)$ в ограниченной области может быть представлена как конечная композиция непрерывных функций одной переменной и бинарной операции сложения [22, 23].

Теорема Колмогорова-Арнольда [22, 23] имеет следующую формулировку.

Теорема. Если $f : [0,1]^n \to \mathbb{R}$ – это многомерная непрерывная функция, то её можно записать в виде конечной композиции непрерывных функций одной переменной и бинарной операции сложения. А именно,

$$f(x_1, ..., x_n) = \sum_{q=1}^{2n+1} \phi_q \sum_{p=1}^{n} \psi_{q,p}(x_p), \tag{6.5}$$

где $\phi_{q,p} : [0,1] \to \mathbb{R}$ и $\psi_q : \mathbb{R} \to \mathbb{R}$.

Назовем функции ψ внутренними, а ϕ внешними. Тогда в качестве замечаний к теореме можно заметить, что

- Внешних функций ϕ требуется $2n + 1$, где n — количество переменных исходной функции f;
- Внутренние функции ψ могут быть выбраны независимо от f, а внешние функции ϕ зависят от конкретной f;
- Теорема работает для функций, определённых на компактных множествах, таких как гиперкуб $[0,1]^n$;
- Все функции в представлении непрерывны, но не обязательно гладкие (могут, например, быть нигде не дифференцируемыми);
- Теорема доказывает существование функций для представления, но не предоставляет алгоритма их построения.

В теории этот результат, несмотря на указанные выше проблемы, выглядит многообещающим, поскольку позволяет свести многомерную задачу к серии одномерных преобразований, что должно снижать вычислительную сложность и облегчать моделирование сложных зависимостей.

Мотивируясь этой теоремой, авторы [22] предложили аппроксимировать $f(x)$ следующим образом:

$$f(x) \approx \sum_{i_{L-1}=1}^{n_{L-1}} \phi_{L-1,i_L,i_{L-1}} \Big(\sum_{i_{L-2}=1}^{n_{L-2}} \cdots \Big(\sum_{i_2=1}^{n_2} \phi_{2,i_3,i_2} \Big(\sum_{i_1=1}^{n_1} \phi_{1,i_2,i_1} \Big(\sum_{i_0=1}^{n_0} \phi_{0,i_1,i_0}(x_{i_0}) \Big) \Big) \Big) \cdots \Big).$$

Правая часть формулы представляет собой сеть KAN ($KAN(x)$), где L обозначает количество слоев, $\{n_j\}_{j=0}^{L}$ — количество узлов (то есть нейронов) в j-м слое, а $\phi_{i,j,k}$ — одномерные функции активации. Конкретная форма каждой $\phi(x)$ определяет вариации среди различных архитектур KAN.

В оригинальной реализации KAN [22] предлагалось определить $\phi(x)$ как взвешенную комбинацию базисной функции $b(x)$ и B-сплайнов. В частности:

$$\phi(x) = w_b b(x) + w_s \text{spline}(x),$$

где базисная функция $b(x)$ и сплайн $\text{spline}(x)$ определяются следующим образом:

$$\begin{aligned} b(x) &= \frac{x}{1+e^{-x}}, \\ \text{spline}(x) &= \sum_i c_i B_i(x). \end{aligned}$$

Здесь c_i, w_i и w_s являются обучаемыми параметрами. Сплайны $B_i(x)$ характеризуются полиномиальным порядком k и числом точек сетки g.

Прототип такой KAN с размерностью входных данных $n = 2$, вычислительный граф которого точно задан уравнением (6.5), и представляющий собой двухслойную нейронную сеть с функциями активации, размещенными на ребрах вместо узлов (на узлах выполняется простое суммирование), и с шириной $2n+1$ в среднем слое, приведен на рис. 6.9.

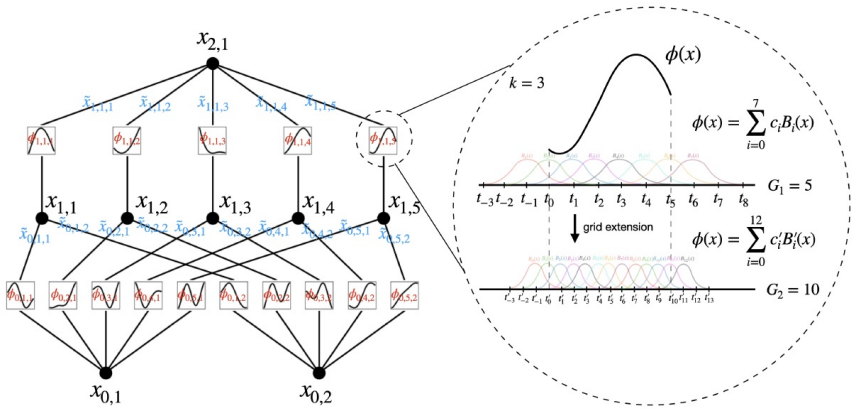

Рис. 6.9: Слева: схема сети. Справа: B-сплайн функция активации, позволяющая переключаться между крупнозернистыми и мелкозернистыми сетками (G - количество интервалов на сетке). [22]

