Искусственные нейронные сети

В настоящее время один из самых популярных методов создания систем искусственного интеллекта и машинного обучения – это глубокие нейронные сети.

Искусственный интеллект (Artificial Intelligence, AI) – раздел информатики, изучающий возможность обеспечения разумных рассуждений и действий с помощью вычислительных систем и устройств.

Выделяют два типа искусственного интеллекта (ИИ):

1. **Сильный ИИ** – осознает себя, способен самостоятельно мыслить и может заменить человека почти во всех сферах.
2. **Слабый ИИ** – реализует какие-либо аспекты, похожие на мышление, например, компьютерное зрение, машинный перевод, голосовые помощники и т.д.

В настоящее время даже такой современный подход, как искусственные нейронные сети, в целом пока не может быть отнесен к сильному ИИ.

Искусственные нейронные сети

Искусственные нейронные сети (ИНС):

- **С математической точки зрения** обучение ИНС – это многопараметрическая задача нелинейной оптимизации.
- **С точки зрения машинного обучения** ИНС представляет собой частный случай методов распознавания образов, дискриминантного анализа, методов кластеризации и т.д.
- **На технологическом уровне** ИНС используется как черный ящик, который позволяет распознавать образы, автоматизировать перевод текстов и т.д.

В целом ИНС – это крайне удобный и развитый инструмент машинного обучения, который позволяет решать множество задач. В настоящее время алгоритм ИНС реализован с помощью разных библиотек, которые достаточно хорошо автоматизированы, поэтому ИНС могут применяться для решения практически любой задачи неспециалистами в области ИИ.

Искусственные нейронные сети

Искусственные нейронные сети (ИНС) – это математические модели, а также их программные или аппаратные реализации, построенные по принципу организации и функционирования биологических нейронных сетей – сетей нервных клеток живого организма.

Модель искусственного нейрона была разработана по аналогии с устройством нейрона головного мозга человека. Однако эта модель была создана давно, когда об устройстве мозга знали гораздо меньше, чем сейчас, поэтому данная аналогия в действительности очень поверхностная.

В головном мозге человека содержится примерно 80-90 миллиардов нейронов, а ИНС содержат всего лишь десятки тысяч искусственных нейронов, поэтому до сильного ИИ все еще очень далеко. ИНС в настоящее время – это пока лишь попытка приблизиться к мозгу насекомого.

Искусственные нейронные сети

В традиционных методах машинного обучения важным этапом является выделение из всех имеющихся данных только тех, которые важны для решения задачи. Эти данные называются признаками, и есть целые направления, которые описывают, как правильно выделять признаки. Если удается выделить качественные признаки, то задача решается хорошо, в противном же случае алгоритм машинного обучения работает плохо.

Особенность нейронных сетей, которая обеспечивает их популярность, – это отсутствие необходимости выделять признаки. На вход нейронной сети можно подавать все имеющиеся данные и она сама определит, какие данные важны, а какие нет. Однако для этого требуются большие объемы данных и мощные вычислительные ресурсы.

Сейчас нейронные сети успешно применяются для создания различных сложных систем искусственного интеллекта, многими из которых мы уже пользуемся.

Искусственные нейронные сети

Одно из самых известных и впечатляющих применений нейронных сетей – это **управлением автомобилями без водителей**. Сейчас многие компании в мире занимаются созданием систем автоматического управления для своих автомобилей. Первые в мире такси без водителей запустила **компания nuTonomy в Сингапуре**. Поездки были доступны только в ограниченной области внутри города.

Искусственные нейронные сети

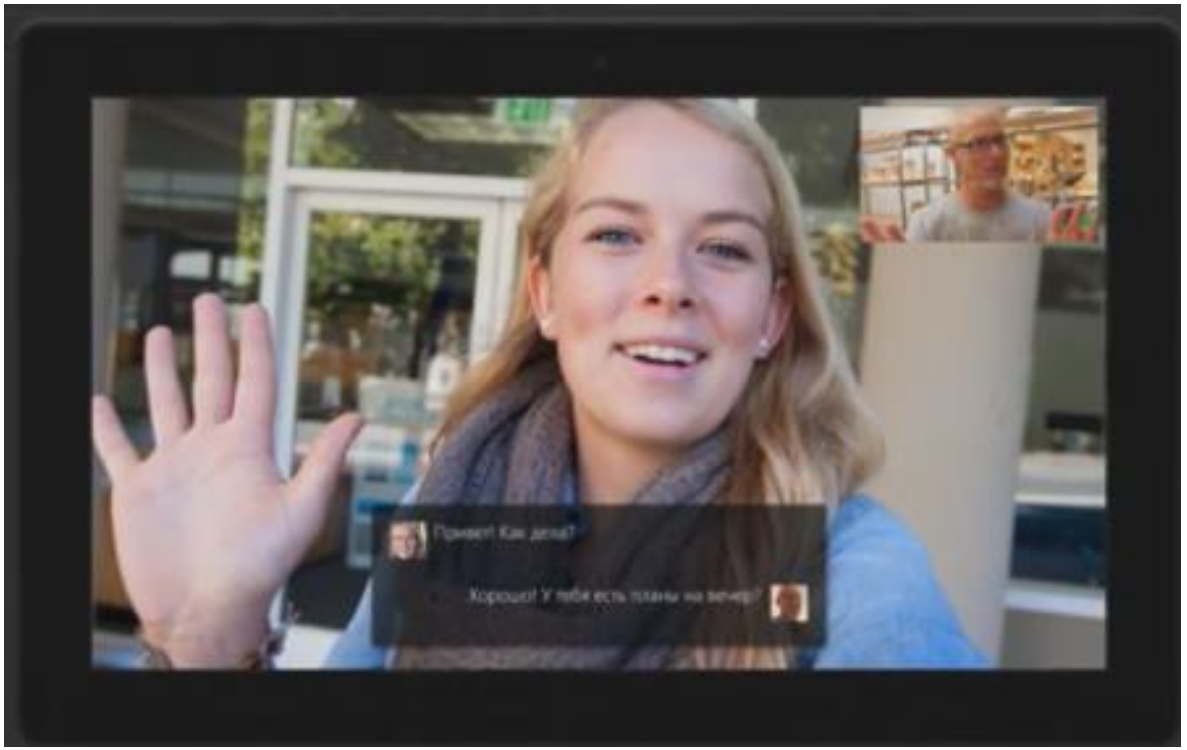
Компания Uber запустила такси без водителя **в США** в гораздо большем масштабе, чем nuTonomy. Однако широкое распространение такси без водителя было существенно замедлено, когда в марте 2018 года автомобиль Uber сбил женщину, которая переходила дорогу, и она умерла в больнице. С тех пор при разработке и тестировании самоуправляемых автомобилей применяются повышенные меры безопасности.

В нашей стране также занимаются разработкой самоуправляемых автомобилей, это делают, например, компании **Яндекс** и **Камаз.** Правительство России выбрало два региона, в которых будет проводится тестирование самоуправляемых автомобилей на дорогах общего пользования – это **Москва** и **Татарстан**. Эксперимент проходит **с 2018 по 2022 год**. Сейчас проехать на такси без водителя можно в **Иннополисе** и **Сколково**.

Искусственные нейронные сети

Другое популярное применение глубоких нейронных сетей – это **автоматический перевод**. Современные системы перевода от **Google** и **Яндекс** используют нейронные сети для перевода текстов.

В Skype есть возможность синхронного перевода речи собеседников **с 10 языков**, включая русский. Ваш собеседник говорит на английском, а вы слышите русскую речь и видите текст перевода. Это реализовано с помощью **сервиса Skype Translator** на основе глубоких нейронных сетей.



Искусственные нейронные сети

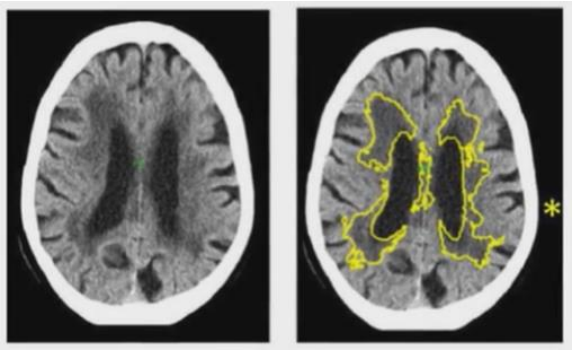
Нейронные сети все чаще применяются **для автоматизации рутинного интеллектуального труда**. Глава Сбербанка Герман Греф объявил, что еще в 2016 году для подготовки исков стали применяться нейронные сети. Сети заменили 450 юристов, которые ранее готовили такие иски вручную. В дальнейшем, по утверждению Грефа, Сбербанку не будут нужны юристы, которые не могут работать с нейронными сетями.

Нейронные сети позволяют похожим образом **автоматизировать и бухгалтерию**. Например, компания Кнопка создала более 40 роботов, которые автоматически выполняют бухгалтерские операции. Многие из этих роботов используют нейронные сети.

Также роботы на основе нейронных сетей хорошо справляются с **поиском сотрудников и анализом резюме**. Например, банк «Открытие» искал более 100 сотрудников в кол-центр. В привлечении сотрудников использовалась в том числе **платформа Skillaz**, робот которой за неделю нашел больше кандидатов, чем пять сотрудников отдела кадров за месяц.

Искусственные нейронные сети

Разрабатываются системы по применению глубоких нейронных сетей **для помощи врачам при анализе медицинских изображений**. Такие изображения, как правило, зашумлены, чтобы их корректно интерпретировать нужна высокая квалификация врачей и много времени. Нейронные сети позволяют выделить на медицинских изображениях интересующие участки, что позволяет врачам значительно быстрее поставить правильный диагноз.



Глубокие нейронные сети также применяются и **для творческих задач**. Одно из популярных направлений: **рисование картин нейронной сетью**. Такие картины уже стали приобретать популярность. Группа французских студентов Obvious на основе программы с открытыми исходными кодами сделала нейросеть, которую обучили на картинках из Википедии. Нейросеть нарисовала картину "Портрет Эдмонда Белами". Картину продали на престижном аукционе Christies за 432 тысячи долларов. Для Christies это была первая проданная картина, нарисованная искусственным интеллектом.

Искусственные нейронные сети

Нейронные сети применяются и **для сочинения стихов и музыки**. Программисты **Яндекса** разработали нейронную сеть, которая сочинила стихи в стиле Егора Летова. Музыку для этих стихов сочинили и сыграли также сотрудники Яндекса, в результате получился альбом «Нейронная оборона», который можно послушать на Яндекс Музыке. Разработчики Яндекса продолжили заниматься этим направлением и создали нейронную сеть, способную сочинять музыку, в том числе классическую.

Нейронная сеть обучалась на произведениях Баха, Шнитке и Рахманинова и генерировала мелодичные линии. Их дорабатывали композиторы-люди. Одно из таких произведений, симфонию «Digital Sunrise», сочиненную совместно нейросетью и композитором Кузьмой Бодровым, сыграл оркестр Юрия Башмета. Другие композиции есть в альбоме «Нечеловеческая музыка» на Яндекс.Музыке.

Искусственные нейронные сети

Технологии обработки изображений нейронными сетями стали настолько мощными, что позволяют создавать изображения, не отличимые от реальных. Один из первых известных примеров – видео с бывшим президентом США Бараком Обамой, которое создали сотрудники университета Вашингтона (**https://www.youtube.com/watch?v=cQ54GDm1eL0**). Видео, с выступлением Барака Обамы сгенерировано нейронной сетью, но выглядит как настоящее.

Искусственные нейронные сети

Технологии обработки изображений нейронными сетями стали настолько мощными, что позволяют создавать изображения, не отличимые от реальных. Один из первых известных примеров – видео с бывшим президентом США Бараком Обамой, которое создали сотрудники университета Вашингтона (**https://www.youtube.com/watch?v=cQ54GDm1eL0**). Видео, с выступлением Барака Обамы сгенерировано нейронной сетью, но выглядит как настоящее.

Слова для видео произносит актер. Нейронная сеть заменяет голос актера на голос Барака Обамы, и генерирует движения губ, подходящие для этих слов. Такая **технология** называется **DeepFake** – комбинация из Deep Learning (обучение глубоких нейронных сетей) и Fake (фальшивка). Сейчас технологии DeepFake стали еще более доступны. К примеру, сейчас популярны **тесты DeepFake радар**, в которых нужно определить, является ли фотография человека реальной или сгенерированной нейронной сетью.

Искусственные нейронные сети

Часто возникает вопрос, почему глубокие нейронные сети стали так популярны именно сейчас. Ведь модель искусственного нейрона придумали еще в 1943 года, и большая часть идей глубоких нейронных сетей было предложено в прошлом веке. У роста популярности есть **несколько причин**.

- Сейчас резко возросла производительность компьютеров. Современные многоядерные процессоры обеспечивают высокую производительность. А некоторые графические ускорители, которые разработаны для компьютерных игр, но могут также применяться для обучения нейронных сетей, по производительности сравнимы с суперкомпьютерами 80-х или 90-х годов. Таким образом, оборудование, которое раньше было только у крупных корпораций и ученых, теперь доступно всем.
- Чтобы обучить нейронную сеть решать сколько-нибудь сложные задачи, нужны большие объемы данных. Сейчас у нас такие объемы данных есть. В совокупности с ростом вычислительных ресурсов появляются технические возможности обучать глубокие нейронные сети.

Искусственные нейронные сети

- В последние несколько лет разработано большое количество улучшений алгоритмов обучения нейронных сетей. Эти улучшения не такие значительные, как создание модели искусственного нейрона, но играют большую роль именно при обучении нейронных сетей для практических задач. Они существенно сокращают время обучения, при этом увеличивая качество обучения.
- И последняя причина – появление и широкое распространение готовых к использованию библиотек нейронных сетей, в которых уже реализованы современные алгоритмы обучения. Таким образом, есть возможность применять эти готовые библиотеки для решения практических задач.

Поэтому порог вхождения в обучение глубоких нейронных сетей снижается, а скорость разработки решений увеличивается. Сейчас даже школьники способны за несколько дней написать программу обучения нейронной сети.

Искусственные нейронные сети

Сфера ИИ – это автоматизация задач, ранее решаемых лишь человеческим интеллектом. Система машинного обучения обучается, но не программируется в явном виде. На вход система принимает многочисленные и часто разнородные примеры, имеющие отношение к решаемой задаче, а она ищет во входных примерах некую статистическую структуру. В свою очередь, данная структура позволяет алгоритму определить правила для автоматического решения поставленной задачи.

Глубокое обучение – это особый раздел машинного обучения, который делает упор на анализ последовательных слоев (или уровней) все более значимых представлений. Под **глубиной** в глубоком обучении подразумевается не более глубокое понимание, которое достигается этим подходом, а многослойность в представлении модели. Количество слоев, из которых состоит модель данных, называют глубиной модели. Иными подходящими названиями для этой части машинного обучения могли бы быть «многослойное обучение» и «иерархическое обучение».

В глубоком обучении многослойные представления изучаются (чаще всего) с применением моделей ИНС. Их структура представлена в виде слоев, наложенных друг на друга.

