УДК 004.85 DOI: 10.14529/cmse170303

ОБЗОР МЕТОДОВ ОБУЧЕНИЯ ГЛУБОКИХ НЕЙРОННЫХ СЕТЕЙ

© 2017 А.В. Созыкин

Институт математики и механики им. Н.Н.Красовского УрО РАН
(620990 Екатеринбург, ул.Софьи Ковалевской, д. 16),
Уральский федеральный университет
(620002 Екатеринбург, ул. Мира, д. 19)
E-mail: avs@imm.uran.ru

Поступила в редакцию: 12.04.2017

Глубокие нейронные сети в настоящее время становятся одним из самых популярных подходов к созданию систем искусственного интеллекта, таких как распознавание речи, обработка естественного языка, компьютерное зрение и т.п. В статье представлен обзор истории развития и современного состояния методов обучению глубоких нейронных сетей. Рассматривается модель искусственной нейронной сети, алгоритмы обучения нейронных сетей, в том числе алгоритм обратного распространения ошибки, применяемый для обучения глубоких нейронных сетей. Описывается развитие архитектур нейронных сетей: неокогнитрон, автокодировщики, сверточные нейронные сети, ограниченная машина Больцмана, глубокие сети доверия, сети долго-краткосрочной памяти, управляемые рекуррентные нейронные сети и сети остаточного обучения. Глубокие нейронные сети с большим количеством скрытых слоев трудно обучать из-за проблемы исчезающего градиента. В статье рассматриваются методы решения этой проблемы, которые позволяют успешно обучать глубокие нейронные сети с более чем ста слоями. Приводится обзор популярных библиотек глубокого обучения нейронных сетей, которые сделали возможным широкое практическое применение данной технологии. В настоящее время для задач компьютерного зрения используются сверточные нейронные сети, а для обработки последовательностей, в том числе естественного языка, — рекуррентные нейронные сети, прежде всего сети долго-краткосрочной памяти и управляемые рекуррентные нейронные сети.

Ключевые слова: глубокое обучение, нейронные сети, машинное обучение.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Созыкин А.В. Обзор методов обучения глубоких нейронных сетей // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2017. Т. 6, № 3. С. 28–59. DOI: 10.14529/cmse170303.

Введение

Глубокие нейронные сети в настоящее время становятся одним из самых популярных методов машинного обучения. Они показывают лучшие результаты по сравнению с альтернативными методами в таких областях, как распознавание речи, обработка естественного языка, компьютерное зрение [1], медицинская информатика [2] и др. Одна из причин успешного применения глубоких нейронных сетей заключается в том, что сеть автоматически выделяет из данных важные признаки, необходимые для решения задачи. В альтернативных алгоритмах машинного обучения признаки должны выделяться людьми, существует специализированное направление исследований — *инженерия признаков* (feature engineering). Однако при обработке больших объемов данных нейронная сеть справляется с выделением признаков гораздо лучше, чем человек.

В статье представлен исторический обзор развития архитектур глубоких нейронных сетей и подходов к их обучению. Задача составления такого обзора существенно

затруднена тем, что вариантов глубоких нейронных сетей было предложено очень много и терминология менялась со временем [3]. Модель искусственных нейронных сетей была предложена в 1943 году [4], а сам термин *глубокое обучение* (deep learning) стал широко использоваться только начиная с 2006 года [5, 6]. До этого применялись термины *загрузка глубоких сетей* (loading deep networks) [7, 8] и *обучение глубокой памяти* (learning deep memories) [9].

Рост популярности глубоких нейронных сетей, происходящий в последние несколько лет, можно объяснить тремя факторами. Во-первых, произошло существенное увеличение производительности компьютеров, в том числе ускорителей вычислений GPU (Graphics Processing Unit), что позволило обучать глубокие нейронные сети сети значительно быстрее и с более высокой точностью [10]. Ранее имеющихся вычислительных мощностей не хватало для обучения сколько-нибудь сложной сети, пригодной для решения практических задач. Во-вторых, был накоплен большой объем данных, который необходим для обучения глубоких нейронных сетей. В-третьих, разработаны методы обучения нейронных сетей, позволяющие быстро и качественно обучать сети, состоящие из ста и более слоев [11], что раньше было невозможно из-за проблемы исчезающего градиента и переобучения. Сочетание трех факторов привело к существенному прогрессу в обучении глубоких нейронных сетей и их практическом использовании, что позволило глубоким нейронным сетям занять лидирующую позицию среди методов машинного обучения.