С момента появления концепции KAN в апреле 2024 года исследователи по всему миру активно изучают разработку сетей и PINN на их основе, адаптированных для различных приложений. Существует довольно много вариаций KAN, которые можно протестировать, см. репозиторий Github для коллекции KAN (https://github.com/mintisan/awesome-kan). Среди них можно отметить следующие вариации KAN, основанные на использовании вместо B-сплайнов следующих функций:

- радиальных базисных функций [24];
- вейвлетов [25];
- полиномиальных базисных функций [26], таких как полиномы Чебышева [27] и описанные в главе 3 функции Эрмита.

Есть также работы по одновременному использованию нескольких видов функций в KAN [28].

Анализ эффективности использования MLP и KAN в задачах численного решения дифференциальных уравнений и операторных сетях был сделан в [29]. Схематически соответствие PINN, PIKAN (комбинации физически информированного машинного обучения и оригинальной KAN) и DeepONet проиллюстрировано на рис. 6.10. На рисунке функция активации для MLP в PINN и DeepONet выбрана как гиперболический тангенс только для демонстрации.

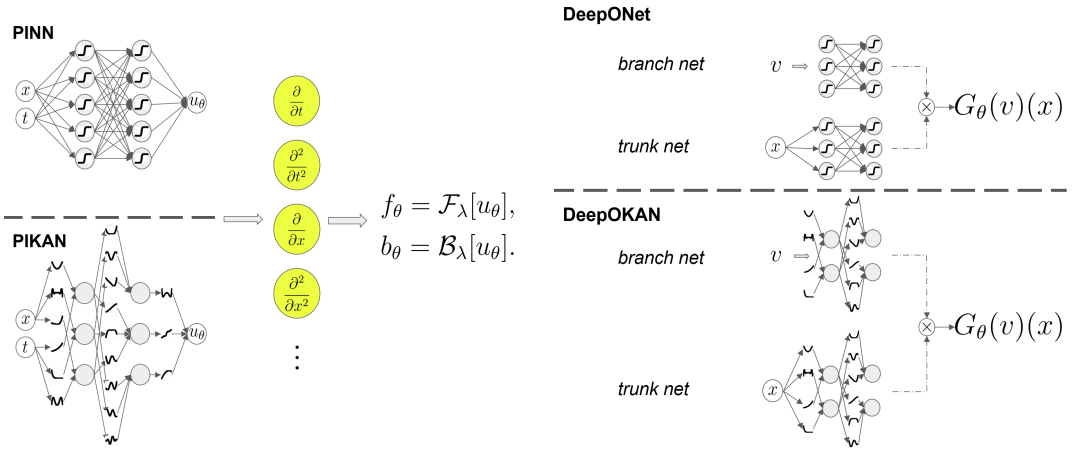

Рис. 6.10: Иллюстрация MLP и KAN для дифференциальных уравнений (слева) и операторных сетей (справа) [29]. В качестве модели представления для обучения операторов взята DeepONet.

Литература к главе 6

1. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators / L. Lu, P. Jin, G. Pang et al. // *Nature machine intelligence*. — 2021. — Vol. 3, no. 3. — Pp. 218–229.

2. Fourier neural operator for parametric partial differential equations / Z. Li, N. Kovachki, K. Azizzadenesheli et al. // *arXiv preprint arXiv:2010.08895*. — 2020.

3. Partial differential equations meet deep neural networks: A survey / S. Huang, W. Feng, C. Tang, J. Lv // *arXiv preprint arXiv:2211.05567*. — 2022.

4. *Hafiz A. M., Faiq I., Hassaballah M.* Solving partial differential equations using large-data models: a literature review // *Artificial Intelligence Review*. — 2024. — Vol. 57, no. 6. — P. 152.

5. *Hou Y., Zhang D.* A comprehensive survey on Kolmogorov-Arnold networks (KAN) // *arXiv preprint arXiv:2407.11075*. — 2024.

6. *Zhao Z., Ding X., Prakash B. A.* PINNsFormer: A transformer-based framework for physics-informed neural networks // *arXiv preprint arXiv:2307.11833*. — 2023.

7. *Cao S.* Choose a transformer: Fourier or Galerkin // *Advances in neural information processing systems*. — 2021. — Vol. 34. — Pp. 24924–24940.