Искусственные нейронные сети

Глубокое обучение достигло множества прорывов в сложных для машинного обучения областях:

- классификация изображений на уровне человека;
- распознавание речи на уровне человека;
- распознавание рукописного текста на уровне человека;
- повышение качества машинного перевода с одного языка на другой;
- улучшение качества чтения текста вслух машиной;
- создание цифровых помощников: Yandex – Алиса, Google Now и Amazon – Alexa;
- управление автомобилем на уровне, сравнимом с человеком;
- повышение точности целевой рекламы Google, Baidu и Bing;
- повышение релевантности поиска в Интернете;
- машины могут отвечать на вопросы, заданные вслух;
- алгоритм обыграл человека в «Го».

Искусственные нейронные сети

Глубокие нейронные сети **автоматизировали** одну из сложных задач машинного обучения – **выбор признаков**, значимых для решения задачи, из множества доступных. Глубокие нейронные сети автоматически извлекают важные признаки в процессе обучения. Но для обучения ИНС требуются очень большие вычислительные ресурсы. Однако обученная ИНС работает быстро и требует очень мало ресурсов (способна работать даже на мобильном устройстве). К тому же ИНС дает еще одно преимущество перед другими алгоритмами машинного обучения – она **может дообучаться** на отдельных примерах без полного переобучения на всех данных.

Методика глубокого обучения имеет **две важные характеристики**:

- Поэтапно, послойно конструирует все более сложные представления.
- Исследует промежуточные представления совместно, за счет чего каждый слой обновляется в соответствии с информацией, полученной от представлений других слоев.

Искусственные нейронные сети

Различные подходы к машинному обучению и нейронным сетям можно распределить в хронологическом порядке следующим образом:

- вероятностное моделирование (позволило, к примеру, решить задачу определения спама);
- полносвязные нейронные сети (сеть LeNet смогла распознать рукописные цифры);
- ядерные методы (в частности метод опорных векторов);
- деревья решений и основанные на них случайные леса и градиентный бустинг (долгое время оставляли нейронные сети на втором месте);
- современные глубокие нейронные сети Inception, ResNet, BERT (вышли в задачах классификации изображений и текстовой информации на уровень человека).

Искусственные нейронные сети

В целом машинное обучение держится на трех «китах»:

- оборудование;
- наборы данных и тесты;
- алгоритмические достижения.

Оборудование

В 2007 году компания NVIDIA выпустила **CUDA** – программный интерфейс для производимых ими графических процессоров GPU. Теперь несколько GPU могут заменить мощные кластеры на обычных процессорах в ряде задач, которые допускают возможность массового распараллеливания расчетов. Глубокие нейронные сети выполняют (в основном) умножение множества маленьких матриц, поэтому допускают высокую степень распараллеливания.

В 2016 году на своей ежегодной конференции Google I/O компания Google презентовала свой проект тензорного процессора (**Tensor Processing Unit, TPU**). Данный процессор с новой архитектурой предназначается для использования в глубоких нейронных сетях. Он более чем в 10 раз производительнее и энергоэффективнее самых лучших моделей GPU.

Искусственные нейронные сети

В целом машинное обучение держится на трех «китах»:

- оборудование;
- наборы данных и тесты;
- алгоритмические достижения.

Наборы данных и тесты

Данные – это своего рода «сырье», питающее интеллектуальные машины и глубокое обучение в частности. А экспоненциальный рост емкости устройств хранения информации, наблюдавшийся в последние 20 лет (согласно закону Мура), и перемены в игровом мире вызвали бурный рост интернет-данных, благодаря чему удалось накопить и распространить огромные объемы данных для машинного обучения (нужно отметить, что глубокие нейронные сети требуют сотни тысяч, а иногда и миллионы обучающих примеров). Сейчас крупные компании работают с наборами изображений, видео и текстовых материалов, которые невозможно было бы собрать без Интернета.

Искусственные нейронные сети

Алгоритмы

Нейронные сети до конца 2000-х были не очень глубокими – имели один или два слоя представления, из-за этого они уступали более совершенным поверхностным методам, таким как метод опорных векторов и случайные леса. Основная проблема заключалась в распространении градиента через глубокие пакеты слоев. Сигнал обратной связи, который используют для обучения нейронных сетей, затухает по мере увеличения количества слоев.

В 2009–2010 годах появились простые, но важные алгоритмические усовершенствования, позволившие улучшить распространение градиента:

- усовершенствованные функции активации;
- более оптимальные схемы инициализации весов, начиная с предварительного послойного обучения, от которого быстро отказались;
- лучшие схемы оптимизации, такие как RMSProp и Adam.

Эти улучшения позволили строить модели с 10 слоями и более. Начался расцвет глубокого обучения. В 2014, 2015 и 2016 годах были открыты еще более совершенные способы распространения градиента: **пакетная нормализация**, **обходные связи** и **отделимые свертки**. В настоящее время можно обучать с нуля модели с тысячами слоев в глубину.

Искусственные нейронные сети

Библиотека обучения глубоких нейронных сетей на языке Python **TensorFlow** от компании Google распространяется бесплатно и используется во многих компаниях и университетах мира.

С помощью библиотеки TensorFlow и высокоуровневого интерфейса описания нейронных сетей **Keras** из состава TensorFlow можно разрабатывать программы на Python, которые анализируют табличные данные, изображения и тексты.

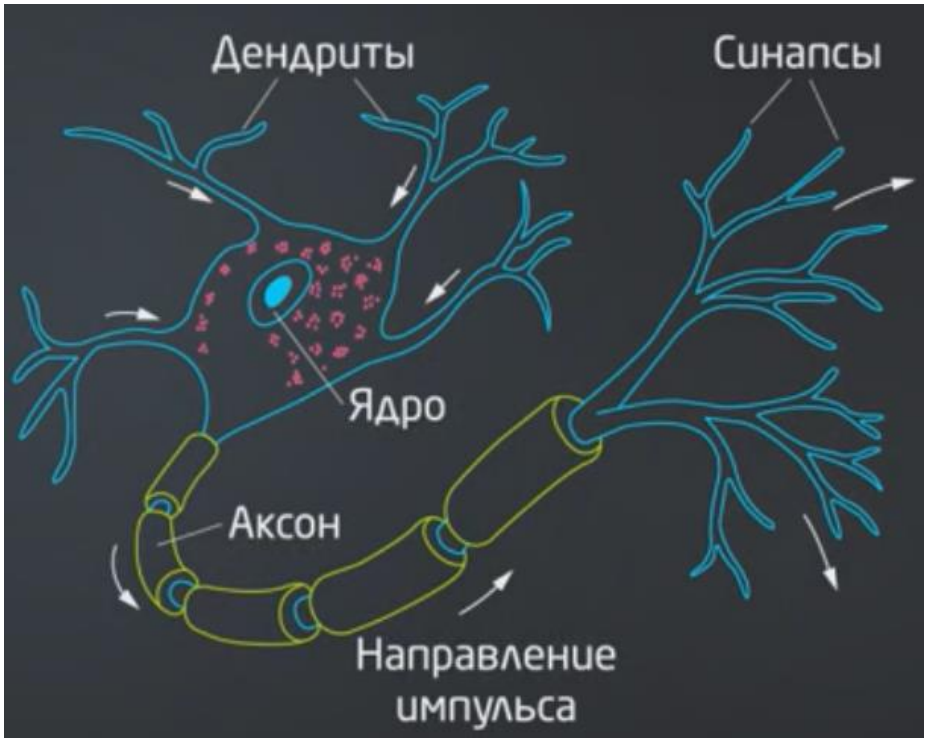
Для обучения нейронных сетей можно использовать бесплатную **облачную платформу Google Colaboratory**. На этой платформе уже установлено большое количество библиотек машинного обучения и анализа данных (в том числе TensorFlow), а также есть GPU, которые позволяют обучать нейронные сети значительно быстрее, чем на центральных процессорах.

Модель искусственного нейрона

Искусственные нейронные сети (ИНС) состоят из небольших вычислительных элементов – искусственных нейронов, которые построены по аналогии, достаточно поверхностной, с нейронами головного мозга.

Биологический нейрон состоит из ядра, в котором накапливается заряд. Заряд поступает в ядро через входные отростки – **дендриты**. При накоплении заряда выше определенного уровня, нейрон возбуждается и выдает нервный импульс на выходной отросток – **аксон**. Аксон, в свою очередь, подключен к дендритам следующих нейронов, за счет таких соединений образуется **нейронная сеть**.

Аксоны соединяются с дендритами через **синапсы**. Синапсы могут изменять проходящий через них сигнал. Возбуждающие синапсы усиливают сигнал, а тормозящие – ослабляют.



Модель искусственного нейрона

Модель искусственного нейрона предложили двое ученых Уоррен Мак-Каллок и Уолтер Питтс в работе «Логические исчисления идей, относящихся к нервной деятельности» в 1943 году.

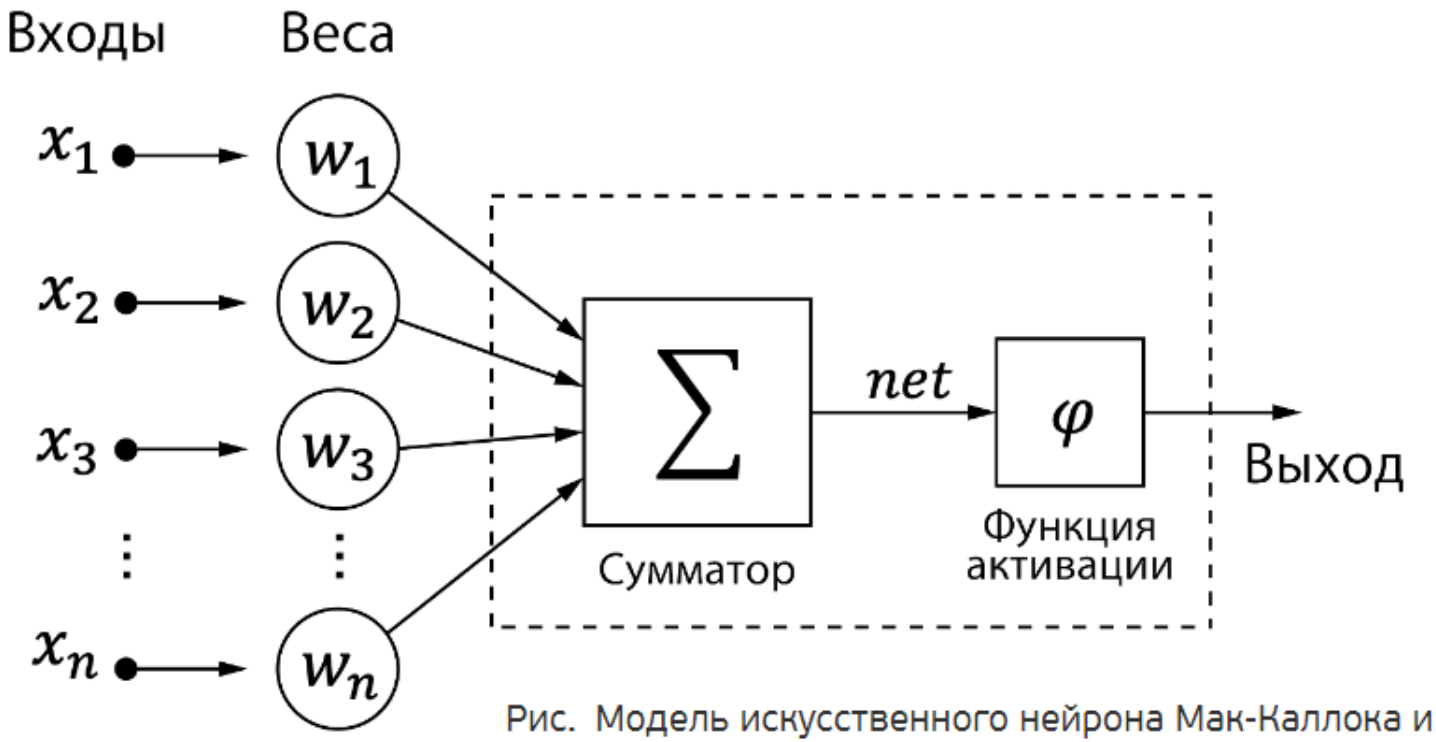


Рис. Модель искусственного нейрона Мак-Каллока и Питса

В этой модели у нейрона есть входы, аналоги дендритов. Количество таких входов n. На входы поступают сигналы, вещественные числа, от x_1 до x_n, описывающие наши объекты. Например, для изображения входными сигналами будут числовые значения пикселей.

Модель искусственного нейрона

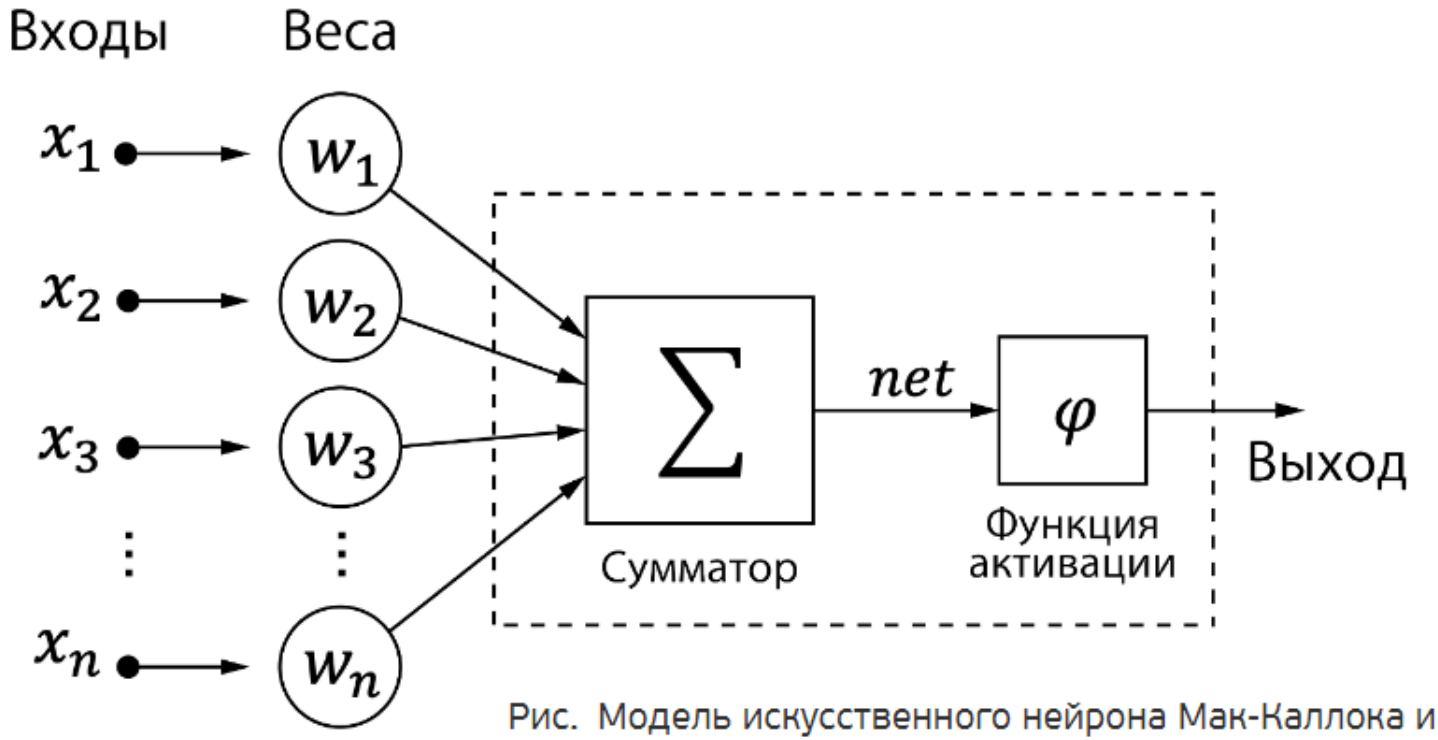


Рис. Модель искусственного нейрона Мак-Каллока и Питса

Входные значения умножаются на веса входов от w_1 до w_n. Веса входов в искусственный нейрон – это аналоги синапсов в биологическом нейроне. Положительные веса действуют как возбуждающие синапсы, а отрицательные – как тормозящие. Значения сигналов на входе в нейроны, умноженные на веса входов, складываются и поступают на вход нелинейной функции φ. Эта функция называется функцией активации. Выход искусственного нейрона является аналогом аксона.