Статья организована следующим образом. В разделе 1 рассмотрена модель искусственных нейронных сетей. Раздел 2 посвящен методам обучения нейронных сетей. В разделе 3 приведен исторический обзор развития архитектур глубоких нейронных сетей. В разделе 4 рассматриваются популярные программные системы для обучения глубоких нейронных сетей. В заключении подводятся итоги и описываются перспективы развития архитектур и методов обучения глубоких нейронных сетей.

1. Искусственные нейронные сети

Модель искусственного нейрона была предложена Уорреном МакКаллоком (Warren McCulloch) и Уолтером Питтсом (Walter Pitts) в 1943 году в работе [4]. В качестве основы для своей модели авторы использовали биологический нейрон. Искусственный нейрон МакКаллока—Питтса имеет N входных бинарных величин $x_1, ..., x_n$, которые трактуются как импульсы, поступающие на *вход нейрону* (рис. 1). В нейроне импульсы складываются с *весами* $w_1, ..., w_n$.

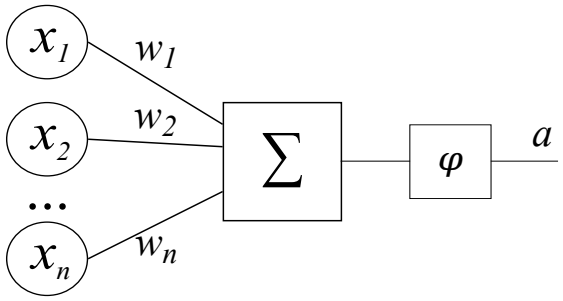

Рис. 1. Модель искусственного нейрона МакКаллока—Питтса

Выходной сигнал нейрона определяется по формуле:

$$a = \varphi(\sum_{i=1}^{N} w_i x_i), \tag{1}$$

где нелинейная функция φ (*функция активации*) преобразует суммарный импульс в выходное значение нейрона. В модели МакКаллока—Питтса для этой цели использовалась функция Хевисайда. В дальнейшем было предложено использовать другие типы функций активации: логистическую сигмоидальную ($f(x) = \frac{1}{1+e^{(-x)}}$) [12], гиперболический тангенс ($tanh(x) = \frac{2}{1+e^{(-2x)}} - 1$) [13] и радиально-базисную функцию [14]. Такие функции активации обеспечивали более плавное изменение выходного сигнала нейрона.

МакКаллок и Питтс предложили также метод объединения отдельных нейронов в *искусственные нейронные сети*. Для этого выходные сигналы нейрона передаются на вход следующему нейрону (рис. 2). Нейронная сеть состоит из нескольких *слоев*, на каждом из которых может находиться несколько нейронов. Слой, который принимает сигналы из внешнего мира, называется *входным*. Слой, который выдает сигналы во внешний мир, — *выходным*. Остальные слои называются *скрытыми*.

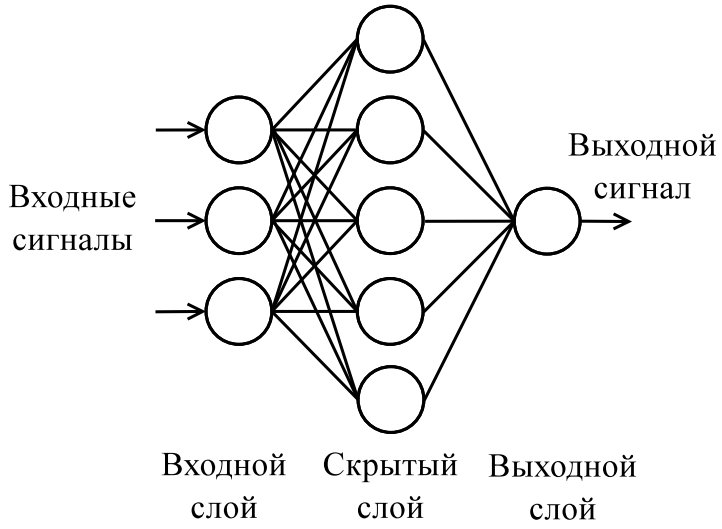

Рис. 2. Искусственная нейронная сеть

Искусственные нейронные сети делятся на *сети прямого распространения сигнала* (feedforward networks), в которых нет циклов, и *рекуррентные сети* (recurrent networks), в которых циклы разрешены.

Однозначного определения, что такое *глубокая нейронная сеть*, не существует. В данной статье под глубокой нейронной сетью будет пониматься такая нейронная сеть, которая содержит более одного скрытого слоя.

Искусственные нейронные сети, заданные таким образом, способны приблизить любую непрерывную функцию с любой требуемой точностью [15–18]. Однако в настоящее время не существует конструктивного подхода, который бы позволял гарантированно создавать нейронные сети с заранее заданными свойствами. Это является существенным недостатком, ограничивающим применение нейронных сетей.

