РАЗДЕЛ I. МАТЕМАТИКА

УДК 004.032.26

DOI: 10.18384/2310-7251-2021-2-6-17

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ РАБОТЫ НЕЙРОСЕТИ В РАСПОЗНАВАНИИ ОБЪЕКТОВ

Аксёнов К. А., Ключников С. А., Евстафьева С. Е., Калашников Е. В.

*Московский государственный областной университет
141014, Московская область, г. Мытищи, ул. Веры Волошиной, д. 24, Российская
Федерация*

Аннотация

Цель. Построение компьютерной модели работы нейронной сети для распознавания объектов.

Процедура и методы. На основании идей, положенных в основу теории распознавания и теории нейронных сетей, построена модель работы нейросети, предназначенной для распознавания исследуемых отображений с заданной точностью. Для успешной работы нейронной сети привлекались базы данных в открытом доступе с удалённых серверов. Это позволило использовать нейросети в стеснённых условиях (в отсутствии мощных вычислительных машин). Для организации работы и управления нейронной сетью разработана программа на языке Python.

Результаты. Построена нейронная сеть, распознающая исследуемые отображения с заданной точностью. Для управления построенной нейросетью и привлечения массивов баз данных с удалённых серверов разработана программа на языке Python. Продемонстрирован принцип работы нейросети на практике на примере распознавания изучаемых отображений.

Практическая значимость. Модель даёт реальный рецепт построения нейронной сети и использования её на практике в условиях отсутствия мощной вычислительной машины.

Ключевые слова: нейронные сети, математическая модель, библиотеки, программа управления, язык Python

COMPUTER SIMULATION OF NEURAL NETWORK OPERATION FOR OBJECT RECOGNITION

K. Aksenov, S. Klyuchnikov, S. Evstafyeva, E. Kalashnikov

Moscow Region State University

ul. Very Voloshinoy 24, 141014 Mytishchi, Russian Federation

Abstract

Aim. We construct a computer model of a neural network for object recognition.

Methodology. Based on the ideas underlying the theory of recognition and the theory of neural networks, we have constructed a model of neural network operation designed to recognize the studied mappings with a given accuracy. For the successful operation of a neural network, we have used open access databases from remote servers, which makes it possible to use neural networks in cramped conditions (in the absence of powerful computers). A program in Python has been developed to organize and manage the neural network.

Results. A neural network has been constructed that recognizes the studied mappings with a given accuracy. To manage the built neural network and attract database arrays from remote servers, a program in Python has been developed. The principle of neural network operation is demonstrated in practice, using the example of recognition of the studied mappings.

Research implications. The model provides a real recipe for constructing a neural network and using it in practice in the absence of a powerful computer.

Keywords: neural networks, mathematical model, libraries, program for managing the operation of a neural network, Python language

Введение

Нейронная сеть – это математическая модель, построенная по примеру деятельности нервной системы живого организма. Такая искусственная нейронная сеть отображает коллективное поведение взаимодействующих нейронов [3; 7–9] и имеет перспективы дальнейшего развития на основе квантовых явлений в наноструктурах [1].

Искусственная нейронная сеть (ИНС) [7; 8] представляет собой программную реализацию процессов, происходящих в коллективе взаимодействующих нейронов (искусственных). Любому нейрону можно поставить в соответствие функцию состояния γ , и эта функция зависит от особенностей взаимодействия между нейронами в коллективе. Взаимодействующие нейроны образуют нейросеть.

Нейросеть (ИНС) позволяет моделировать процесс обучения, запоминания и распознавания [8; 9]. ИНС является обучаемой системой. Она способна запоминать, сравнивать и воспроизводить на основе полученных данных новую информацию.