Модель искусственного нейрона

В математическом виде **модель искусственного нейрона** можно описать:

- **Входы** – это некий вектор $X = [x_1, x_2, \ldots, x_n]$ входных значений, они являются аналогом дендритов в реальном нейроне.
- **Веса** – это вектор обучаемых параметров нейронной сети $\mathrm{w} = [\mathrm{w}_1, \mathrm{w}_2, \ldots, \mathrm{w}_n]$, которые обозначают значимость каждого входного признака для итогового результата (по сути, аналог синапсов реального нейрона). В конечном итоге, если какой-то вес $\mathrm{w}_i = 0$, то признак x_i является абсолютно незначимым для нейронной сети. И наоборот, чем больше значение w_i, тем сильнее влияет признак x_i на результат, выданный нейронной сетью. Длина вектора весов совпадает с длиной вектора входных параметров.
- **Сумматор** – функция, которая просто суммирует произведения признаков на их веса. Можно сказать, что это ядро нейрона, которое аккумулирует заряд:

$$x_1\mathrm{w}_1 + x_2\mathrm{w}_2 + \ \ldots\ + \ x_n\mathrm{w}_n = \sum_{i=1}^{n} x_i\mathrm{w}_i.$$

- **Функция активации** – функция, которая принимает на вход результат сумматора и выполняет некое преобразование, чтобы превратить сумму взвешенных входов в адекватный выход, который можно интерпретировать с точки зрения решения поставленной задачи. Так как в случае малой величины сумматорной функции функция активации может вернуть 0, то есть «ничего», то она является неким аналогом того механизма в реальном нейроне, которые отвечает за возбуждения нейрона (его «активацию»).

Модель искусственного нейрона

Функции активации

Итак, предположим, что нам нужно выполнить классификацию объектов на две группы: съедобные грибы и несъедобные. Взвешенная сумма входов может быть любым числом в диапазоне от минус бесконечности до плюс бесконечности, что никак не приближает нас к решению поставленной задачи. Значит, необходимо как-то преобразовать это число так, чтобы оно обязательно попадало в строго заданный нами диапазон, и именно в рамках этого диапазона мы могли принимать решение. Для этого были разработаны функции активации.

Функция единичного скачка (функция Хэвисайда):

$$\theta(x) = \begin{cases} 0, x < 0; \\ 1, x \geqslant 0. \end{cases}$$

где x – взвешенная сумма входных параметров.

Данная функция принимает на вход взвешенную сумму входных параметров и, если они меньше 0, то возвращает 0, иначе – 1.

Модель искусственного нейрона

Но что, если мы видим, что корректное разделение классов будет происходить не по нулю, а по любому другому значению?

Тут пригодится понятие **«порог»** (от англ. bias) – это значение, которое будет смещать место скачка от 0 в нужное нам место числовой оси:

$$f(S) = \begin{cases} 0, S < \theta \\ 1, S \geq \theta \end{cases}$$

где θ – заданный порог,

S – взвешенная сумма входных параметров.

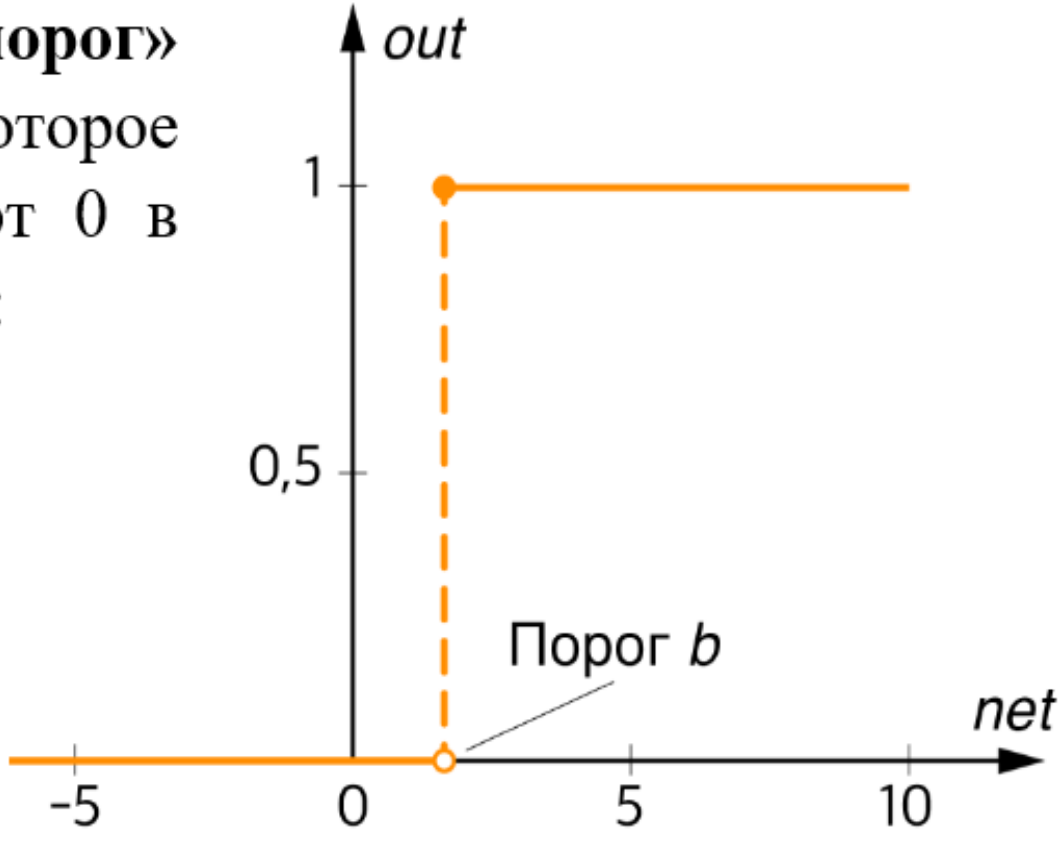


Рис. Функция Хэвисайда (функция единичного скачка)

Искусственные нейронные сети

Но такое строгое разделение на 0 и 1 является слишком грубым и в реальности не используется, так как часто возникают ситуации, когда о принадлежности к классу можно сказать лишь с некоей долей вероятности. Поэтому нужна функция, которая позволит выдавать не 0 и 1, а число в этом диапазоне. И чем ближе число к 0, тем больше вероятность, что это, к примеру, несъедобный гриб, а чем ближе к 1, тем больше вероятность, что гриб съедобен.

Сигмоидальная функция (логистическая функция):

$$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$$

где x – взвешенная сумма входных параметров.

Кроме того, что она возвращает необходимое нам распределение значений от 0 до 1, данная функция обладает еще одним интересным свойством – она «прижимает» распределения получаемых значений к 0 и к 1, а в центре около 0,5 старается оставить как можно меньше значений. Это происходит за счет очень резкого возрастания функции на участке вокруг 0,5 и плавных переходов около 0 и 1. Таким образом, мы получаем минимальное количество значений, которые попадают в зону неопределенности около 0,5.

Искусственные нейронные сети

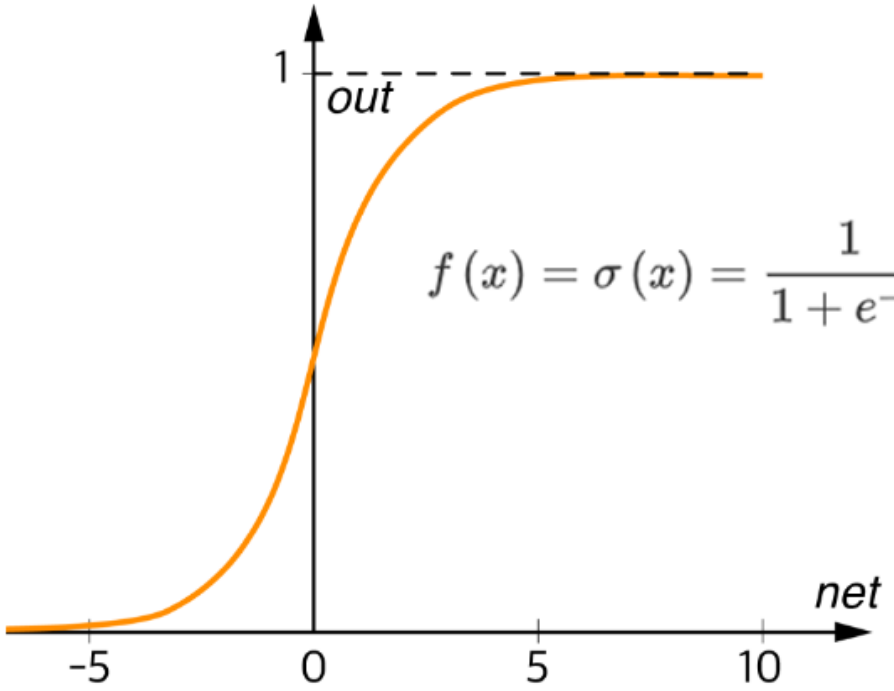


Рис. Сигмоидальная функция (логистическая функция)

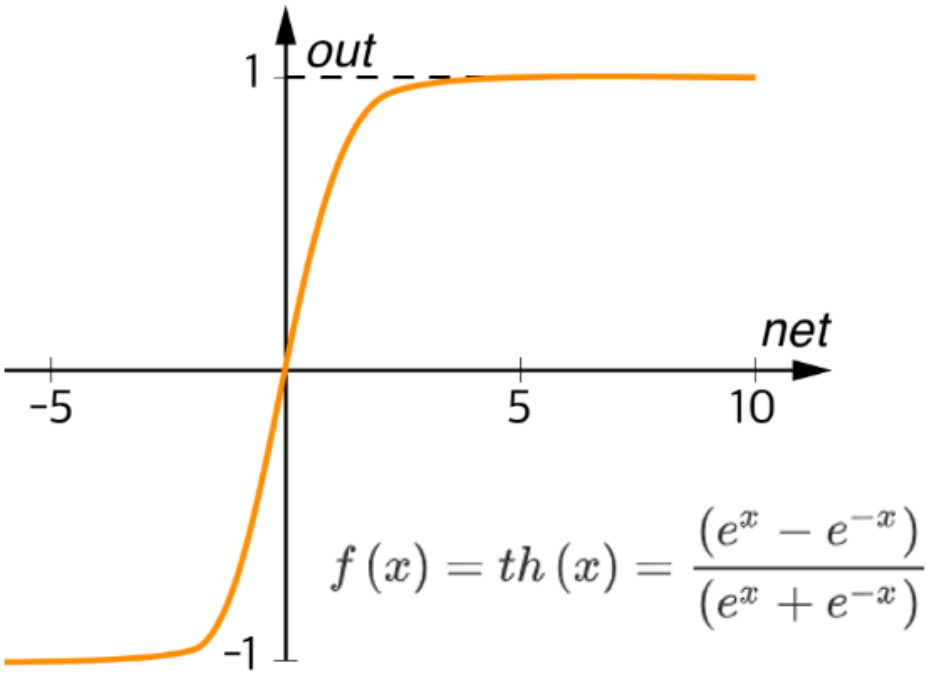


Рис. Функция гиперболического тангенса

Функция гиперболического тангенса (rectified linear units, ReLU)

Данная функция ведет себя похожим образом с сигмоидой и обладает всеми ее свойствами, но возвращает значения в диапазоне от -1 до 1. Иногда это может быть удобно.

$$f(x) = th(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$$

Эта функция активации возвращает значение x, если x положительно, и 0 в противном случае: $f(x) = max(0, x)$

Искусственные нейронные сети

Функция гиперболического тангенса (rectified linear units, ReLU) является хорошим аппроксиматором – любая функция может быть аппроксимирована комбинацией ReLU. **Преимуществом** также является то, что эта функция создает разреженность при активации нейронов. То есть, в отличие от сигмоиды и гиперболического тангенса, будут активироваться не все нейроны, что уменьшит вычислительную трудоемкость. Это свойство вытекает из того факта, что все отрицательные значения обращаются в ноль. Использование ReLU значимо **повышает скорость сходимости** градиентного спуска по сравнению с сигмоидой и гиперболическим тангенсом.

Но есть и **недостатки**. Все из-за того же обнуления всех отрицательных значений градиент на этой части также равен 0. Поэтому веса не будут корректироваться во время градиентного спуска. А значит нейроны, пребывающие в таком состоянии, не будут реагировать на ошибки и обучаться. Данное явление также называют **проблемой умирающего ReLU**.

Существуют **модификации ReLU**, которые частично решают эту проблему: Leaky ReLU, Parametric ReLU, Randomized ReLU.

Искусственные нейронные сети

Мак-Каллок и Питтс предложили не просто модель искусственного нейрона, но и способ объединения нескольких таких нейронов в искусственные нейронные сети. Для этого выход нейрона подключается ко входу другого нейрона (или нескольких других нейронов). Было предложено много различных методов организации нейронных сетей, сейчас чаще всего используется подход, при котором нейроны организованы в слои. Это тоже имеет аналогию с устройством головного мозга. Выходы нейронов одного слоя подключены ко входам нейронов следующего слоя.

В нейронных сетях есть **три типа слоев**:

- **Входной** слой, на который поступают сигналы из внешнего мира
- **Выходной** слой, который выдает сигналы во внешний мир
- **Скрытый** слой, который находится между входным и выходным слоями. Слой называется скрытым, т.к. по логике работы нейронной сети мы не можем знать, что там происходит. Однако на практике программные реализации нейронных сетей позволяют получить доступ ко всему, что происходит в скрытом слое. Скрытых слоев может быть несколько.

Классификация ИНС

По типу распространения сигнала искусственные нейронные сети (ИНС) можно разделить на два типа:

- нейронные **сети с прямым распространением сигнала**;
- нейронные сети с наличием по крайней мере одной обратной связи (**рекуррентные** нейронные сети).