2. Обучение нейронных сетей

Обучение нейронной сети — это процесс определения весов соединений между нейронами таким образом, чтобы сеть приближала необходимую функцию с заданной точностью. Существует три подхода к обучению нейронных сетей [3]: *обучение с учителем* (supervised learning), *обучение без учителя* (unsupervised learning) и *обучение с подкреплением* (reinforcement learning). При обучении с учителем на вход сети подаются наборы входных сигналов (*объектов*), для которых заранее известен правильный ответ (*обучающее множество*). Веса меняются по определенным правилам в зависимости от того, правильный ли выходной сигнал выдала сеть. При обучении без учителя на вход сети подаются объекты, для которых правильный выходной сигнал заранее не известен. Обучение с подкреплением предполагает наличие внешней среды, с которой взаимодействует сеть. Обучение происходит на основании сигналов, полученных от этой среды.

Нейронные сети МакКаллока—Питтса не обучались. Веса для всех входов нейронов должны были быть заданы заранее.

Впервые идею обучения нейронных сетей предложил Дональд Хэбб (Donald Hebb) в 1949 году [19]. Согласно Хэббу, связи нейронов, которые активируются вместе, должны усиливаться, а связи нейронов, которые срабатывают отдельно друг от друга, должны ослабевать. Хэбб предложил правила изменения веса входных сигналов нейронов в соответствии с тем, правильный ответ выдавала сеть, или нет [19] (обучение с учителем). А.В. Новиков доказал сходимость предложенного метода обучения нейрона на основе правил Хэбба [20], при условии, что выборка объектов линейно разделима. Впоследствии было предложено несколько аналогичных правил как для обучением с учителем [21–23], так и без учителя [24–28].

В 1970 году А.Г. Ивахненко разработал *метод группового учета аргументов* (group method of data handling) [29, 30], позволяющий не только вычислять веса связей между нейронами, но и определять количество слоев в сети и нейронов в них в зависимости от потребностей прикладной задачи. Используя подход обучения с учителем, уровни сети инкрементально строятся и обучаются на основе обучающего множества с использованием регрессионного анализа. Затем происходит этап упрощения сети с применением отдельного множества объектов с известными правильными ответами, которое не использовалось при обучении (*проверочное множество*, validation set). Для исключения ненужных нейронов из сети используется регуляризация. В работе [30] описано применение метода группового учета аргументов для обучения глубокой нейронной сети, состоящей из восьми слоев. Метод широко использовался на практике [31–33].

В настоящее время для обучения нейронных сетей, в том числе глубоких, используется алгоритм *обратного распространения ошибки* (error backpropagation algorithm), основанный на методе градиентного спуска. Алгоритм был предложен в 1970 году в магистерской диссертации [34, 35] без связи с нейронными сетями. Первое применение этого алгоритма для обучения нейронных сетей описано в работе [36], вышедшей в 1981 году. После этого появилось еще несколько работ на эту тему [37–39]. Алгоритм обратного распространения ошибки использует обучение с учителем, для него требуется обучающее множество с заранее известными правильными ответами. Вводится мера ошибки, которая определяет, насколько сильно выходные значения сети отличаются от правильных ответов. Затем мера ошибки минимизируется с помощью метода градиентного спуска путем

изменения значений весов в сети. Для того чтобы оценить, насколько сильно каждый вес влияет на выходное значение, рассчитываются частные производные ошибки по весам. Затем производится изменение весов на небольшие значение с учетом градиента. Так повторяется до тех пор, пока ошибка на выходе не сократится до допустимых значений. Начальные значения весов нейронов в сети задаются случайным образом.

В глубокой нейронной сети с несколькими скрытыми слоями производится расчет ошибки, которая передается от одного слоя к другому. На первом этапе рассчитывается значение ошибки на выходе нейронной сети, для которого мы знаем правильные ответы. Затем рассчитывается ошибка на входе в выходной слой сети, которая будет использоваться как ошибка на выходе скрытого слоя. Таким способом расчет продолжается до того момента, когда будет известна ошибка на входном слое. Именно поэтому алгоритм имеет название обратное распространение ошибки.

Возможно несколько вариантов реализации обучения нейронных сетей с помощью алгоритма обратного распространения ошибки. При полном обучении градиент рассчитывается для всех объектов обучающего множества. Однако такой подход часто не является эффективным в случае, когда обучающее множество большое и для обработки всех его элементов требуется значительное время. Альтернативный вариант — использование метода стохастического градиентного спуска, при котором веса изменяются при обработке одного элемента обучающего множества (онлайн-обучение) или нескольких элементов (обучение на пакетах или мини-выборках). На практике для обучения нейронных сетей чаще всего используется именно метод стохастического градиентного спуска или его модификации [40–42].

