

DOI: 10.51790/2712-9942-2020-1-4-3

**ПРИМЕНЕНИЕ МЕТОДОВ ГЛУБОКОГО ОБУЧЕНИЯ ДЛЯ РЕШЕНИЯ
ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ В ЧАСТНЫХ ПРОИЗВОДНЫХ****А. А. Епифанов***HERE Germany GmbH & Co. KG, г. Берлин, Германия**ORCID: <https://orcid.org/0000-0001-7403-4252>, wave911@yandex.ru*

Аннотация: глубокие нейронные сети стремительно развиваются в связи со значительным прогрессом в технологиях производительных вычислений. В данной работе рассматривается применение подходов, в основе которых лежит использование глубоких нейронных сетей, для решения дифференциальных уравнений в частных производных. Приводится пример численного решения уравнения Пуассона в двухмерной области методом Галеркина с глубокими нейронными сетями.

Ключевые слова: нейронные сети, глубокое обучение.

Для цитирования: Епифанов А. А. Применение методов глубокого обучения для решения дифференциальных уравнений в частных производных. *Успехи кибернетики*. 2020;1(4):22–28. DOI: 10.51790/2712-9942-2020-1-4-3.

DEEP LEARNING FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS**Alexander A. Epifanov***HERE Germany GmbH & Co. KG, Berlin, Germany**ORCID: <https://orcid.org/0000-0001-7403-4252>, wave911@yandex.ru*

Abstract: recently deep learning networks made huge progress due to the advances in high-performance computing technologies. This study covers a range of approaches to solving partial differential equations with deep learning. An example of solving the Poisson equation in a two-dimensional domain using the Galerkin method with deep neural networks is presented.

Keywords: deep learning, machine learning, partial differential equations, DGM, DRM.

Cite this article: Epifanov A. A. Deep Learning for Solving Partial Differential Equations. *Russian Journal of Cybernetics*. 2020;1(4):22–28. DOI: 10.51790/2712-9942-2020-1-4-3.

Введение

Дифференциальные уравнения в частных производных являются одним из самых широко применяемых инструментов для моделирования различных систем в природе. Наиболее сложные уравнения характеризуются большим количеством размерностей. Какими элегантными эти уравнения ни были бы, при их решении применительно к практическим задачам вычислительные затраты растут экспоненциально с увеличением числа размерностей. Основная проблема в традиционном подходе при приближении нелинейной многомерной функции с использованием кусочно-конечной аппроксимации, вейвлетов или других базисных функций — это большие размерности в задачах. Также существующие приближенные методы требуют решения важной вспомогательной задачи разбиения исходной области, например, как это делается в методе конечных элементов. Например, применение метода конечных размерностей становится невозможным в многомерном случае ввиду того, что многократно возрастает количество расчетных точек сетки исходной области и требования к небольшому размеру шага по времени. Размерность систем линейных алгебраических уравнений, порождённых сеткой, и временные рамки их решения определяют растущие требования в гонке роста производительности ЭВМ. В частности, источниками задач такого уровня сложности служат модели порового пространства в нефтегазовой промышленности, газовая и гидродинамика, физическая кинетика. Моделирование динамики крови в теле человека также порождает задачи похожего уровня сложности. Математически классы этих задач характеризуются существенно нелинейной динамикой на крайне сложно устроенном многообразии [1].

Со стремительным развитием технологий высокопроизводительных вычислений стали стремительно развиваться различные методы машинного обучения, и, в частности, методы глубокого обучения. Они показали хорошую производительность в различных задачах распознавания образов, обра-

ботки и распознавания голоса и текста и др. Эти значительные успехи в применении нейронных сетей вызваны их способностью хорошо аппроксимировать нелинейные функции.

В последние годы внимание исследователей захватила идея применения искусственных нейронных сетей для решения дифференциальных уравнений в частных производных, а также сравнения их эффективности с традиционным подходом с использованием численных методов.