Математическая Модель Работы Нейросети

Моделируется деятельность системы распознавания. Распознавание подразумевает, что в исследуемом наборе объектов (в толпе людей и сопровождающих

их вещей), которые отображаются на экран монитора (X), нужно найти и выделить объекты, обладающие вполне конкретными характеристиками. Другими словами, каждому пикселю входного монитора X должен быть поставлен в соответствие пиксель с заданными характеристиками Y на выходном мониторе. Но такая зависимость не является однозначной, поскольку набору пикселей исходного прообраза X может соответствовать не одно, а множество значений из некоторой области значений характеристик (например, разные части цифр от 0 до 9 могут быть покрашены в разные цвета радуги). В таком случае каждому конкретному значению исходной переменной X соответствует некоторое вероятностное распределение зависимой переменной Y . Другими словами, зависимая переменная Y рассматривается как случайная величина. А влияние исходной величины X на зависимую переменную Y «в среднем» выражается через математическое ожидание M следующим образом:

$$M(Y | x) = f(x). \quad (1)$$

Эту связь называют **регрессией** Y на X [4; 5].

Чтобы учесть ситуацию, в которой реальные значения зависимой переменной не всегда совпадают с её условными математическими ожиданиями и могут быть различными при одном и том же значении, объясняющей переменной x , фактическая зависимость должна быть дополнена слагаемым ε , которое, по существу, является случайной величиной и указывает на статистическую суть этой зависимости (1). Тогда связь между зависимой и объясняющей переменной выражают соотношением:

$$Y = M(Y | x) + \varepsilon. \quad (2)$$

Такую связь называют регрессивной моделью или регрессивным уравнением [4].

Нейросети используют как набор входных нейронов (рис. 1), которые активируются с помощью пикселей входного сигнала $\hat{f}: x \rightarrow y$ по распределению x и y одновременно (здесь под x и y понимаются координаты картинки, выраженные в пикселях).

Для характеристики нейронной сети можно ввести функцию состояния:

$$\gamma(x) = \varphi(z_i, w_i, g_i(x)). \quad (3)$$

Этому соотношению соответствует графическое представление формулы (рис. 1):

Здесь φ – активационная функция нейросети (в нашем случае имеющая вид ступеньки); z_i – множество значений [z_i является переменной, несущей в себе все важные значения объясняющей переменной, составляющие функции нейросети, то есть $g_i(x) * W_i$]; $g_i(x)$ – набор нейронов (переменных, полученных на вход, которые могут быть представлены в разном виде, числом, объектом и т. д.); W_i – веса, предназначенные для учёта важности критериев (веса или «якоря», представляют собой важность одних значений над другими).

Рис. 1 / Fig. 1. Графическое представление функции $\gamma(x)$ нейросети, состоящей из одного нейрона / Graphical representation of the function $\gamma(x)$ of a neural network consisting of a single neuron.

Входные сигналы $X_1, X_2, \dots, X_n$ являются совокупностью $(g_i(x))$, с весами W_{in} (нейроны с определённой нагрузкой информации), суммой является z_i (набор всех входных сигналов вместе со всеми весами) и активационной функции ϕ , осуществляющей выход за порог и сопутствующим сообщением на экран / The input signals $X_1, X_2, \dots, X_n$ are a set $(g_i(x))$, with weights W_{in} (neurons with a certain load of information), the sum is z_i (the set of all input signals together with all weights) and the activation function ϕ , which goes beyond the threshold and the accompanying message to the screen.

Источник: составлено авторами.

Например, если рассматривать распознавание лица, то цвет волос или глаз будет иметь важное значение при определении личности человека.

Различают однослойные и многослойные нейронные сети (рис. 2).

Рис. 2 / Fig. 2. Графическое представление о слоях нейросетей / Graphical representation of neural network layers.

Источник: составлено авторами.

Первым слоем (рис. 2) является набор входных значений (сигналов) и он называется входным слоем, далее идут обработочные слои (осуществляющие набор весов, вычисление значений данных об объекте и, соответственно, обучение нейросети); и последним слоем является выходной, с выводом значений на экране и подписью (найденных) объектов.