8. Mamba Neural Operator: Who wins? Transformers vs. state-space models for PDEs / C.-W. Cheng, J. Huang, Y. Zhang et al. // *arXiv preprint arXiv:2410.02113*. — 2024.

9. Physics-informed machine learning / G. E. Karniadakis, I. G. Kevrekidis, L. Lu et al. // *Nature Reviews Physics.* — 2021. — Vol. 3, no. 6. — Pp. 422–440.

10. *Raissi M., Perdikaris P., Karniadakis G. E.* Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations // *Journal of Computational physics.* — 2019. — Vol. 378. — Pp. 686–707.

11. *Zou Z., Karniadakis G. E.* L-HYDRA: Multi-head physics-informed neural networks // *arXiv preprint arXiv:2301.02152.* — 2023.

12. A physics-informed neural network for quantifying the microstructural properties of polycrystalline nickel using ultrasound data: A promising approach for solving inverse problems / K. Shukla, A. D. Jagtap, J. L. Blackshire et al. // *IEEE Signal Processing Magazine.* — 2021. — Vol. 39, no. 1. — Pp. 68–77.

13. *McClenny L. D., Braga-Neto U. M.* Self-adaptive physics-informed neural networks // *Journal of Computational Physics.* — 2023. — Vol. 474. — P. 111722.

14. Residual-based attention in physics-informed neural networks / S. J. Anagnostopoulos, J. D. Toscano, N. Stergiopulos, G. E. Karniadakis // *Computer Methods in Applied Mechanics and Engineering.* — 2024. — Vol. 421. — P. 116805.

15. Can physics-informed neural networks beat the finite element method? / T. G. Grossmann, U. J. Komorowska, J. Latz, C.-B. Schönlieb // *IMA Journal of Applied Mathematics.* — 2024. — Vol. 89. — Pp. 143–174.

16. An expert's guide to training physics-informed neural networks / S. Wang, S. Sankaran, H. Wang, P. Perdikaris // *arXiv preprint arXiv:2308.08468.* — 2023.

17. PINNs for medical image analysis: A survey / C. Banerjee, K. Nguyen, O. Salvado et al. // *arXiv preprint arXiv:2408.01026.* — 2024.

18. A comprehensive and fair comparison of two neural operators (with practical extensions) based on fair data / L. Lu, X. Meng, S. Cai et al. // *Computer Methods in Applied Mechanics and Engineering.* — 2022. — Vol. 393. — P. 114778.

19. NeuralUQ: A comprehensive library for uncertainty quantification in neural differential equations and operators / Z. Zou, X. Meng, A. F. Psaros, G. E. Karniadakis // *SIAM Review.* — 2024. — Vol. 66, no. 1. — Pp. 161–190.

20. *Chen T., Chen H.* Approximations of continuous functionals by neural networks with application to dynamic systems // *IEEE Transactions on Neural networks.* — 1993. — Vol. 4, no. 6. — Pp. 910–918.

21. *Hornik K., Stinchcombe M., White H.* Multilayer feedforward networks are universal approximators // *Neural networks.* — 1989. — Vol. 2, no. 5. — Pp. 359–366.

22. KAN: Kolmogorov-Arnold networks / Z. Liu, Y. Wang, S. Vaidya et al. // *arXiv preprint arXiv:2404.19756.* — 2024.

23. *Khavinson S. I.* Best approximation by linear superpositions (approximate nomography). — American Mathematical Soc., 1997. — Vol. 159.

24. *Li Z.* Kolmogorov-Arnold networks are Radial Basis Function networks // *arXiv preprint arXiv:2405.06721.* — 2024.

25. *Bozorgasl Z., Chen H.* Wav-KAN: wavelet Kolmogorov-Arnold networks // *arXiv preprint arXiv:2405.12832.* — 2024.

26. *Seydi S. T.* Exploring the potential of polynomial basis functions in Kolmogorov-Arnold networks: A comparative study of different groups of polynomials // *arXiv preprint arXiv:2406.02583.* — 2024.

27. *SS S.* Chebyshev polynomial-based Kolmogorov-Arnold networks: An efficient architecture for nonlinear function approximation // *arXiv preprint arXiv:2405.07200.* — 2024.

28. FC-KAN: Function combinations in Kolmogorov-Arnold networks / H.-T. Ta, D.-Q. Thai, A. B. S. Rahman et al. // *arXiv preprint arXiv:2409.01763.* — 2024.

29. A comprehensive and FAIR comparison between MLP and KAN representations for differential equations and operator networks / K. Shukla, J. D. Toscano, Z. Wang et al. // *arXiv preprint arXiv:2406.02917.* — 2024.