Ключевыми работами, которые легли в основу теории методов нейронных сетей и глубокого обучения, стали работы А. Н. Колмогорова и В. И. Арнольда [2]. Предложенная ими теорема позволила заложить основу для развития искусственных нейронных сетей. Работы Р. Хект-Нильсена [3], К. Фунахаши [4] и Г. Цыбенко [5] позволили получить теоретическое обоснование для развития теории нейронных сетей и разработки методов глубокого обучения.

Глубокое обучение — это область машинного обучения, где некоторая построенная модель нейронной сети аппроксимирует целевую функцию на основе подготовленных знаний из предоставленных ей данных — будь то текст, изображение, звук и т.д. Термин «глубокое обучение» подразумевает, что используется многослойная искусственная нейронная сеть и в процессе ее обучения множество скрытых слоев. И в целом, чем больше слоев, тем лучше нейронная сеть может аппроксимировать целевую функцию. Согласно определению искусственная нейронная сеть представляет собой структуру параллельной распределенной обработки информации в виде ориентированного графа, где узлы графа называются обрабатывающими элементами, а связи называются соединениями. Пример структуры нейронной сети можно увидеть на рисунке 1.

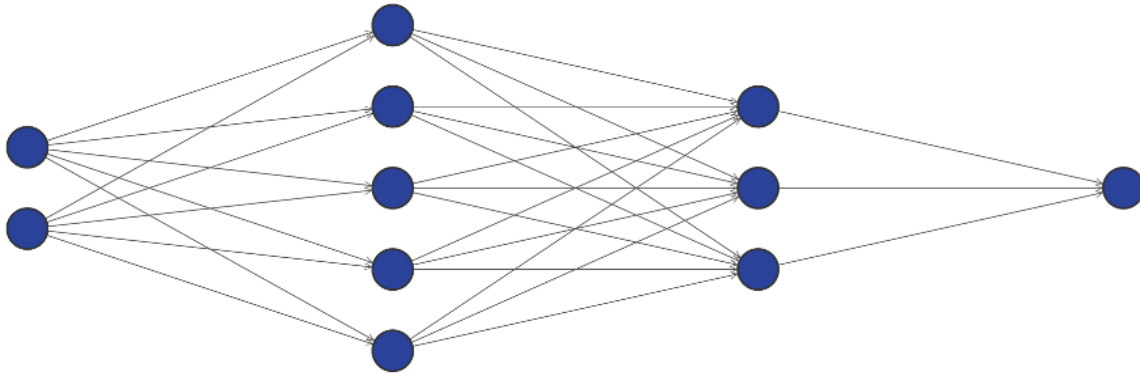


Рис. 1. Пример структуры нейронной сети

Таким образом, успех методов глубокого обучения во многих областях, связанных с вычислениями, позволяет рассмотреть возможность его применения при решении уравнений в частных производных.

Метод Галеркина с глубокими нейронными сетями (Deep Galerkin Method)

Метод для решения многомерных нелинейных дифференциальных уравнений был предложен Д. Сиригнано и К. Спилиопулусом [6], в основе его лежит идея метода Галеркина, но для аппроксимации решения дифференциального уравнения используется искусственная нейронная сеть вместо линейной комбинации базисных функций.

Рассмотрим нелинейное дифференциальное уравнение в общем виде:

$$\begin{aligned} \partial_t u(t, x) + \mathcal{L}u(t, x) &= 0, (t, x) \in [0, T] \times \Omega, \\ u(0, x) &= u_0(x), x \in \Omega \\ u(t, x) &= g(t, x), (t, x) \in [0, T] \times \partial\Omega, \end{aligned} \tag{1}$$

где $\partial\Omega$ — граница области Ω и заданы граничные $g(t, x)$ и начальные условия $u_0(x)$. Аппроксимация точного решения $u(t, x)$ дифференциального уравнения (1) может быть найдена с помощью минимизации функционала по норме L^2 :

$$J(f) = \|\partial_t f + \mathcal{L}f\|_{2, [0, T] \times \Omega}^2 + \|f - g\|_{2, [0, T] \times \partial\Omega}^2 + \|f(0, \cdot) - u_0\|_{2, \Omega}^2. \tag{2}$$