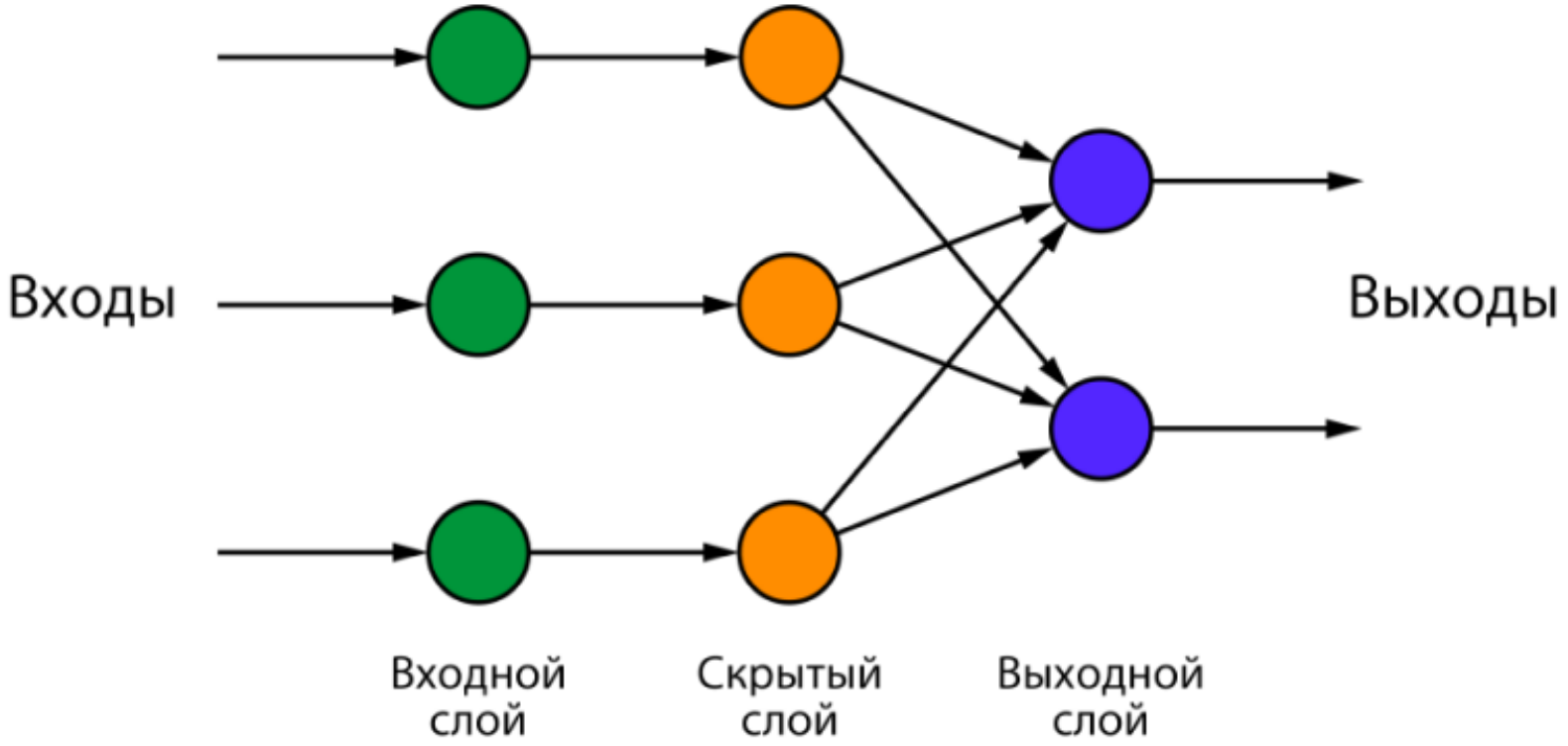


Рис. Нейронная сеть с прямым распространением сигнала

Классификация ИНС

Сети с прямым распространением сигнала решают множество задач классификации и регрессии на табличных данных, а также практически все задачи на изображениях и часть задач обработки естественного языка.

Необходимость в **рекуррентных нейронных сетях** возникает при решении таких задач, когда требуется владеть информацией о предыдущих элементах обучающей последовательности: временные ряды, обработка текста с учетом последовательности слов.

В этом случае выход нейрона может быть соединен с его входом, со входами нейронов на том же слое, или теоретически с любым предыдущим нейроном в сети.

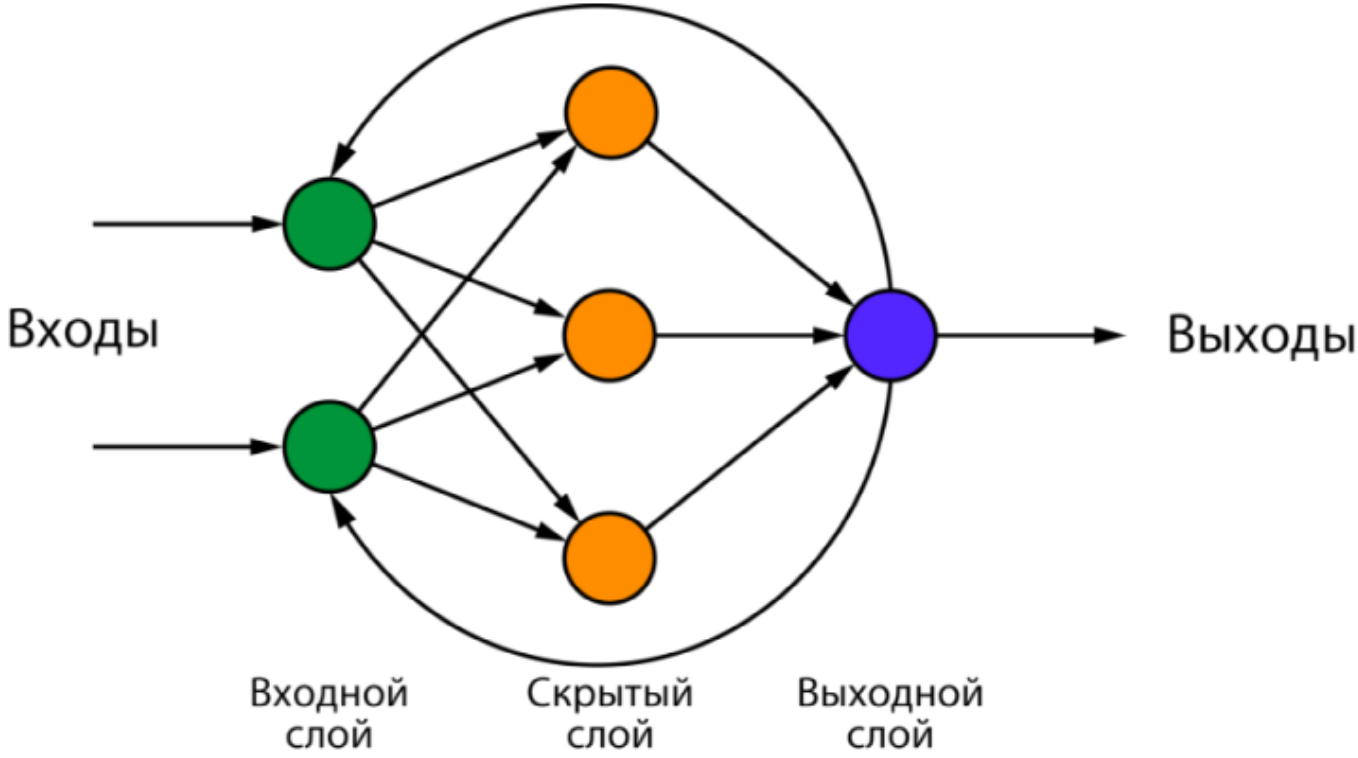


Рис. Нейронная сеть с обратной связью (рекуррентная нейронная сеть)

Классификация ИНС

Однозначного определения, что такое глубокая нейронная сеть не существует. Более того, понятие глубокой нейронной сети меняется со временем. Раньше глубокой считалась такая нейронная сеть, у которой более одного скрытого слоя. Нейронные сети, дающие достаточно хорошие результаты на задачах компьютерного зрения содержат 16 или 19 слоев. Однако сейчас есть сети, включающие сто и более скрытых слоев. Поэтому точно определить, где находится граница, отделяющая «глубокую» нейросеть от «неглубокой», невозможно. Вы можете выбрать для себя любое устраивающее значение, например, 10 или 100 скрытых слоев.

Глубокими нейронными сетями чаще всего считают такие нейросети, у которых **больше одного скрытого слоя**. Для таких сетей необходимы специальные алгоритмы обучения, которые подходят как для сетей с двумя скрытыми слоями, так и для сетей, у которых 100 и более скрытых слоев.

Классификация ИНС

Также архитектуры нейронных сетей можно разделить на однослойные и многослойные (они же глубокие) нейронные сети.

Однослойной нейронной сетью считается сеть, имеющая лишь входной и выходной слой, без наличия скрытых слоев. В такой сети сигналы попадают сразу с входного слоя на выходной. Сеть подобного рода способна настроиться не под все сложные зависимости, очень чувствительна к предварительной обработке признаков и требует их тщательного отбора. Преимуществом является достаточно быстрая скорость обучения.

Многослойной (глубокой) нейронной сетью считается сеть с минимум двумя скрытыми слоями (входной и выходной, естественно, также остаются). Способна самостоятельно от слоя к слою преобразовывать входные признаки и проводить их отбор за счет уменьшения/увеличения весов. Но наличие большого количества слоев, а значит и обучаемых параметров, также ведет к увеличению времени обучения (некоторые нейронные сети обучались по несколько месяцев). А еще из-за очень сложной структуры данный тип сетей подвержен переобучению при малом количестве обучающих данных. То есть нейронная сеть идеально подстраивается под обучающий набор, но в дальнейшем, на реальных данных, показывает очень плохое качество.

Обучение ИНС

В машинном обучении **под обучением** понимается подбор параметров модели таким образом, чтобы модель лучше всего описывала имеющиеся данные. Параметрами искусственной нейронной сети являются веса входов в нейроны w_i. Именно **в подборе правильных значений весов** и заключается обучение нейронной сети.

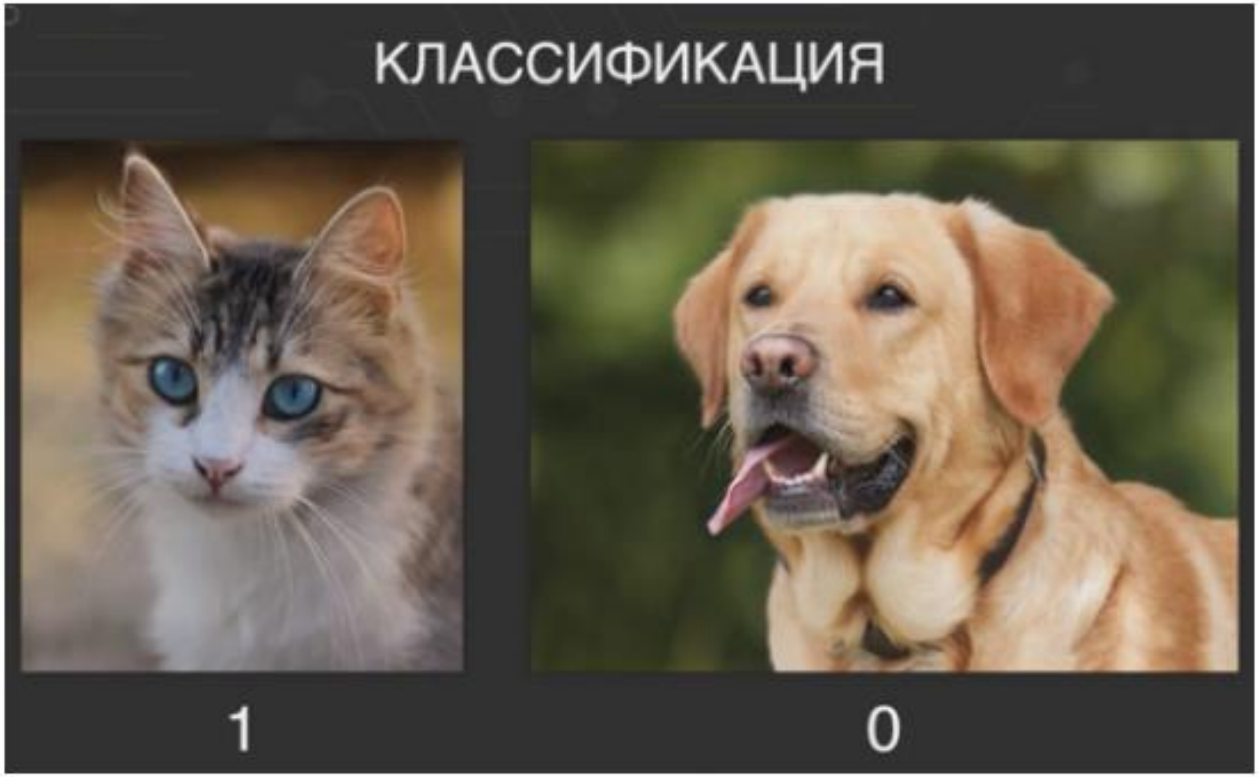
В первоначальном варианте, который предложили Мак-Каллок и Питтс, нейронные сети не обучались. Веса входов в нейроны нужно было подбирать вручную. Идея обучения описана Дональдом Хеббом в книге 1949 года «Организация сознания». Она основана на особенностях работы мозга человека и, упрощенно, заключается в том, что когда два нейрона в мозге часто срабатывают вместе, то связь между ними усиливается. В противном случае связь становится слабее. Первые правила обучения искусственного нейрона, основанные на таком подходе, называются **правила Хебба**.

Обучение ИНС

Какие же задачи может решать искусственная нейронная сеть? Чаще всего встречаются две типовые для машинного обучения задачи: классификация и регрессия.

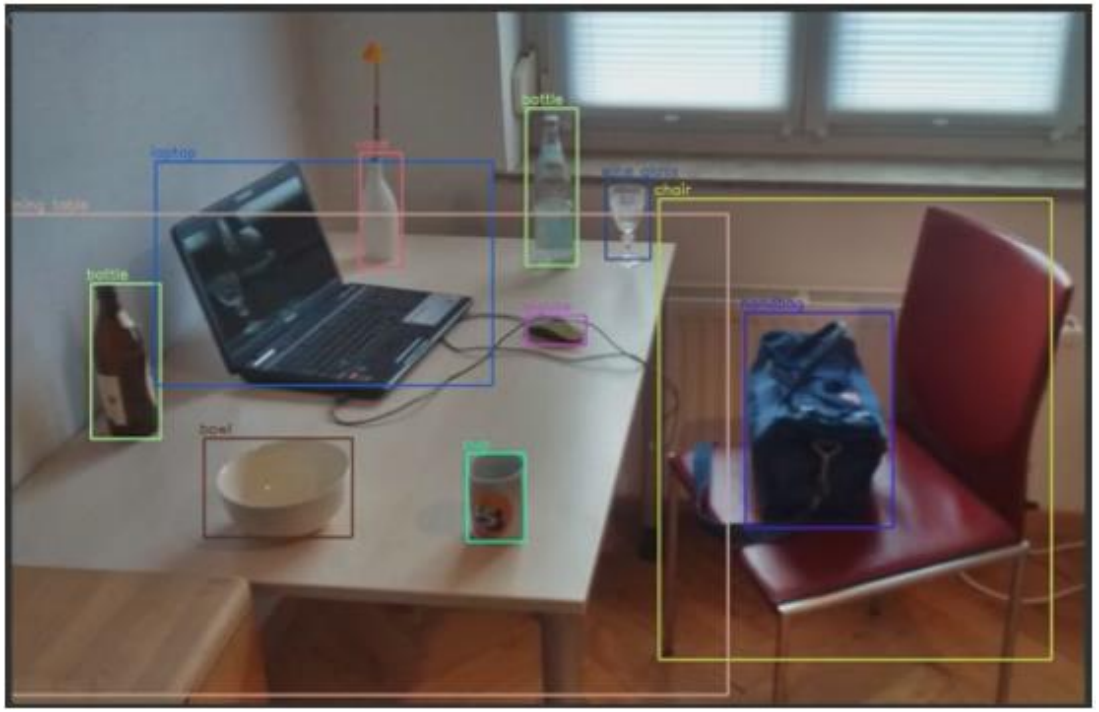
В задаче классификации у нас есть несколько классов объектов, например, фотографии котов и собак. Нейронная сеть должна научиться определять, к какому классу относится объект: кот на фотографии или собака.