Нейросеть, используемая в нашей работе, является многослойной. Функциональная пирамидальная сеть (FPN) [6] используется для построения богатой многомасштабной функциональной пирамиды из одного входного изображения с одним разрешением. Однако, программа не сможет рассматривать полученные изображение масштабом $A \times B$ менее чем 32×32 .

В нашей нейросети активационная функция ϕ принимает значения выхода за порог. В зависимости от того, насколько нам важен результат (при установленном нами пороге, например, на 95% или же на 43%), настолько можно говорить об успешности результатов.

Для получения полной информации об объекте с входного монитора или фотографии необходимо провести разбиение входного изображения на элементарные области (площади). Это предполагает разделение весов (для каждой площади свой набор весов).

Использование концепции разделяемых весов позволяет получать большее количество связей при меньшем количестве (наборе) весов. Имея изображения, каждый нейрон принимает изображение площадью 32×32 , причём каждый из таких фрагментов будет обработан одним набором весов.

Наборы весов, применяемые к таким фрагментам, называют ядрами.

Каждый фрагмент изображения поэлементно умножается на небольшую матрицу весов (ядро, kernel), результат суммируется. Эта сумма является пикселем выходного изображения, которое называется картой признаков.

Отметим, что взвешенная сумма входов ещё пропускается через функцию активации (как в любой другой нейросети). На самом деле это может происходить и в некотором S -ом слое, принципиальной разницы нет. Кроме того, отметим, что, в идеальном случае, через ядро проходят последовательно не разные фрагменты, а параллельно проходит всё изображение через идентичные ядра. Количество ядер (наборов весов) определяется разработчиком и зависит от того, какое количество признаков необходимо выделить. Ещё одна особенность свёрточного слоя в том, что он немного уменьшает изображение за счёт краевых эффектов.

Чередование слоёв позволяет составлять карты признаков. На практике это означает способность распознавания сложных иерархий признаков [10].

Обычно после прохождения нескольких слоёв карта признаков вырождается в вектор или даже скаляр, но таких карт признаков становятся сотни. В таком виде они подаются на один-два слоя полносвязной сети. Выходной слой такой сети может иметь различные функции активации. В простейшем случае функции активации может быть тангенциальной функцией.

Выход сети представляет собой карту объектов 50×50 . Слой свёрток [kernel 3×3] (матрица весов) проходит через изображение. В каждом месте он предска-

зывает 4 $[x1, y1, h1, w1]$ значения для каждого якорного ящика (z_i). Всего выходной слой имеет $50*50*9*4$ (здесь $50*50$ – «площадь» изображения в пикселях, 9 – набор весов, 4 – требуемые нами критерии/значения оценок вероятности выхода). Обычно это представлено в массиве numpy, как `pr.array (2500, 36)`.

Организация нейросети

Цель данной работы: моделирование распознавания работы ИНС, а также её применение на практике.

Для организации работы и управления нейронной сетью разработана программа на языке Python [2]. Для обеспечения работы нейросети необходимо выполнить следующие операции.

Шаг 1. Необходимо создать набор специальных библиотек, обеспечивающих распознавание объектов (официальная английская документация для ImageAI). Она доступна на GitHub¹, а также на своём официальном сайте². На ImageAI³ мы берём пакеты, которые нам необходимо скачать для функционирования нейросети. Устанавливаем набор библиотек, таких как:

- 1) **Tensorflow** : `pip3 install – upgrade tensorflow`;
- 2) **OpenCV** : `pip3 install opencv-python`;
- 3) **Keras** : `pip3 install opencv-python`;
- 4) **ImageAI** : `pip3 install imageai – upgrade`.

Шаг 2. Для того, чтобы установить эти команды (управления библиотеками) нужно зайти в терминал и прописать установку. После установки команд все библиотеки попадают в папку «проект» (см. рис. 3).

Шаг 3. Затем скачиваем и устанавливаем последнюю библиотеку ImageAI (рис. 4).

Шаг 4. После скачивания и установки библиотеки можно приступить к написанию кода.