Функция ошибки $J(f)$ используется для оценки того, насколько хорошо приближенное решение f удовлетворяет дифференциальному оператору, граничным и начальным условиям. При этом никаких предварительных знаний о точном решении u нет и цель состоит в том, чтобы построить аппроксимацию f точного решения u так, чтобы значение функционала $J(f)$, часто в литературе можно встретить обозначение его как «функции ошибки аппроксимации», стремилось к нулю.

Алгоритм DGM аппроксимирует $u(t, x)$ с помощью глубокой нейронной сети $f(t, x, \theta)$, где $x \in \Omega \subset \mathbb{R}^d$, параметры (веса) нейронной сети $\theta \in \mathbb{R}^k$. Необходимо найти такой набор параметров θ , при котором функция ошибки $J(f)$ будет минимальной, при этом нейронная сеть $f(t, x, \theta)$ будет аппроксимировать точное решение уравнения (1) с заданными начальными и граничными условиями. Суть алгоритма DGM состоит в следующем:

1. Генерируются случайным образом точки (t_n, x_n) в области $[0, T] \times \Omega$ и точки (τ_n, z_n) в области $[0, T] \times \partial\Omega$ в соответствии с функциями плотности распределения вероятностей ν_1 и ν_2 . Также генерируются точки $\omega_n \in \Omega$ с функцией плотности распределения вероятности ν_3 .

2. Подсчитывается квадратичная ошибка $G(\theta, s_n)$ в сгенерированных точках $s_n = \{(t_n, x_n), (\tau_n, z_n), \omega_n\}$, где:

$$G(\theta, s_n) = \left(\frac{\partial f}{\partial t}(t_n, x_n, \theta_n) + \mathcal{L}f(t_n, x_n, \theta_n) \right)^2 + (f(\tau_n, z_n, \theta_n) - g(\tau_n, z_n))^2 + (f(0, \omega_n, \theta_n) - u_0(\omega_n))^2.$$

3. Делается шаг по методу стохастического градиентного спуска в точке s_n :

$$\theta_{n+1} = \theta_n - \alpha_n \nabla_{\theta} G(\theta_n, s_n),$$

где α_n — параметр, обозначаемый как «скорость обучения сети».

4. Повторяются шаги 1–3 до достижения критерия сходимости.

Следует отметить, что процесс минимизации функционала (2) может и остановиться в точке локального, а не глобального минимума для набора параметров θ , если $f(t, x, \theta)$ не выпуклая. В целом, известно, что глубокие нейронные сети не выпуклые функции, поэтому метод стохастического градиентного спуска может сойтись только к локальному минимуму, а не к глобальному.

Для расчета вторых производных в шаге 3 Д. Сиригнано и К. Спилиопулусом предложено использовать метод Монте-Карло, а в качестве архитектуры искусственной нейронной сети была предложена архитектура, очень похожая на архитектуру LSTM нейронных сетей:

$$\begin{aligned} S^1 &= \sigma(W^1(t, x) + b^1) \\ Z^l &= \sigma(U^{z,l}(t, x) + W^{z,l}S^l + b^{z,l}), & l = 1, \dots, L \\ G^l &= \sigma(U^{g,l}(t, x) + W^{g,l}S^l + b^{g,l}), & l = 1, \dots, L \\ R^l &= \sigma(U^{r,l}(t, x) + W^{r,l}S^l + b^{r,l}), & l = 1, \dots, L \\ H^l &= \sigma(U^{h,l}(t, x) + W^{h,l}(S^l \circ R^l) + b^{h,l}), & l = 1, \dots, L \\ S^{l+1} &= (1 - G^l) \circ H^l + Z^l \circ S^l, & l = 1, \dots, L \\ f(t, x, \theta) &= WS^{L+1} + b, \end{aligned}$$

где число скрытых слоев — $L+1$, \circ — скалярное произведение. В качестве функций активации используются нелинейные функции, такие как ReLU $\max(x, 0)$, сигмоидальная функция $\frac{e^x}{1+e^x}$ или гиперболический тангенс \tanh . Показано, что достаточно оптимальными значениями являются следующие: число нейронов в скрытом слое = 50, число слоев = 3 и функция активации — \tanh .