Особенность такого типа задач в том, что на выходе из нейросети дискретные значения, соответствующие количеству классов. Например, если на фотографии собака, то сеть выдает значение 0, а если кот – то значение 1.



Обучение ИНС

В задаче регрессии на выходе нейронной сети непрерывное значение. Это может быть, например, стоимость акций, недвижимости, объем продаж интернет-магазина или ресторана.



Нейронные сети могут решать и другие, более сложные задачи, например, поиск объекта на изображении, сегментация изображения, автоматическая генерация текста. Но это комбинации базовых задач классификации и регрессии.

Обучение ИНС

Есть три типа обучения искусственных нейронных сетей:

1. **Обучение с учителем** – в этом случае необходим набор данных для обучения, в котором для каждого объекта указан правильный ответ. Такой набор данных называется **размеченным**, а правильные ответы – **метками**. Обычно для обучения нейронной сети нужны тысячи или десятки тысяч изображений, но это не всегда так. Для обучения нейронных сетей чаще всего используется обучение с учителем.

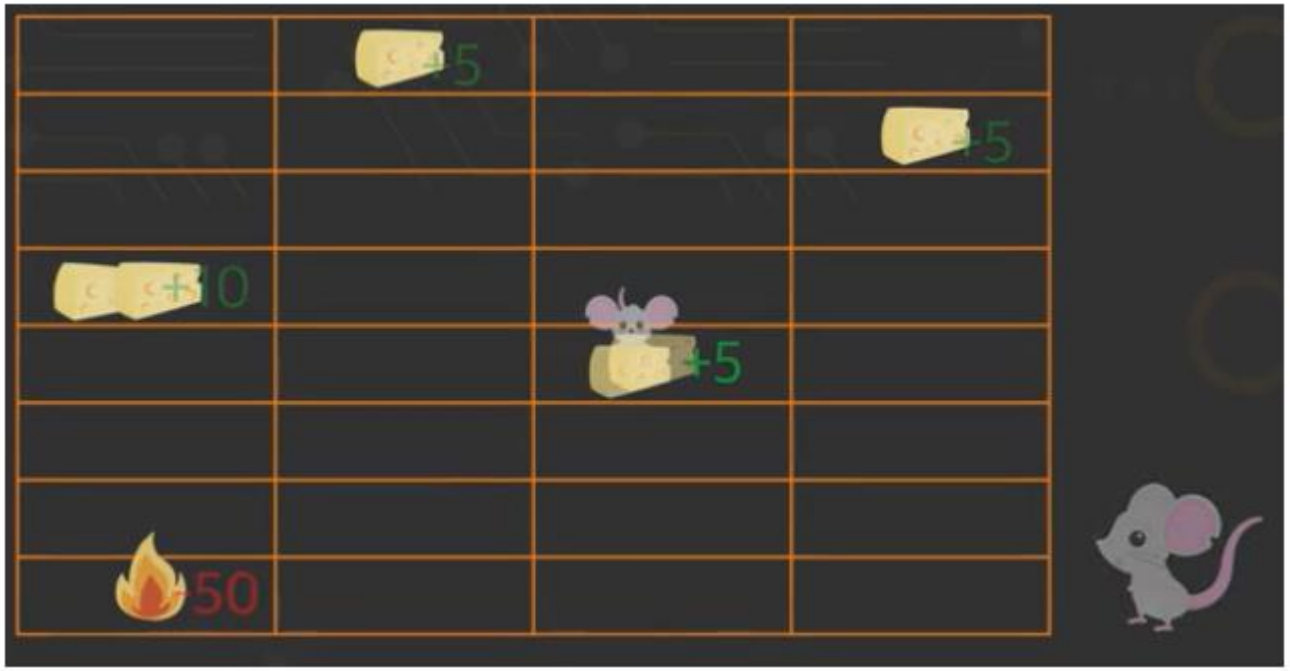
2. **При обучении без учителя** используется набор данных, для которых не указаны правильные ответы. В этом случае нейронная сеть может найти закономерности в структуре данных. Алгоритмы обучения без учителя значительно сложнее, чем обучение с учителем.

Например, нейронная сеть может определить, что фотографии собак похожи друг на друга, а фотографии котов в свою очередь похожи на фотографии других котов. Нейронная сеть может разбить их на отдельные группы, которые называются **кластерами**.

Обучение ИНС

Есть три типа обучения искусственных нейронных сетей:

3. При обучении с подкреплением нейронная сеть обучается за счет взаимодействия со средой. От среды поступают подкрепления, которые могут быть положительными или отрицательными. Обучение с подкрепление часто применяется для игр. Например, у нас есть мышь, которая бегает по лабиринту. В лабиринте есть сыр, мышь получает положительное подкрепление, если попадает на клетки с сыром. Однако если мышь попадет на клетку с огнем, то она получает большое отрицательное подкрепление, соответствующее окончанию игры.



Обучение ИНС

Итак, нейронная сеть представляет собой следующий набор: входные сигналы; веса, которые как-то учитывают силу влияния входных сигналов; нейроны, которые преобразовывают входные сигналы и передают их следующим нейронам или на выходной слой. Функций, которые преобразовывают сигнал (функций активации), существует множество и выбирать их необходимо исходя из задачи. Также можно создавать свои функции активации. Количество нейронов в слое и слоев в нейронной сети можно варьировать как угодно. Из чего следует, что нейронная сеть – достаточно гибкий инструмент.

Но основная сила нейронной сети не в этом. Даже при всех вышеописанных плюсах, если бы веса нейронной сети устанавливались один раз в начале обучения (как это происходит с количеством слоев и нейронов в слое, а также с функциями активации), то был бы получен крайне ограниченный инструмент, неспособный найти обобщенное решение для множества разнородных наблюдений. В лучшем случае нейронная сеть запоминала бы правильный ответ на одно из наблюдений (к примеру, запомнила бы, что на этой конкретной фотографии кошка), но уже следующее наблюдение, с немного измененными параметрами, заставило бы нейронную сеть ошибиться.

Обучение ИНС

Сила нейронных сетей в обучении весов связей между нейронами. В ходе множества итераций нейронная сеть сравнивает свой выход с настоящим ответом и, в случае ошибки, корректирует все свои веса таким образом, чтобы на следующей итерации добиться верного ответа. Алгоритм такого обучения называется **«алгоритм обратного распространения ошибки»**. Это связано с тем, что нейронная сеть понимает, ошиблась она или нет, только на выходе, и затем информацию об ошибке она распространяет в обратном порядке от выхода ко входу.

Чем больше в нейронной сети слоев и нейронов, тем больше связей (а значит, и весов, эти связи характеризующих) и тем более тонкой может быть настройка нейронной сети при обучении. Но также выше будет опасность подкрутить веса так, что они идеально будут решать задачу на обучающих данных, но начнут выдавать абсолютно неверный ответ на новых данных из реального мира. В этом случае говорят, что нейронная сеть **переобучилась**.

1. Обучение ИНС с учителем

Это достаточно привычный нам способ, когда кто-то проверяет. В случае с нейронной сетью создаются специальные учебные наборы данных, в них парами идут входные данные и ответ (см. таблицу 1). Наборы формируются людьми (это может быть сам исследователь или нанятые им сторонние разметчики данных).

Примеры формирования тренировочного набора для обучения с учителем

Таблица 1

Прикладная сфера	Входные данные	Верный ответ
Классификация изображений	Матрицы изображений рукописных цифр (одинаковой размерности)	Число от 0 до 9 – цифра, которая реально написана на изображении
Классификация текстов	Вектор, полученный из текста новостной ленты	Тема новостей: спорт, политика, финансы и т. д.
Регрессия	Вектор с параметрами дома: площадь, количество комнат, удаленность от центра и т. д.	Число, обозначающее реальную стоимость дома
Временной ряд	Вектор с последовательным почасовым трафиком посетителей на сайте	Число (или вектор чисел), обозначающее число посетителей в следующий момент времени (ряд последовательных моментов времени)

1. Обучение ИНС с учителем

Таким образом, нейронная сеть получает возможность исправлять свои ошибки, как это делает прилежный ученик, опираясь на авторитетное мнение учителя. Данный класс задач сильнее изучен и дает наиболее качественные результаты.

Сразу же хочется отметить, что из данных таблицы 1 следуют два основных класса задач обучения с учителем:

- **классификация** – множество ответов нейронной сети конечно и представляет собой некий конечный набор классов;
- **регрессия** и **аппроксимация** – множество ответов нейронной сети бесконечно и представляет собой действительное число или вектор действительных чисел.

Временной ряд можно отнести к задаче регрессии, но с условием, что выборка наблюдений берется не случайно, а строго упорядоченно во времени и с одинаковым интервалом. К примеру, цена на конкретный дом фиксировалась на протяжении 10 лет с интервалом в 1 год, а далее нужно предсказать цену данного дома на 11-й год.

2. Обучение ИНС без учителя

Как понятно из названия, тут уже нейронная сеть должна «посмотреть» на данные и сделать какие-то свои выводы, попытаться найти какие-то шаблоны, которые выделяют те или иные части набора данных. Тут заранее стоит предупредить, что шаблоны, найденные нейронной сетью, далеко не всегда могут быть интерпретированы в человеческих терминах, то есть не всегда ясно, по какому признаку нейронная сеть разделила те или иные группы или вывела какие-то правила. К задачам обучения без учителя относятся:

- **кластеризация** – задача разбиения выборки на N (число кластеров может задаваться заранее, а может вычисляться алгоритмом в процессе обучения) кластеров по заранее неведомым признакам. Отличие от классификации в том, что метки классов заранее не заданы и полученные кластеры необходимо затем интерпретировать в язык человеческих терминов;
- **сокращения размерности** – задача представления многомерного исходного вектора в вектор меньшей размерности с минимальной потерей информации. Таким образом может происходить отсев малоинформативных признаков, что уменьшит шум в данных и увеличит скорость обработки данных алгоритмом, а также может повысить качество какой-нибудь последующей задачи (например, классификации). Вырожденные случаем сокращения размерности (до двумерных или трехмерных векторов) можно использовать для визуализации данных.

3. Обучение ИНС с подкреплением

Это направление в машинном обучении, при котором модель не имеет информации о системе, но при этом может производить некие действия, влияющие на систему. При воздействии модели на систему та переходит в новое состояние. В зависимости от того, приближает это модель к желаемой цели или отдаляет от нее, система посылает модели положительное или отрицательное вознаграждение.

Например, есть сеть улиц, по которым едет машина. Модель может управлять скоростью машины и ее поворотами в заданных пределах. Целевую функцию можно сформулировать, как «доехать из пункта А в пункт В за минимальное время», а к тому же можно еще наложить ограничение, запрещающее врезаться в стены. Тогда модель будет управлять машиной, а система будет ее штрафовать за удар о стену или поощрять за оптимально выбранный маршрут.

Обучение ИНС

Есть три варианта практической реализации обучения ИНС.

1. Полное обучение – в этом случае мы обрабатываем весь обучающий набор данных и только после этого изменяем веса нейронной сети. Такой подход работает стабильно, но требует много времени для обучения.

2. Онлайн обучение – при этом веса в нейронной сети изменяются после обработки каждого элемента данных. В этом случае обучение проходит значительно быстрее, но может быть не стабильным. При определенных условиях сеть может вообще не обучиться. Однако у онлайн-обучения есть и определенное преимущество: можно **дообучать** нейронную сеть на новых данных сразу же, как только они появятся.

3. Комбинация двух предыдущих вариантов – это **обучение на мини-выборках**. В этом случае обучающий набор данных делится на отдельные части, которые называются мини-выборки. Изменения весов нейронной сети производится после обработки каждой мини-выборки. За счет этого обучение происходит быстрее, т.к. не нужно ждать обработки всего набора данных для изменения весов. С другой стороны, в каждой мини-выборке достаточно много объектов для обеспечения стабильности обучения. На практике чаще всего используется именно обучение на мини-выборках, при этом размер одной мини-выборки от нескольких десятков до нескольких сотен объектов.

Библиотеки для обучения ИНС

Реализовывать все это многообразие методов и архитектур нейронных сетей с нуля очень трудоемко и нерационально. Поэтому были созданы библиотеки, в которых все основные архитектуры и методы уже реализованы и оптимизированы под параллельные вычисления. Таким образом создание нейронной сети превращается из сложной задачи низкоуровневого программирования и оптимизации в задачу непосредственно экспериментирования с нейронными сетями на высоком уровне абстракции. На уровне слоев и готовых функций активации, оптимизации и обратного распространения ошибки. Исследователь может полностью сосредоточиться на сути нейронных сетей и изучать их поведение при разных конфигурациях. Необходимость опускать на уровень ниже может возникать лишь при опытах с новыми экспериментальными функциями активации, конфигурациями соединения нейронов в слое или слоев между собой. Но начинающий исследователь может абстрагироваться от сложной математики и низкоуровневых программных реализаций, постепенно погружаясь в мир нейронных сетей.

Библиотеки для обучения ИНС

Библиотеки для обучения нейронных сетей

Название библиотеки	Описание
TensorFlow	Базовый язык C++, но имеет API для Python. Разработан Google. Вычисления с использованием графов потоков данных. Предлагает мощные средства мониторинга процесса обучения моделей и визуализации. Поддерживает распределенное обучение. Имеет достаточно высокий входной порог
Theano	Базовый язык Python. Разработан университетом Монреаля. Вычисления с использованием графов потоков данных. Эффективная обработка тензоров. Вычисления выражаются NumPy – подобным синтаксисом. Имеет высокий входной порог
PyTorch	Базовый язык Lua. Поддерживает API для Lua, Python, Java, C++. Разработан Ронаном Коллабертом. Имеет множество модульных элементов, которые легко комбинировать. Легко писать собственные типы слоев и работать на GPU. Имеет API разных уровней абстракции
CNTK	Базовый язык C++. Поддерживает API для C++, C#, Python, Java. Разработана Microsoft. Обеспечивает скорость обучения, сравнимую с TensorFlow, а на рекуррентных сетях превосходит его. Имеет API разных уровней абстракции
Caffe	Базовый язык C++. Поддерживает API для C++, Python. Разработан в центре компьютерного зрения и обучения Беркли. Имеет небольшую скорость относительно других библиотек
Keras	Библиотека верхнего уровня. Поддерживает Python. В качестве вычислительного back-end использует TensorFlow или Theano. Позволяет создавать и обучать нейронные сети на очень высоком уровне абстракции. Имеет низкий порог вхождения

Библиотеки для обучения ИНС

Еще несколько лет назад, чтобы заниматься нейронными сетями необходимо было глубоко знать математику, в первую очередь линейную алгебру, статистику, теорию вероятностей и методы оптимизации.

Обучение нейронных сетей вычислительно трудоемко и занимает много времени, поэтому при их программной реализации важна производительность. Алгоритмы обучения нейронных сетей приходилось реализовывать на C++, который обеспечивает нужную производительность, но сложен в изучении и эффективном применении.