¹ OlafenwaMoses / ImageAI [Электронный ресурс] // GitHub : [сайт]. URL: <https://github.com/OlafenwaMoses/ImageAI/tree/master/imageai/Detection> (дата обращения: 20.11.2020).

² Ptitov. Самопроизвольное разлогинивание [Электронный ресурс] // Хабр : [сайт]. URL: <https://habr.com/ru/post/7/> (дата обращения: 20.11.2020).

³ Official English Documentation for ImageAI! [Электронный ресурс]. URL: <https://imageai.readthedocs.io/en/latest/index.html> (дата обращения: 20.11.2020).

Рис. 3. / Fig. 3. Весь набор библиотек ImageAI / The entire set of ImageAI libraries
Источник: составлено авторами

Рис. 4 / Fig. 4. Финальная библиотека ImageAI / The final ImageAI Library.
Источник: составлено авторами.

Шаг 5. Далее следует скачать файл RetinaNet на официальном сайте GitHub. Он позволяет понимать, что конкретно происходит на изображении, и отображать пользователю то, что происходит на данном изображении. RetinaNet вмещает в себя описание множества объектов.

Перечисление всех объектов, которые способна распознать RetinaNet, представлено ниже:

person, bicycle, car, motorcycle, airplane, bus, train, truck, boat, traffic light, fire hydrant, stop_sign, parking meter, bench, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie, suitcase, frisbee, skis, snowboard, sports ball, kite, baseball bat, baseball glove, skateboard, surfboard, tennis racket, bottle, wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donot, cake, chair, couch,

potted plant, bed, dining table, toilet, tv, laptop, mouse, remote, keyboard, cell phone, microwave, oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy bear, hair dryer, toothbrush.

Шаг 6. Далее после скачивания модели (фотографии) размещаем её в папку «проект», куда устанавливали все библиотеки (шаги 1–3).

Код программы для управления нейронной сетью

Код программы пишем на языке Python [2]:

1. `from imagery. Detection import Object Detection` // Этот класс позволит обнаруживать некоторые объекты на картинках
2. `import` // это встроенная библиотека в Python и она позволяет работать с операционной системой.
3. `exec_path = os. getcwd()` // создаём новую переменную, называем её `exec_path` и помещаем в неё `os. getcwd()` эта функция позволяет указать путь к проекту. Этот путь позволяет обнаружить объекты на изображении.
4. `detector = Object Detection()` // создаём объект на основе Object Detection
5. `detector.setModelTypeAsRetinaNet()` // данная функция позволяет установить то, что мы используем RetinaNet для поиска объектов на изображении.
6. `detector.setModelPath(os.path.join (exec_path,"resnet50_coco_best_v2.0.1.h5"))` // установка пути к модели.
7. `detector.loadModel()` // загрузка модели.
8. `list=detector.detectObjectsFromImage(input_image=os.path.join(exec_path,"Название любой картинки"))` // создаём список, в который будут помещаться различные обнаружения на картинке.
9. `output_image_path=os.path.join(exec_path," Название любой картинки")` // указать место сохранения полученной фотографии.

Далее сохраняем и переходим к поиску любого изображения (например, в Google). После загрузки изображения оно будет добавлено в папку программы «проект».

Результат работы нейросети

Рис. 5. / Fig. 5. Скриншот рабочей программы / Screenshot of the working program.

Источник: составлено авторами.

Рис. 6. / Fig. 6. Фрагмент рис. 5. Лицо закрыто намеренно / Fragment of fig. 5. The face is covered intentionally.

Источник: составлено авторами.

Алгоритм выполнен, и на картинке видно, что программа обвела в рамку все найденные объекты и написала процентное соотношение того, кто или что находится на картинке, а также были выведенные проценты, которые показывают, насколько программа уверена в том, кто или что находится на картинке. На картинке были найдены люди и сумки (см. рис. 5 и 6).