3. Архитектура глубоких нейронных сетей

Нейронная сеть, показанная на рис. 2, называется *полносвязной*. В такой сети каждый нейрон следующего слоя связан со всеми нейронами предыдущего слоя. Однако это не единственный вариант соединения нейронов в сеть. В данном разделе рассматривается развитие архитектур нейронных сетей.

В 1980 году Кунихико Фукушима (Kunihiko Fukushima) предложил архитектуру нейронной сети, которая называется *неокогнитрон* [43]. Архитектура использовала аналогию со сложными и простыми клетками в зрительной коре кошки [44]. Простые клетки срабатывают в ответ на простые визуальные сигналы, такие как ориентация границ. Сложные клетки менее зависимы от пространственного расположения сигналов и ориентируются на более общие признаки. В неокогнитроне простым клеткам соответствуют *сверточные слои* (convolutional layers), а сложным — *слои подвыборки* (subsampling layers). В сверточных слоях окно *сверточного узла* (convolutional unit) с заданным набором весов (*ядро свертки*) перемещается по двумерному массиву входных данных, например, по пикселям изображения (рис. 3). Значения совпадающих элементов в данных и ядре свертки перемножаются, полученные результаты складываются и поступают в нейрон следующего слоя. Все сверточные узлы используют одинаковые ядра свертки, поэтому для описания сверточной сети требуется относительно немного параметров. Как правило, в сверточных слоях используется не одно, а несколько ядер свертки.

Выходы сверточных слоев в неокогнитроне подключаются ко входам слоев подвыборки. Причем к одному нейрону в слое подвыборки подключаются несколько нейронов сверточного слоя, как правило из квадратной области размером 2×2 или больше. Нейроны

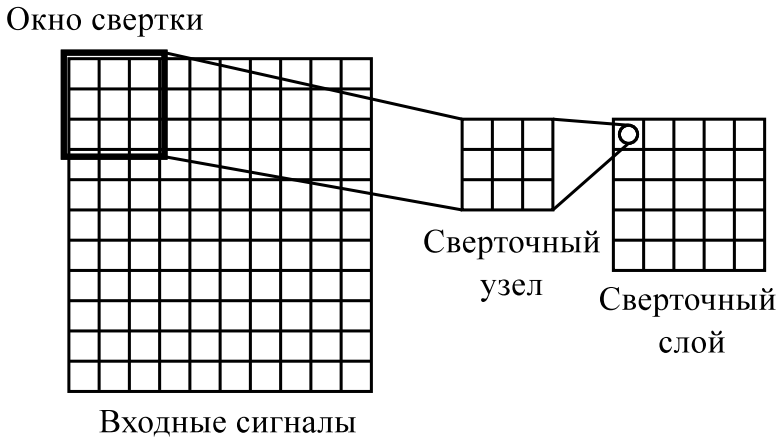

Рис. 3. Схема сверточного слоя нейронной сети неокогнитрон

в слое подвыборки срабатывают в случае активности хотя бы один из входных сигналов. На этом слое важно наличие самого сигнала, а не его конкретные координаты, поэтому слои подвыборки менее чувствительны к незначительным сдвигам и изменениям масштаба изображения. Обучение сверточных слоев в неокогнитроне производится с помощью локальных алгоритмов обучения без учителя, либо веса задаются заранее [45, 46]. Для слоев подвыборки используется *пространственное усреднение* (spatial averaging) [43, 47]. Таким образом, несмотря на то, что неокогнитрон является глубокой нейронной сетью, глубокое обучение в нем не используется.

В 1987 году Дана Баллард (Dana Ballard) предложил подход к обучению нейронных сетей без учителя на основе *автокодировщика* (autoencoder) [48]. Простой автокодировщик содержит всего один скрытый слой (рис. 4) с кодом h, который служит для представления входного сигнала x. Автокодировщик содержит функцию кодирования f, которая используется для преобразования входного сигнала в код $h = f(x)$ и функцию декодирования g, которая по коду восстанавливает значения входных сигналов $r = g(x)$.