Пример решения двухмерного уравнения Пуассона методом Deep Galerkin:

$$\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} = 2, \quad \{(x, y) \in X : (x^2 + y^2 \leq R^2)\}.$$

С граничными условиями Дирихле:

$$P = g = -\frac{R^2 - (x^2 + y^2)}{2}, \quad x \in \partial\Omega.$$

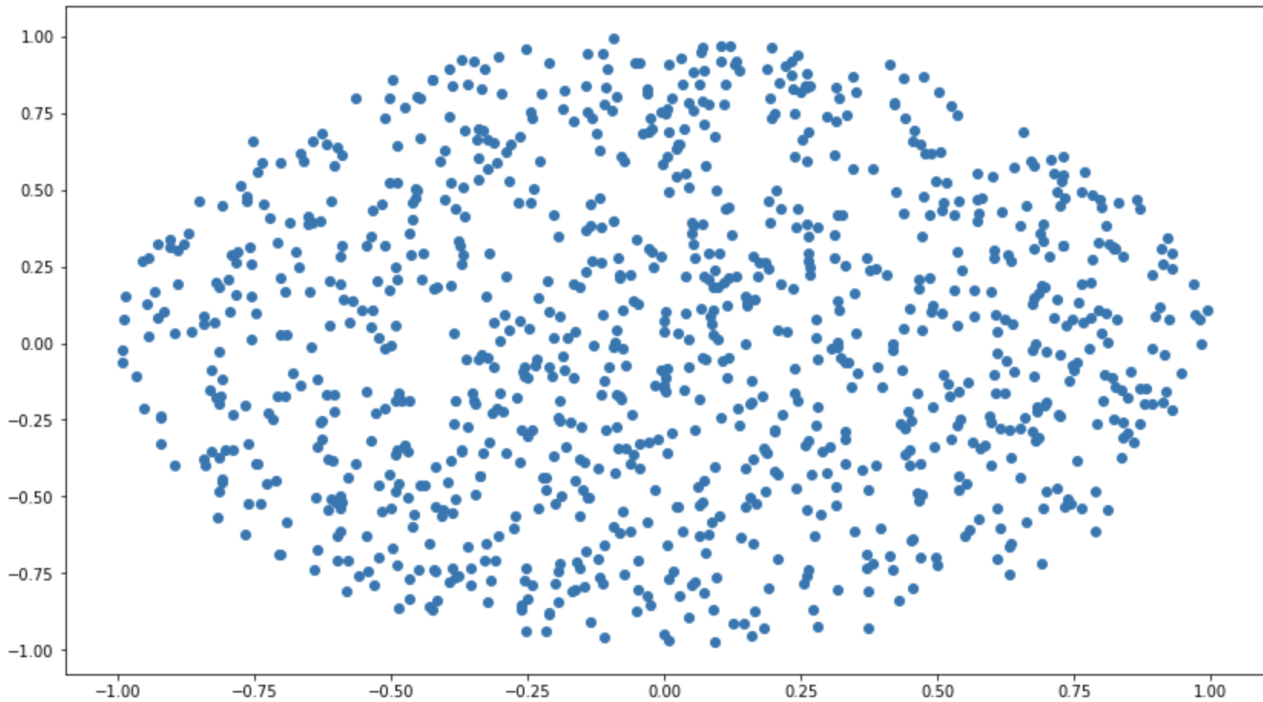


Рис. 2. Сгенерированные точки в области

Метод Ритца с глубокими нейронными сетями (Deep Ritz Method)

Метод для численного решения вариационных задач, в частности, тех, которые возникают при решении дифференциальных уравнений в частных производных, с помощью методов глубокого обучения был предложен Вейнань Е и Бинг Ю. [7], который может быть использован для решения многомерных задач.