В целях максимального использования возможностей современного вычислительного оборудования для повышения производительности обучения нейросетей, специалист должен был хорошо разбираться в архитектурах параллельных вычислительных систем: многоядерных процессорах, многопроцессорных вычислительных комплексах, ускорителях вычислений. Кроме того, нежно было уметь работать с технологиями больших данных, т.к. для качественного обучения нейронных сетей нужно много данных.

Библиотеки для обучения ИНС

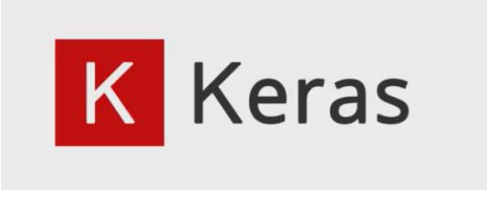
Сейчас ситуация кардинально поменялась. Существует большое количество готовых к использованию библиотек, в которых уже реализованы алгоритмы обучения нейронных сетей. Таким образом, можно брать готовые библиотеки и применять их для решения практических задач, например, распознавания изображений или анализа текстов. Это позволяет достаточно быстро создавать прикладные системы машинного обучения или искусственного интеллекта.

Наиболее популярными библиотеками машинного обучения являются **TensorFlow** от компании Google и **PyTorch** от компании Facebook. У них примерно одинаковые возможности. Библиотека TensorFlow существует дольше, поэтому чаще применяется для создания продуктивных систем. Библиотека PyTorch более новая, она популярна для исследовательских целей. Однако и промышленные системы машинного обучения все чаще создаются на PyTorch. Как TensorFlow, так и PyTorch позволяют создавать нейронные сети на Python.

Библиотеки для обучения ИНС

Кроме TensorFlow и PyTorch существует большое количество других библиотек обучения нейронных сетей, например, **Microsoft CNTK**, **Apache MXNet**, **PaddlePaddle** от китайской компании Baidu. Таким образом, большая часть библиотек машинного обучения создается крупными интернет компаниями, которые активно используют нейронные сети и машинное обучения в своих практических задачах.

Интересна и популярная библиотека нейронных сетей **Keras**, которая позволяет очень удобно и просто описывать нейронные сети и процесс их обучения. При этом для самого процесса обучения Keras использует другие библиотеки в качестве вычислительного бэкенда, включая TensorFlow, CNTK и Theano. Сейчас Keras входит в TensorFlow и является рекомендуемым высокоуровневым программным интерфейсом для описания нейронных сетей.



Библиотеки для обучения ИНС

Существуют также библиотеки, которые оптимизируют обучение нейронной сети на какой-либо аппаратной архитектуре. Например, библиотека **cuDNN** ускоряет обучение нейронных сетей на графических ускорителях GPU, а библиотека **Deep Learning 4 Java** позволяет обучать нейронную сеть на кластере из нескольких серверов. Все библиотеки с открытыми исходными кодами, их можно использовать бесплатно.

Часто можно слышать, что **программы на Python** работают медленно, поэтому многие выражают сомнение в применимости Python для вычислительно интенсивных задач, таких как обучение нейронных сетей. Однако программы на Python компактные и понятные, для обучения простой нейронной сети достаточно несколько десятков строк кода, программа занимает 2-3 экрана. Это позволяет быстро проводить эксперименты по поиску наиболее подходящей для задачи архитектуры нейронной сети.

В настоящее время не существует конструктивного метода создания нейронной сети, которая решала бы поставленную задачу. Поэтому и приходится проводить много экспериментов, прежде чем подходящая архитектура будет найдена. Чем быстрее и проще проводить такие эксперименты, тем быстрее мы получим нужную архитектуру нейросети.

Библиотеки для обучения ИНС

Программа Keras на Python оптимизируется с помощью библиотеки **TensorFlow**, которая выступает в качестве вычислительного бэкенда. Причем особенность работы TensorFlow в том, что она может использовать как обычные центральные процессоры, так и графические процессоры GPU.

GPU – это видеокарты, которые обычно используются для графических приложений и игр. Они очень хорошо подходят для параллельного выполнения большого количества простых математических операций, которые используются в компьютерной графике. При обучении нейронных сетей используются подобные операции, поэтому GPU можно использовать **для ускорения обучения** нейронных сетей в несколько десятков раз.

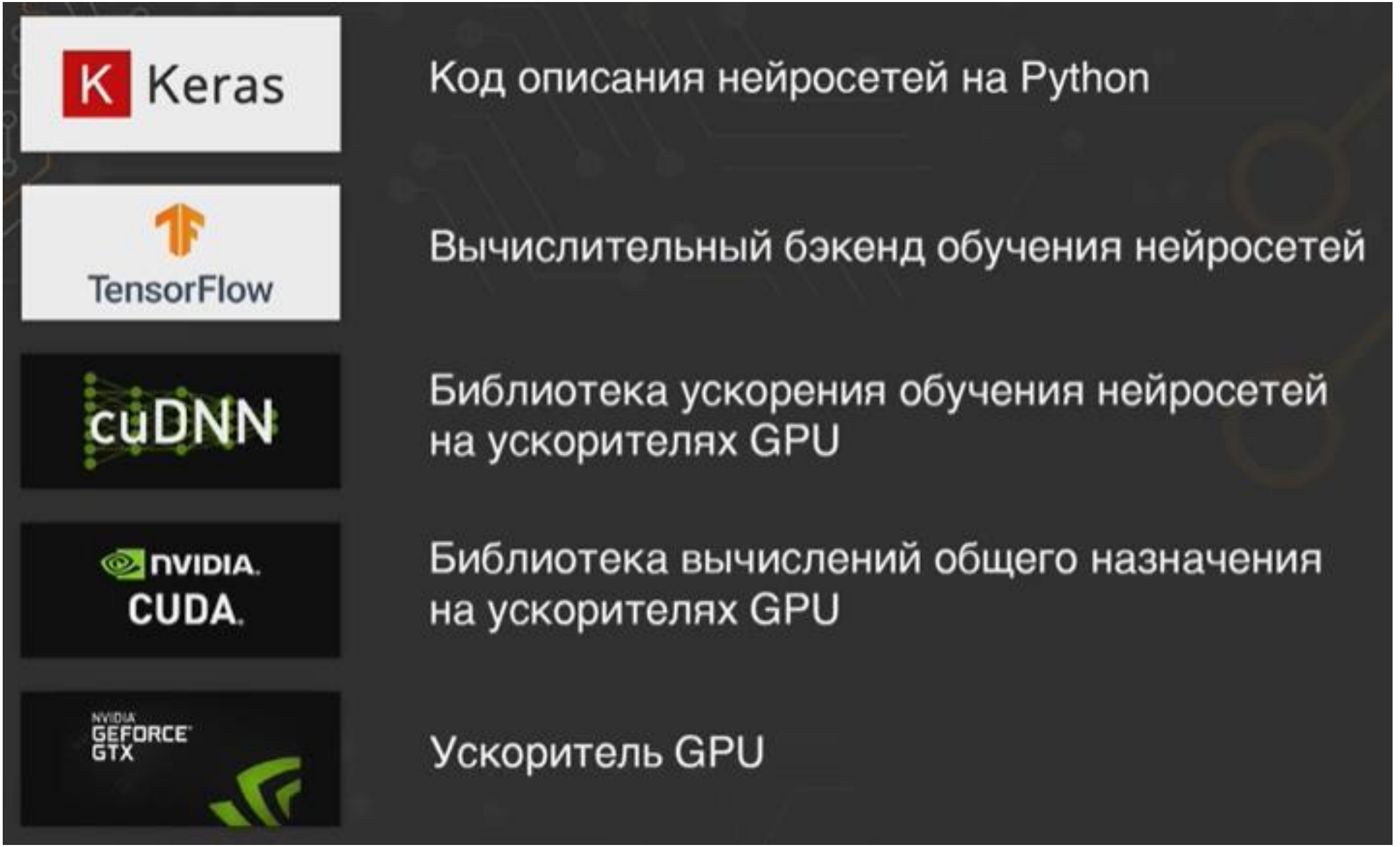
TensorFlow может автоматический проводить расчеты на GPU с помощью соответствующих библиотек. К сожалению, можно работать только с GPU производства компании NVIDIA, которые поддерживают технологию CUDA. Именно эта технология позволяет использовать GPU не только для графики, но и для проведения расчетов общего назначения.

Библиотеки для обучения ИНС

Библиотека NVIDIA **cuDNN** является расширением библиотеки **CUDA** и предназначена для ускорения обучения нейронных сетей на GPU. Приятная особенность заключается в том, что мы пишем программу на Python, а TensorFlow может запустить ее как на центральном процессоре, так и на GPU (если такой установлен в системе). Программу при этом менять не нужно.

Если нет GPU для обучения нейронных сетей, то можно использовать облачные платформы для машинного обучения **Colaboratory** и **Kaggle**, на которых уже установлены все необходимые библиотеки машинного обучения и есть мощные GPU. Эти платформы доступны бесплатно для изучения машинного обучения и нейронных сетей.

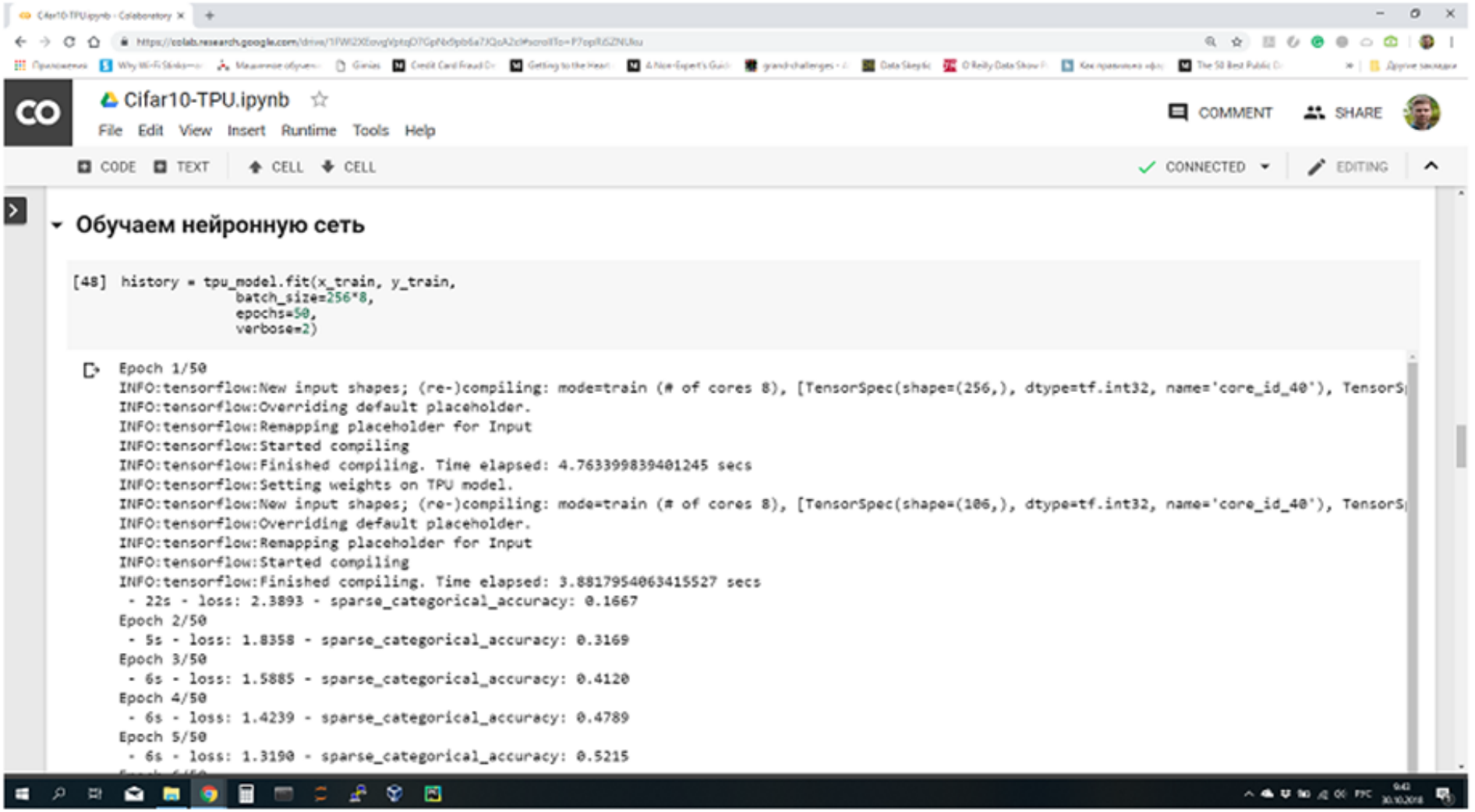
Библиотеки для обучения ИНС



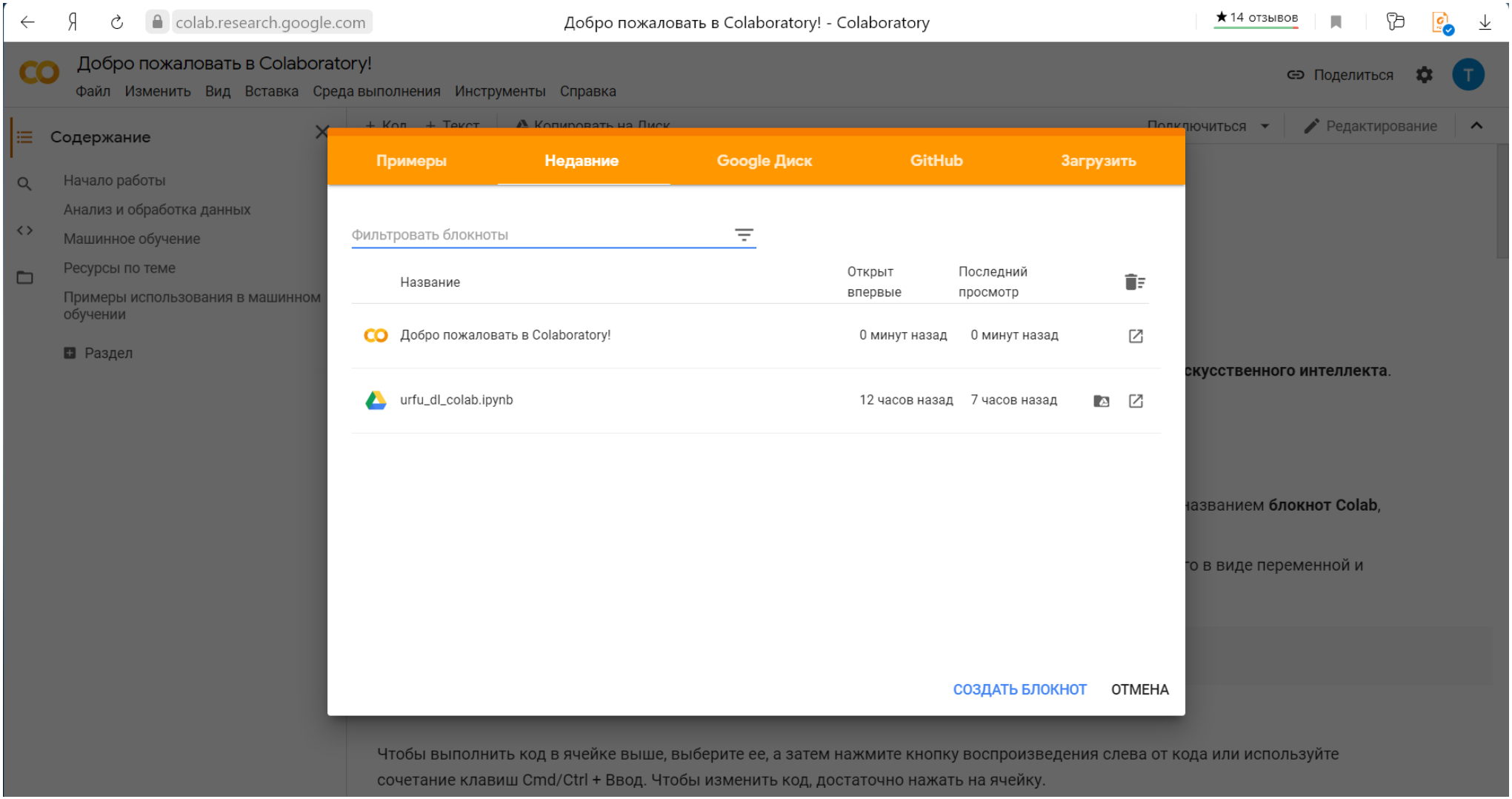
Платформа Colaboratory

на 12 часов

Colaboratory — это облачная платформа от Google для продвижения технологий машинного обучения. На ней можно получить бесплатно виртуальную машину с установленными популярными библиотеками TensorFlow, Keras, sklearn, pandas и т.п. Самое удобное, что на Colaboratory можно запускать ноутбуки, похожие на Jupyter. Ноутбуки сохраняются на Google Drive, можно их распространять и даже организовать совместную работу. Вот так выглядит ноутбук на Colaboratory:

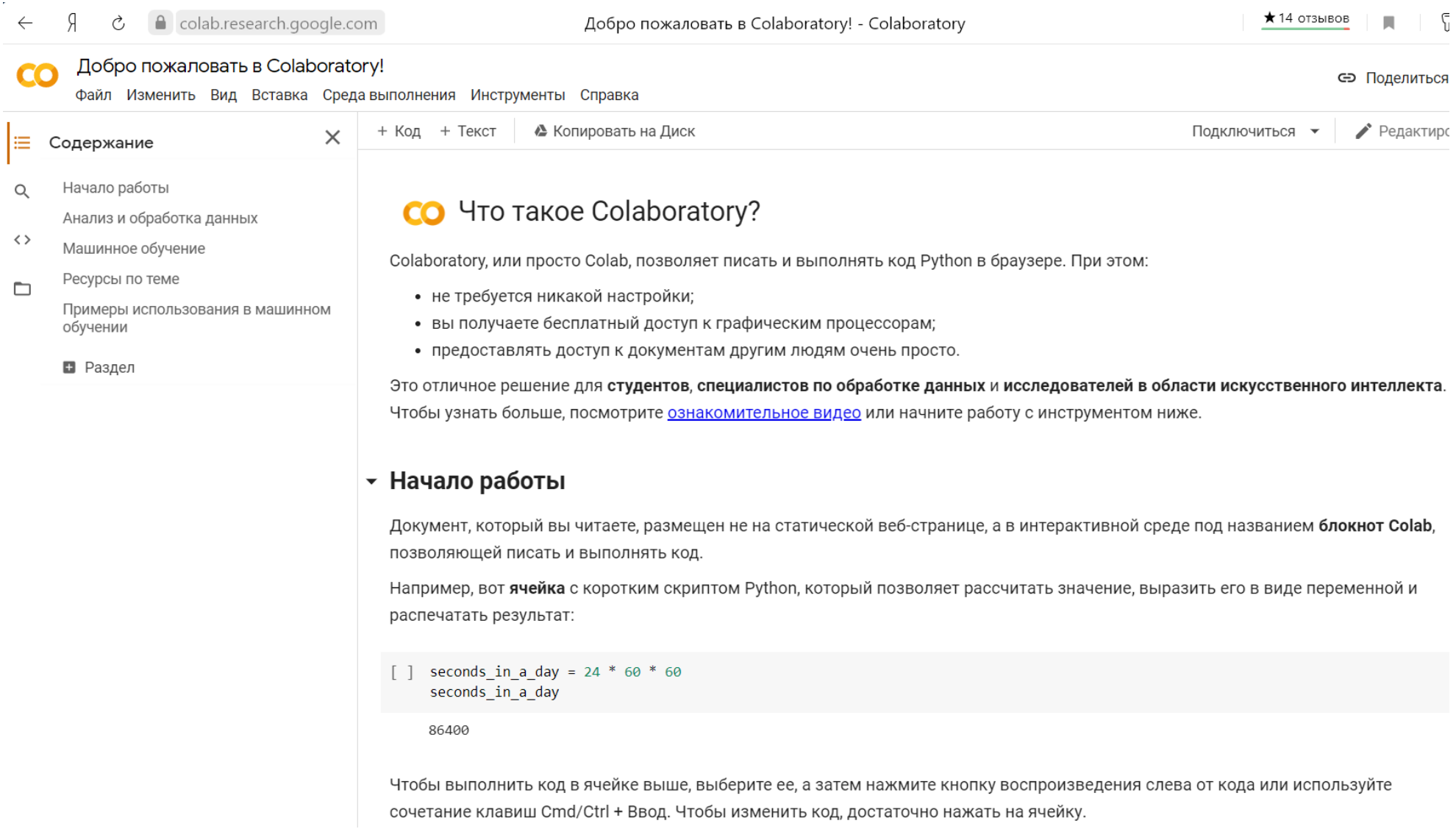


Облачная платформа Google Colaboratory для машинного обучения



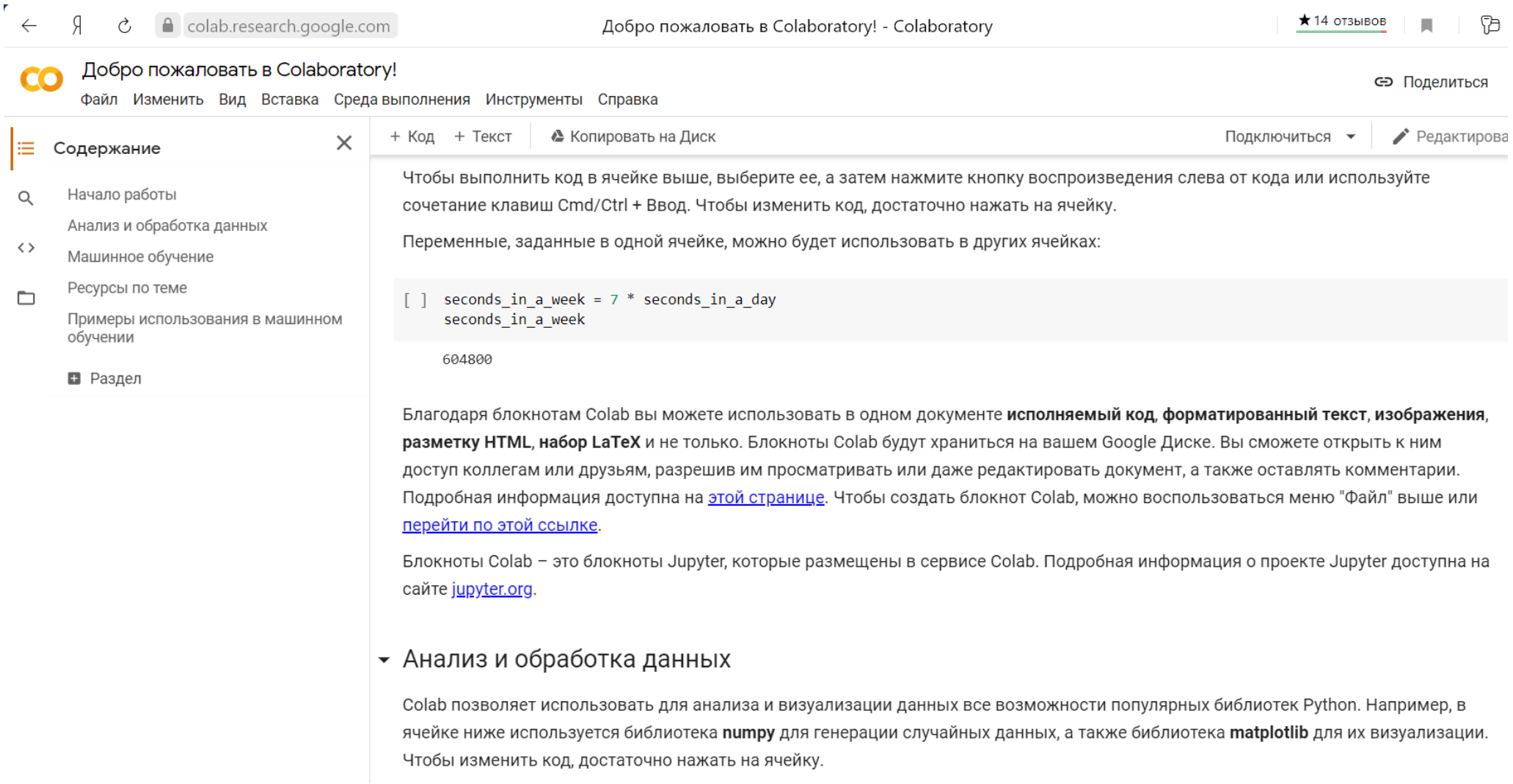
https://colab.research.google.com/notebooks/intro.ipynb

Облачная платформа Colaboratory для машинного обучения



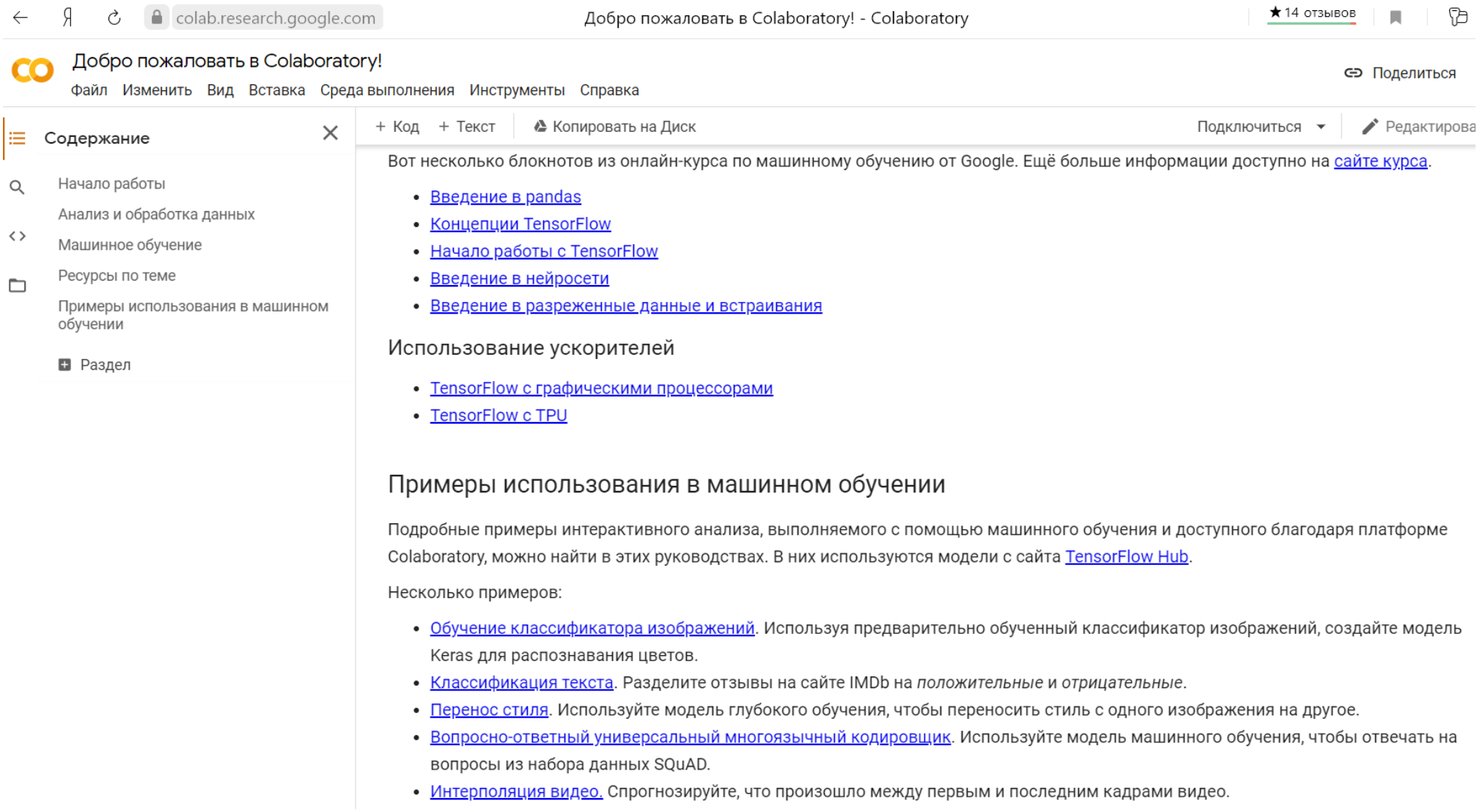
https://colab.research.google.com/notebooks/intro.ipynb

Облачная платформа Colaboratory для машинного обучения



https://colab.research.google.com/notebooks/intro.ipynb

Облачная платформа Colaboratory для машинного обучения



https://colab.research.google.com/notebooks/intro.ipynb

Облачная платформа Colaboratory для машинного обучения

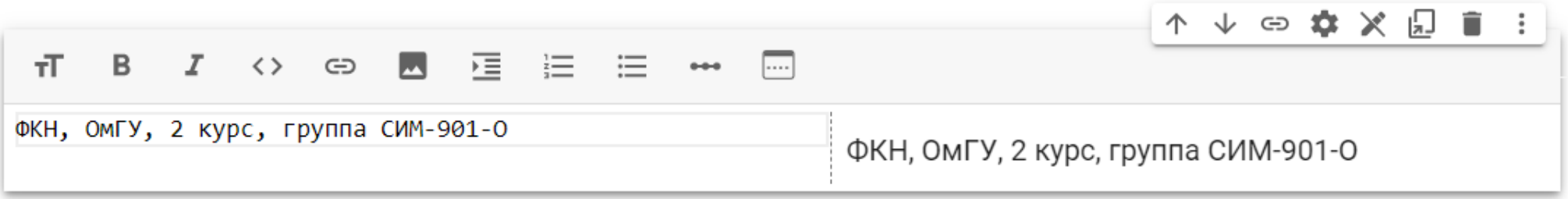
Бесплатная облачная платформа для машинного обучения от компании Google **сокращённо называется Colab**. На этой платформе можно создавать, так называемые, **ноутбуки**, которые содержат в себе код на Python, а также текстовое описание, которое говорит, что именно делает тот или иной фрагмент кода. В таком ноутбуке можно запускать код и получать результаты его выполнения.

Для того, чтобы создавать ноутбуки и запускать код, вам необходимо иметь **учетную запись Google**. Для того чтобы иметь возможность запускать код и редактировать, необходимо создать свой ноутбук или свою копию ноутбука. Для этого в меню "Файл" выберите пункт "Создать блокнот" или "Создать копию на Диске". Запустить ячейку можно поставив на нее курсор и нажав на клавиатуре Ctrl+Enter.