Также в коде программы можно прописать дополнительные характеристики, такие как:

10. `Minimum_percentage_probability=30`, // это минимальный процент точности, если выставить процент 90% машина будет искать только те объекты в которых она уверена именно на 90%. Всё, что будет ниже будет записано как неверный ответ.

11. `Display_percentage_probability=False/True`, // Если `False` –отображение процентов указываться не будет, если `True` – отображение процентов указываться будет.

12. `Display_object_name=False/True`, // Если `False` – отображение имени указываться не будет, если `True` – отображение имени указываться будет.

Заключение

В данной работе были рассмотрены элементы теории построения нейросети и принцип её работы на практике. Для запуска программы нейросети была создана рабочая программа и визуальная картинка к ней. Вся программа была разработана на языке программирования Python. Python обладает широким набором возможностей для создания программ с искусственным интеллектом. Были использованы базы ImageAI, которые позволили пользоваться в

ограниченных условиях возможностями нейронных сетей. Построенная искусственная нейронная сеть, распознаёт исследуемые отображения с заданной точностью. Такая модель даёт реальный рецепт построения нейронной сети и использования её на практике в условиях отсутствия мощной вычислительной машины.

Статья поступила 08.06.2021 г.

ЛИТЕРАТУРА

1. Алтайский М. В., Капуткина Н. Е., Крылов В. А. Квантовые нейронные сети: современное состояние и перспективы развития // Физика элементарных частиц и атомного ядра. 2014. Т. 45. Вып. 5–6. С. 1824–1864.
2. Бутримов А. Л., Лаврентьев В. В. Python. Быстрое погружение в программирование: учебное пособие. М.: ИИУ МГОУ, 2018. 47 с.
3. Гафаров Ф. М., Галимянов А. Ф. Искусственные Нейронные сети и их приложение: учебное пособие. Казань: Казанский Государственный Университет, 2018. 121 с.
4. Любимцев О. В., Любимцева О. Л. Линейно-регрессивные модели в эконометрике. Нижний Новгород: ННГАСУ, 2016. 44 с.
5. Мазуров Вл. Д. Математические методы распознавания образов: учебное пособие. Екатеринбург: Издательство Уральского государственного университета, 2010. 101 с.
6. Ханеев Д. М., Филатова Н. Н. Пирамидальная сеть для классификации объектов, представленных нечёткими признаками // Известия ЮФУ. Технические науки. 2012. № 9 (134). С. 45–49.
7. Haykin S. Neural Networks and Learning Machines; 3rd edition. New Jersey: Pearson Education, 2009. 936 p.
8. Montavon G., Samek W., Müller Kl.-R. Methods for interpreting and understanding deep neural networks // Digital Signal Processing: A Review Journal. 2018. Vol. 73. P. 1–15. DOI: 10.1016/j.dsp.2017.10.011.
9. Thaler S., Furrer D. Neural Network Modeling // Advanced Materials & Processes. 2005. Vol. 163. Iss. 11. P. 42–46.
10. Liu T., Fang Sh., Zhao Y., Wang P., Zhang J. Implementation of Training Convolutional Neural Networks [Электронный ресурс] // arXiv : [сайт]. URL: <https://arxiv.org/ftp/arxiv/papers/1506/1506.01195.pdf> (дата обращения: 20.11.2020).

REFERENCES

1. Altaiskii M. V., Kaputkina N. E., Krylov V. A. [Quantum Neural Networks: Current State and Perspective of Development]. In: *Fizika elementarnykh chastits i atomnogo yadra* [Physics of Elementary Particles and Atomic Nuclei], 2014, vol. 45, no. 5–6, pp. 1824–1864.
2. Bugrimov A. L., Lavrent'ev V. V. *Python. Bystroe pogruzhenie v programmirovaniye* [Python. A Quick Dive into Programming]. Moscow, MRSU Ed. Office Publ., 2018. 47 p.
3. Gafarov F. M., Galimyanov A. F. *Iskusstvennyye Neironnye seti i ikh prilozhenie* [Artificial Neural Networks and Their Applications]. Kazan, Kazan State University Publ., 2018. 121 p.
4. Lyubimtsev O. V., Lyubimtseva O. L. *Lineino-regressivnye modeli v ekonometrike* [Linear Regression Models in Econometrics]. Nizhny Novgorod, Nizhny Novgorod State University of Architecture and Civil Engineering Publ., 2016. 44 p.