Рассмотрим пример вариационной задачи на примере решения дифференциального уравнения Пуассона в ограниченной области Ω , которую будем решать методом Deep Ritz:

$$\begin{cases} -\Delta u(x) = f(x), & x \in \Omega \\ u(x) = 0, & x \in \partial\Omega. \end{cases} \quad (3)$$

Учитывая постановку задачи (3), можно сформулировать соответствующую вариационную задачу в виде:

$$J(v) = \frac{1}{2} \int_{\Omega} \nabla v(x) \cdot \nabla v(x) dx - \int_{\Omega} v(x)f(x) dx, \quad v \in H(\Omega). \quad (4)$$

Тогда решение задачи (3) методом Deep Ritz можно представить как минимизацию функционала (4):

$$u = \min J(v), \quad v \in H(\Omega). \quad (5)$$

Пусть \tilde{u} представляет собой приближение решения уравнения Пуассона (3) с помощью методов глубокого обучения. Тогда можно записать:

$$\tilde{u} = \min J(F), \quad F \in \tilde{\mathbb{F}}, \quad (6)$$

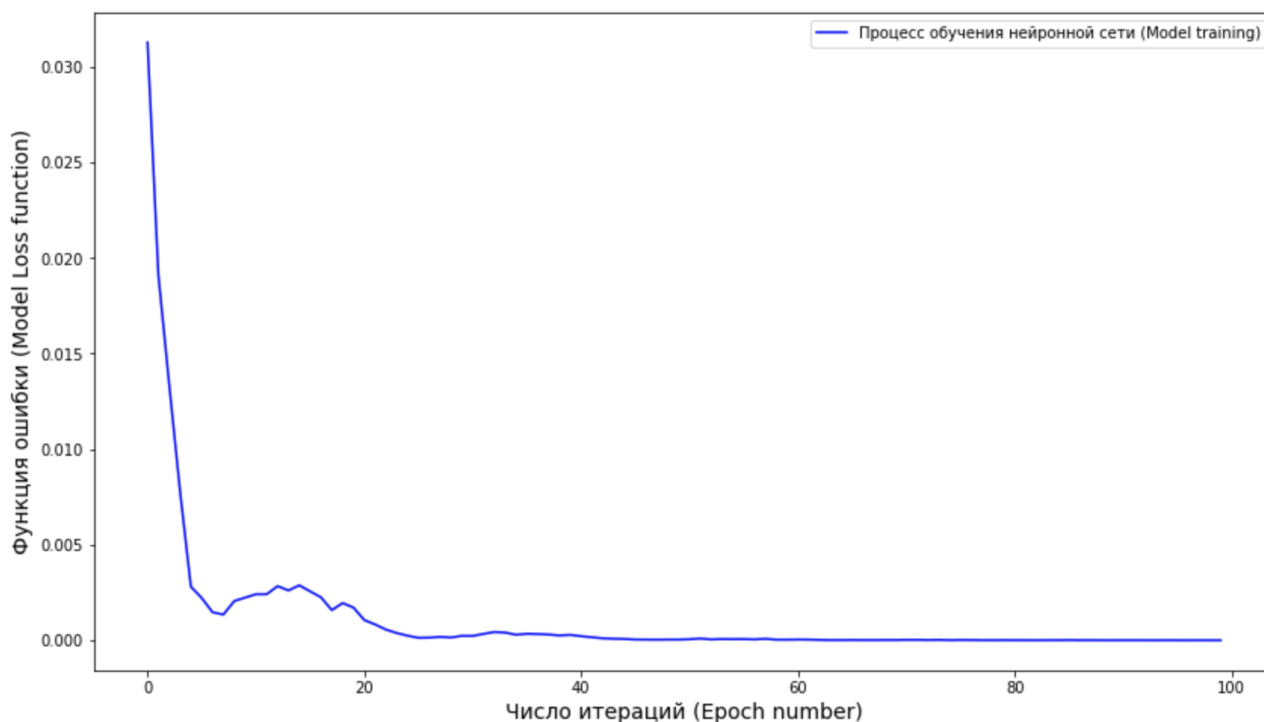


Рис. 3. Процесс обучения нейронной сети

где $\tilde{\mathbb{F}}$ является подпространством \mathbb{F} , которое удовлетворяет граничным условиям на границе $\partial\Omega$. Обозначим через $\theta \in \Theta$ параметры искусственной нейронной сети, которой будем аппроксимировать решение задачи (3). Тогда (4) можно записать в виде:

$$\min J(\theta) |_{\theta \in \Theta} = \frac{1}{2} \int_{\Omega} |\nabla F(x, \theta)|^2 dx - \int_{\Omega} F(x, \theta) f(x) dx. \quad (7)$$

Далее — отыскание оптимальных значений параметров θ нейронной сети, при которых значение функционала (7) минимально будет производиться по методу стохастического градиентного спуска [8]:

$$\theta_{n+1} = \theta_n - \alpha_n \nabla_{\theta} J(x_{j,n}, \theta_n). \quad (8)$$

На каждом шаге алгоритма (6) $n + 1$ генерируется набор точек $\{x_{j,n}\}$ в соответствии с функцией равномерного распределения плотности вероятности. В данном методе применяется архитектура нейронной сети типа ResNet.

Подход с использованием набора данных

Другой подход к решению дифференциальных уравнений в частных производных с помощью методов искусственных нейронных сетей подразумевает процесс «обучения», т.е. отыскания оптимальных значений параметров нейронной сети с помощью подготовленных наборов данных и их верификации.

Например, в работе [9] решались уравнения Навье—Стокса при обтекании двухмерного тела вязкой несжимаемой жидкостью. Для аппроксимации решения уравнений Навье—Стокса использовалась искусственная нейронная с архитектурой типа U-net, а набор данных для обучения сети создавался при помощи генерации двухмерных выпуклых тел случайным образом. Значения полей скорости и давления рассчитывались для каждого сгенерированного двухмерного тела при помощи пакета simLib. В результате подготовки обучающей выборки, на вход нейронной сети для ее «обучения» подавались 3 картинки размером 107x78 пикселей: геометрическая форма, поле скоростей и поле давления. В качестве активационных функций выбраны функции типа ReLu. На выходе работы нейронной сети также получается картинка размерностью 107x78 и для верификации точности аппроксимации решения уравнений Навье—Стокса выполнялось по пиксельное сравнение для верификационного и тестового набора данных.