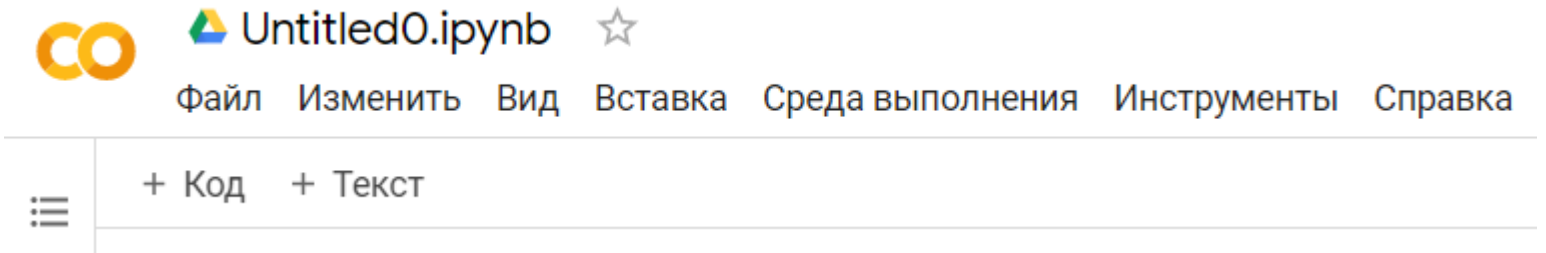
```
[ ]  # Это ячейка с кодом Python. Его можно запускать
     a = 1
     b = 2
     c = a + b
     print(c)

     3
```

Облачная платформа Colaboratory для машинного обучения

Следующая ячейка – это **ячейка с текстом**. Если два раза щелкнем на этой ячейке, то получим возможность изменять текст. В левой части мы видим, что происходит с текстом в режиме редактирования, в правом, что будет после того, как редактирование завершится.



Чтобы изменить содержимое ячейки, дважды нажмите на нее (или выберите "Ввод"). Для того чтобы создать новую ячейку с текстом или с кодом, нужно нажать на кнопки "+ Код" или "+ Текст".



Облачная платформа Colaboratory для машинного обучения

На платформе Colab уже установлено большое количество библиотек машинного обучения. Часто используют библиотеку от Google Tensorflow и библиотеку эффективных вычислений с массивами в Python, которая называется Numpy.

Для того чтобы **подключить** эти **библиотеки** в Python можно использовать **команду import**. Импортируем библиотеку **numpy**, и для вызова функции из этой библиотеки будем использовать сокращенное наименование **np**. Импортируем библиотеку **Tensorflow** и будем использовать сокращенное наименование **tf**. Библиотеки подключаются и не требуется их устанавливать.

Подключаем библиотеки `numpy` и `tensorflow`

```
[ ]  import numpy as np
     import tensorflow as tf
```

Облачная платформа Colaboratory для машинного обучения

В Colaboratory не обязательно использовать функцию print, если вы хотите узнать значение какой-то переменной, то можно просто написать эту переменную последнюю в ячейке.

Пример использования массивов `numpy`

```
[ ]  a = np.array([0, 1, 2, 3])
     b = np.array([4, 5, 6, 7])
     c = a + b
```

```
[ ]  a
```

```
array([0, 1, 2, 3])
```

```
[ ]  c
```

```
array([ 4,  6,  8, 10])
```

Облачная платформа Colaboratory для МО

Библиотеку Tensorflow использовать сложнее, чем библиотеку Numpy, поэтому просто проверим, что она подключилась и получим используемую версию.

Печатаем версию `tensorflow`

```
[ ]  tf.__version__

'2.3.0'
```

Код, который вы пишите на ноутбуке в браузере, выполняется на виртуальной машине в облаке Google. Поэтому, если нужно обработать какие-то файлы вашего компьютера, то вам сначала необходимо **загрузить их на диск виртуальной машины**. Для того чтобы это сделать, можно использовать **библиотеку Files** из Google Colab. Сначала подключаем эту библиотеку. Для того, чтобы загрузить данные с локального компьютера на диск виртуальной машины вызываем files.upload. Запустили ячейку, нажимаем кнопку "Выбрать файлы".

Загружаем файл с локального компьютера на диск виртуальной машины

```
[ ]  files.upload()
```

Выбрать файлы Файл не выбран Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving tshirt.jpg to tshirt.jpg
{'tshirt.jpg': b'\xff\xd8\xff\xe0\x00\x10JFIF\x00\x01\x01\x01\x00`\x00`\x00\x00\xff\xfe\x00;CREATOR: gd-jpeg v1.0 (using IJG JPEG v80), qu
```

Облачная платформа Colaboratory для МО

Виртуальная машина в облаке Google работает под управлением операционной системы **Linux**, поэтому можно использовать ее команды в Google Colab. Для этого в ячейке нужно писать **команды, начиная с восклицательного знака** (благодаря этому Colab поймет, что вы хотите использовать не команду Python, а команду Linux).

Для того чтобы узнать, какие файлы находятся на диске, используем команду Linux "ls", сокращение от list.

```
[ ]  !ls

     sample_data  tshirt.jpg
```

После того как вы обучили нейронную сеть, желательно **сохранить** ее на свой локальный компьютер, для того, чтобы можно было ее использовать в дальнейшем и не тратить время на повторное обучение. Для того чтобы это сделать, из библиотеки Files вызываем **команду download** и указываем путь к тому файлу, который вы хотите сохранить.

```
[ ]  files.download('sample_data/test.csv')
```

Облачная платформа Colaboratory для машинного обучения

Графический ускоритель GPU

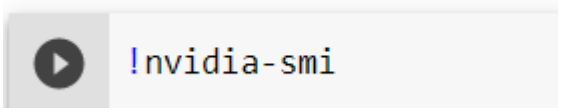
Платформа Colaboratory позволяет использовать GPU для ускорения обучения нейронной сети. Это бесплатно.

GPU можно **подключить** в меню "Среда выполнения" -> "Сменить среду выполнения" -> "Аппаратный ускоритель" -> GPU (Runtime -> Change runtime type -> Hardware Accelerator -> GPU).

После подключения GPU виртуальная машина перезапустится и все загруженные данные будут удалены. Поэтому рекомендуется подключать GPU в самом начале работы.

На платформе Colaboratory доступны несколько моделей GPU: NVIDIA Tesla K80, T4, P4 и P100. Модели отличаются производительностью и объемом памяти. Выбрать модель нельзя.

Узнать, какая модель подключена, можно следующей командой:



Облачная платформа Colaboratory для машинного обучения

Узнать, какая модель подключена, можно следующей командой:

```
!nvidia-smi
```

```
Tue Aug  4 16:48:23 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.57       Driver Version: 418.67       CUDA Version: 10.1     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            Off  | 00000000:00:04.0 Off |                    0 |
| N/A   57C    P8    10W /  70W |      0MiB / 15079MiB |      0%      Default |
|                               |                      |                 ERR! |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Облачная платформа Colaboratory для машинного обучения

Краткие рекомендации

1. Для работы с Colaboratory используйте учетную запись Google. Если у вас нет учетной записи, то создайте.
2. Для запуска кода в ноутбуке, создайте свою копию "Файл" -> "Сохранить копию на Диске" (File -> Save a copy in Drive...)
3. Не забудьте **подключить GPU** перед запуском кода. В противном случае нейросети будут обучаться очень медленно.
4. Ноутбуки – это обычные файлы в Google Drive. Ими можно делиться с другими людьми. Пункт меню "Поделиться" ("Share") в правом верхнем углу.
5. Пишите пояснения к своему коду в текстовых полях. Так вы сами через месяц не забудете, что именно делает код и почему.
6. Виртуальная машина в облаке Google работает **12 часов**. После этого машину можно перезапустить и продолжить работу. Но все **данные будут удалены**.

Работа с платформой Kaggle

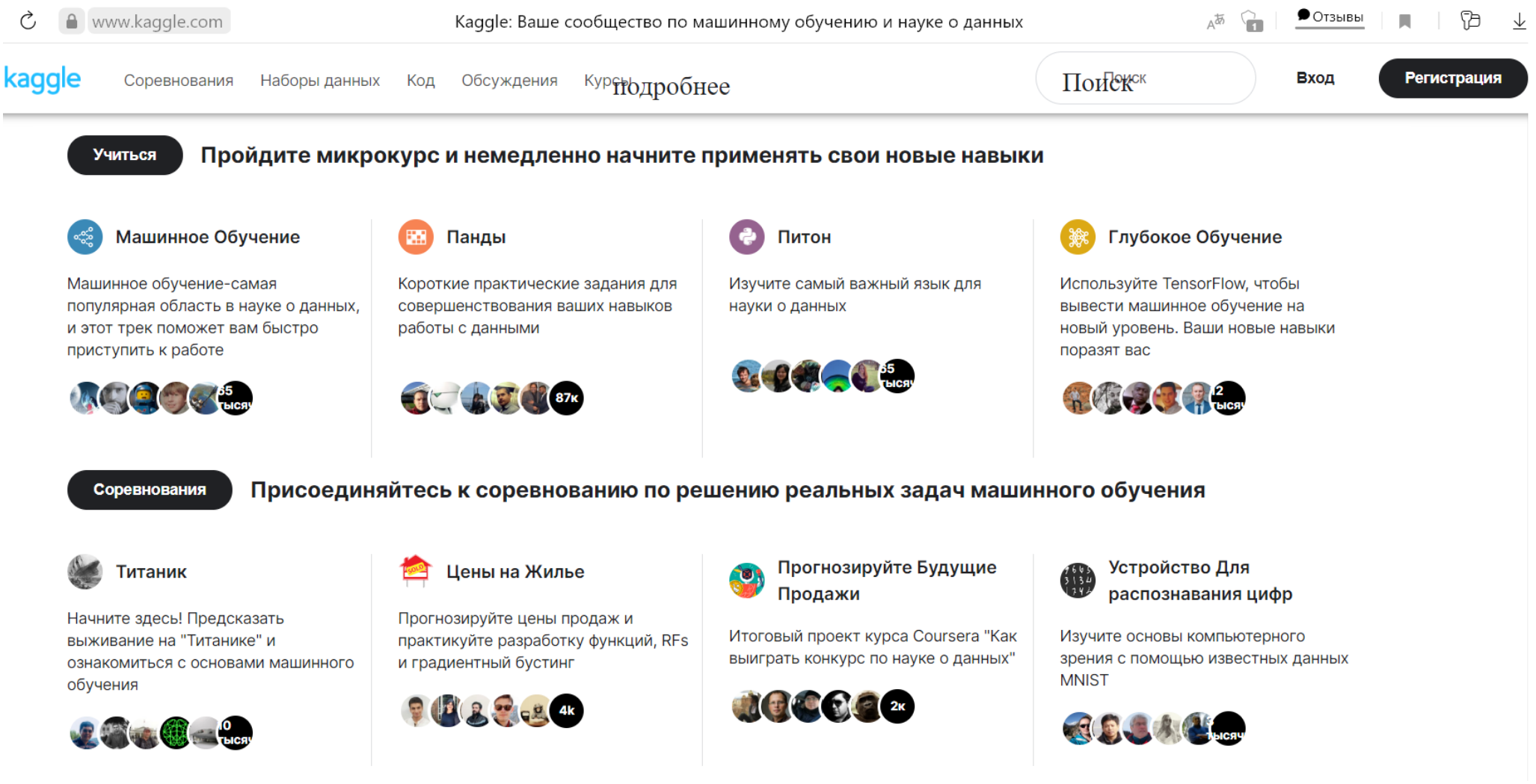
Kaggle – это онлайн-сообщество Data Scientist'ов и специалистов по машинному обучению (machine learning). Платформа Kaggle позволяет пользователям находить или публиковать датасеты, строить модели в специальной среде Kernel, работать с другими ML-специалистами и участвовать в соревнованиях в области Data Science.

Перед началом работы на платформе Kaggle необходимо зарегистрироваться, для этого перейдите по ссылке: https://www.kaggle.com/

Для осуществления регистрации можно войти, используя аккаунт Google, либо свою почту. После прохождения регистрации вы попадете на вашу личную стартовую страницу.

На платформе Kaggle, как и на Goolge Colaboratory, можно создавать блокноты с кодом на Python, использующим библиотеки обучения нейронных сетей. Они запускаются на виртуальной машине в облаке Google, для которой доступен бесплатный GPU.

Работа с платформой Kaggle



Работа с платформой Kaggle

УСЛУГИ И ВОЗМОЖНОСТИ KAGGLE

- ✔ Площадка для соревнований по Machine Learning. Со дня основания и по сей день основная цель Kaggle — дать возможность организациям устроить конкурс на лучшие модели и алгоритмы.
- ✔ Kaggle Kernels — облачная среда разработки по типу Jupyter Notebook и является аналогом Google Colab. Также каждому пользователю дается использовать GPU на 30 часов в неделю бесплатно. Своими разработками в Kernels можно делиться со всеми.
- ✔ Публикация, хранение и использование датасетов. Сообщество и организации, которые проводят соревнования, делятся наборами данных. Здесь можно найти текстовые данные, изображения, аудио и видео всевозможных сфер деятельности.
- ✔ Kaggle Learn — мини-курсы с использование Kernels для ознакомления с Data Science.
- ✔ Job's Board, в которой работодатели выкладывают списки вакансий.
- ✔ User ranking, который включает список лучших конкурсантов, датасетов, блокнотов, а также лучших пользователей форума. Система делит пользователей на следующие категории:

01. Grandmaster
02. Master
03. Expert
04. Contributors
05. Novices

СООБЩЕСТВО

В июле 2020 года, компания объявила о 5 миллионах зарегистрированных пользователях. Это самое большое сообщество по Data Science. Здесь можно найти как начинающих Data Scientis'ов, так и опытных профессионалов. Соревнования привлекают тысячи команд со всего мира.

В Kaggle содержит 50.000 датасетов и 400.000 блокнотов с исходным кодом. Многие исследователи после соревнований пишут научные статьи о своих результатах. Так, например, Джеффри Хинтон, пионер Deep Learning, со своими коллегами выиграли соревнование в сфере медицины и после этого опубликовали статью. Это было одна из первых побед глубокого обучения.

КАК ПРОИСХОДЯТ СОРЕВНОВАНИЯ

Kaggle предоставляет закрытые и открытые соревнования по машинному обучению. В закрытых могут принимать участие только команды по приглашению. Соревнование выглядит следующим образом:

01. Организатор соревнований готовит данные и детальное описание к проблеме, которую нужно решить. Kaggle также предоставляет консультационные услуги по организации.
02. Участники предоставляют свои решения, которые оцениваются через тесты организаторов.
03. Рейтинги работ публикуются в реальном времени.
04. По истечении установленного срока формируются списки победителей и выплачивается призовой фонд.

Организаторы соревнований имеют интеллектуальные права на разработанный победителями алгоритм, модель или ПО.

Модели могут разрабатываться с использованием любых вычислительных ресурсов, поэтому команды с хорошим оборудованием и располагающие облачными ресурсами с GPU будут иметь преимущество.

Участие в соревнованиях может дать конкурсанту практический опыт в разработке моделей Machine Learning. Призовые места обеспечат не только денежным призом, но и всемирной известностью в сообществе Data Science.