5. Mazurov V.I. D. *Matematicheskie metody raspoznavaniya obrazov* [Mathematical Methods for Pattern Recognition]. Ekaterinburg, Ural State University Publ., 2010. 101 p.
6. Khaneev D. M., Filatova N. N. [The pyramidal network for classification of objects, presented by fuzzy features]. In: *Izvestiya YuFU. Tekhnicheskie nauki* [Bulletin of the Southern Federal University. Technical science], 2012, no. 9 (134), pp. 45–49.
7. Haykin S. *Neural Networks and Learning Machines*; 3rd edition. New Jersey, Pearson Education Publ., 2009. 936 p.
8. Montavon G., Samek W., Müller K.I.-R. Methods for interpreting and understanding deep neural networks. In: *Digital Signal Processing: A Review Journal*, 2018, vol. 73, pp. 1–15. DOI: 10.1016/j.dsp.2017.10.011.
9. Thaler S., Furrer D. Neural Network Modeling. In: *Advanced Materials & Processes*, 2005, vol. 163, iss. 11, pp. 42–46.
10. Liu T., Fang Sh., Zhao Y., Wang P., Zhang J. Implementation of Training Convolutional Neural Networks. In: *arXiv*. Available at: <https://arxiv.org/ftp/arxiv/papers/1506/1506.01195.pdf> (accessed: 20.11.2020).

ИНФОРМАЦИЯ ОБ АВТОРАХ

Аксёнов Кирилл Александрович – студент физико-математического факультета Московского государственного областного университета;
e-mail: alexmayercom@mail.ru;

Ключников Семён Александрович – студент физико-математического факультета Московского государственного областного университета;
e-mail: semen.klyuchnikov@mail.ru;

Евстафьева Софья Евгеньевна – студент физико-математического факультета Московского государственного областного университета;
e-mail: e.sony@inbox.ru;

Калашников Евгений Владимирович – доктор физико-математических наук, профессор, кафедры вычислительной математики и методики преподавания информатики Московский государственный областной университет;
e-mail: ekevkalashnikov1@gmail.com

INFORMATION ABOUT THE AUTHORS

Kirill A. Aksjonov – Student, Physics and Mathematics Faculty, Moscow Region State University;
e-mail: alexmayercom@mail.ru;

Semen A. Klyuchnikov – Student, Physics and Mathematics Faculty, Moscow Region State University;
e-mail: semen.klyuchnikov@mail.ru;

Sofya E. Evstafyeva – Student, Physics and Mathematics Faculty, Moscow Region State University;
e-mail: e.sony@inbox.ru;

Evgenii V. Kalashnikov – Dr. Sci. (Phys.-Math.), Prof., Department of Computational Mathematics and Teaching Computer Science, Moscow Region State University;
e-mail: ekevkashnikov1@gmail.com

ПРАВИЛЬНАЯ ССЫЛКА НА СТАТЬЮ

Аксёнов К. А., Ключников С. А., Евстафьева С. Е., Калашников Е. В. Компьютерное моделирование работы нейросети в распознавании объектов // Вестник Московского государственного областного университета. Серия: Физика-Математика. 2021. № 2. С. 6–17.
DOI: 10.18384/2310-7251-2021-2-6-17

FOR CITATION

Aksenov K. A., Klyuchnikov S. A., Evstafyeva S. E., Kalashnikov E. V. Computer simulation of neural network operation for object recognition. In: *Bulletin of the Moscow Region State University. Series: Physics-Mathematics*, 2021, no. 2, pp. 6–17.
DOI: 10.18384/2310-7251-2021-2-6-17