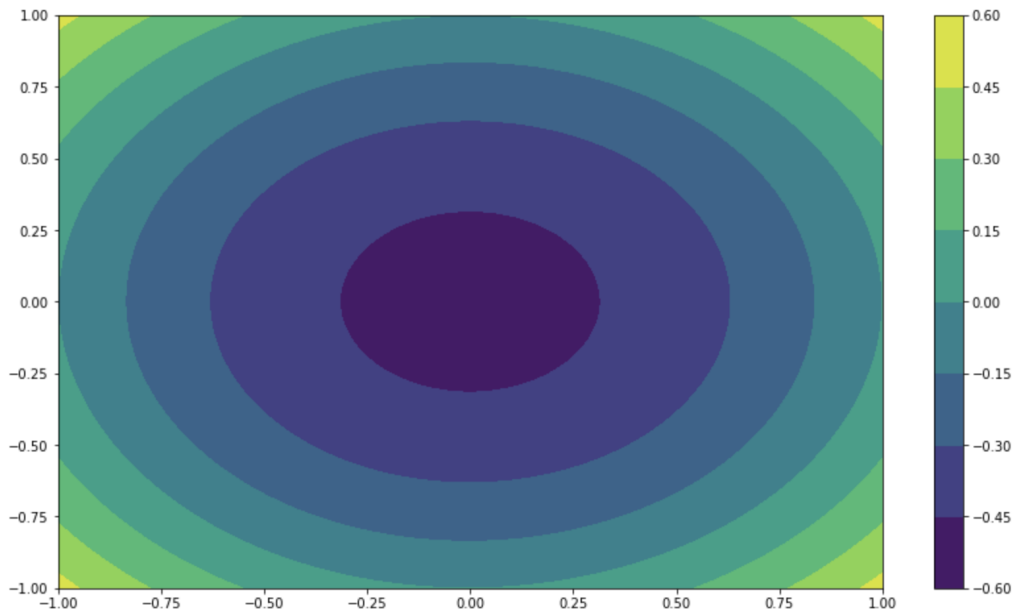


Рис. 4. Численное решение в области

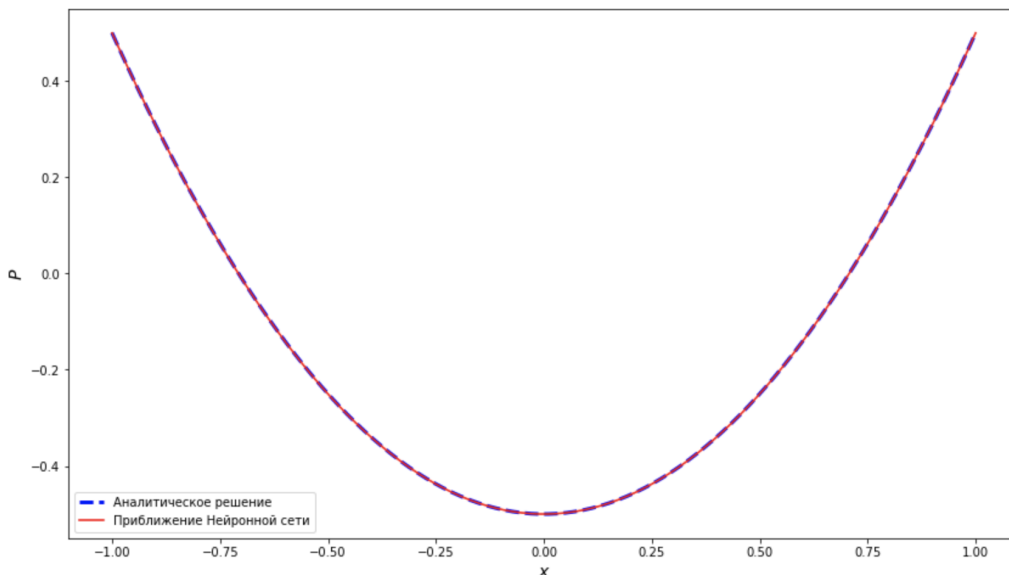


Рис. 5. Численное и аналитическое решения на сгенерированном наборе данных

В данной работе [10] также использовалась подготовка данных в виде набора сгенерированных картинок при предварительном решении дифференциальных уравнений известными численными методами и дальнейшем обучении многослойной нейронной сети по схеме, похожей на описанную в предыдущей работе.

А в работах [11] и [12] набор данных для «обучения» применяемых сверточных нейронных сетей получался путем предварительного решения уравнений Навье–Стокса широко используемыми численными методами, как, например, метод конечных разностей и метод конечных элементов.

Заключение

Можно отметить, что с каждым годом появляется все больше работ, в которых исследуются возможности применения искусственных нейронных сетей для решения дифференциальных уравнений в частных производных. Это становится возможным благодаря не только повышению производительности вычислительных систем, но и развитию методов нейронных сетей, и их активному применению при решении практических задач. Возникают вопросы: можно ли избежать необходимости генериро-

вать тестовых данных, которые получаются путем решения дифференциальных уравнений известными численными методами, например методом конечных разностей и методом конечных элементов. Так как эти данные являются не точными решениями, а только аппроксимацией, это вносит дополнительные ошибки уже на этапе подготовки данных, и затем добавляется ошибка аппроксимации нейронной сетью. Для простых геометрий расчетной области эти ошибки несущественны, но в более сложных многомерных постановках задач они требуют дополнительного изучения.

Также при обучении нейронной сети активно используются метод градиентного спуска и его модификации. Известны при этом сложность поиска глобального минимума минимизируемого функционала, а также зависимость от начальной инициализации параметров нейронной сети. При этом при попадании метода градиентного спуска в локальный минимум процесс обучения сети может остановиться и не достичь необходимого критерия сходимости. Соответственно, процесс надо начинать сначала.

Также не до конца изучен вопрос возможности обобщения решений дифференциальных уравнений с помощью нейронных сетей для различных начальных и граничных условий, которые не использовались в процессе их обучения. Тогда возникает необходимость переобучения нейронных сетей при изменении граничных и/или начальных условий исходной задачи. Этот процесс может быть очень ресурсоемким как с точки зрения вычислительных ресурсов, так и с точки зрения времени. Но тогда в чем же преимущество перед классическими численными методами?

Несомненно, что эти вопросы и описанные проблемы требуют дальнейшего изучения для достижения возможности использования методов глубоких нейронных сетей при решении практических задач моделирования сложных физических процессов и явлений.

ЛИТЕРАТУРА

1. Бетелин В. Б., Галкин В. А. Универсальные вычислительные алгоритмы и их обоснование для приближенного решения дифференциальных уравнений. *Доклады Академии наук*. 2019;488(4):351–357. DOI: <https://doi.org/10.31857/S0869-56524884351-357>.
2. Колмогоров А. Н. О представлении непрерывных функций нескольких переменных в виде суперпозиций непрерывных функций одного переменного и сложения. *Доклады Академии наук СССР*. 1957;114(5):953–956.
3. Hecht-Nielsen R. Kolmogorov's Mapping Neural Network Existence Theorem. *Proceedings of the International Conference on Neural Networks*. NewYork: IEEE Press. 1987. III:11–14.
4. Funahashi K. On the Approximate Realization of Continuous Mappings by Neural Networks. *Neural Networks*. 1989;2(3):183–192.
5. Cybenko G. Approximation by Superposition of a Sigmoidal Function. *Mathematics of Control, Signals, and Systems*. 1989;2:303–314.
6. Sirignano J., Spiliopoulos K. DGM: A Deep Learning Algorithm for Solving Partial Differential Equations. Available at: [arXiv:1708.07469](https://arxiv.org/abs/1708.07469).
7. Weinan E, Bing Yu. The Deep Ritz Method: A Deep Learning-Based Numerical Algorithm for Solving Variational Problems. Available at: <https://arxiv.org/pdf/1710.00211.pdf>.
8. Wang Z., Zhang Z. A Mesh-Free Method for Interface Problems Using the Deep Learning Approach. Available at: <https://arxiv.org/pdf/1901.00618.pdf>.
9. Chen J., Viquerat J., Hachem E. U-net Architectures for Fast Prediction of Incompressible Laminar Flows. Available at: <https://arxiv.org/pdf/1910.13532.pdf>.
10. Liu Y., Dinh N. Sato Y., Niceno B. Data-Driven Modeling for Boiling Heat Transfer: Using Deep Neural Networks and High-Fidelity Simulation Results. Available at: <https://arxiv.org/pdf/1808.01859.pdf>.
11. Tompson J., Schlachter K., Sprechmann P., Perlin K. Accelerating Eulerian Fluid Simulation With Convolutional Networks. Available at: <https://arxiv.org/pdf/1607.03597.pdf>.
12. Miyanawala T. P., Jaiman R. K. An Efficient Deep Learning Technique for the Navier-Stokes Equations: Application to Unsteady Wake Flow Dynamics. Available at: <https://arxiv.org/pdf/1710.09099.pdf>.