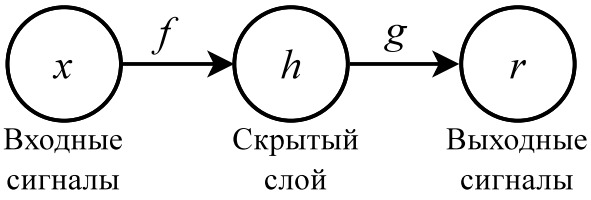

Рис. 4. Схема простого автокодировщика

Автокодировщики применяются для уменьшения размерности обрабатываемых данных. Для этого используются линейные методы, такие как метод главных компонент. За счет снижения размерности автокодировщик выделяет наиболее значимые характеристики данных.

Для обучения автокодировщиков используется метод *рециркуляции* [49]. Автокодировщик обучается таким образом, чтобы на его выходе были те же самые сигналы, как и на входе. После обучения скрытый слой простого автокодировщика вставляется в автокодировщик более высокого уровня, который содержит больше скрытых слоев. Таким образом строится иерархия автокодировщиков в виде стека (рис. 5). При этом размерность данных уменьшается на каждом уровне иерархии. Подобная иерархия может

быть использована не только для автокодировщиков, но и для других методов обучения без учителя [50, 51].

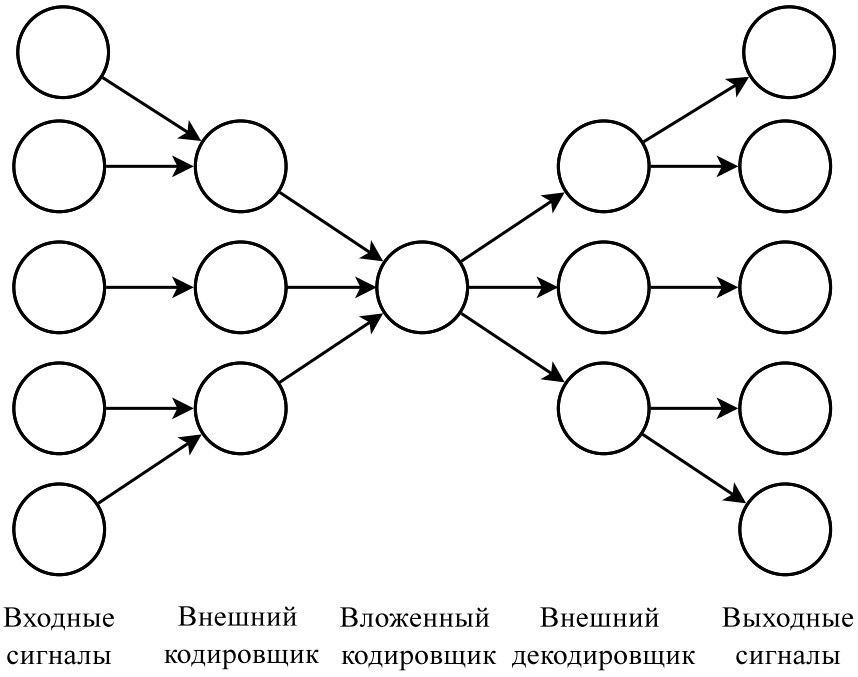

Рис. 5. Иерархия автокодировщиков

Ян Лекун (Yann LeCun) в 1989 году применил алгоритм обратного распространения ошибки для обучения сети с архитектурой, очень похожей на неокогнитрон [52]. Сеть содержала сверточные слои с одинаковыми весами (сверточными ядрами) и слои подвыборки. В этой же работе был представлен набор данных MNIST, содержащий рукописные цифры, распознавание которых со временем стало очень популярным тестом в машинном обучении. Глубокие сверточные сети, обученные алгоритмом обратного распространения ошибки, показали хорошие результаты на распознавании рукописных цифр [53] и отпечатков пальцев [54].

В конце 80-х годов XX века стало понятно, что одного алгоритма обратного распространения ошибки недостаточно для эффективного глубокого обучения. Несмотря на некоторые успешные примеры [53, 54], использование более одного скрытого слоя редко давало преимущества на практике [55–57]. Причина этого была сформулирована в 1991 году в работе [58, 59] — *проблема исчезающего градиента* (vanishing gradient problem). При использовании традиционных функций активации (см. раздел 1), сигналы об обратно распространяемых ошибках быстро становятся очень маленькими (или наоборот, чрезмерно большими). В практических задачах они уменьшаются экспоненциально с количеством слоев в сети. Эта проблема также известна как *проблема длительной задержки* (long time lag problem) [60, 61].

Один из подходов к решению проблемы исчезающего градиента заключается в полном отказе от использования градиента для обучения. Для некоторых задач хороших результатов можно добиться назначением весов случайным образом [62]. Подход к обучению *Еволино* (Evolino) [63] использует линейные методы для определения оптимальных весов для выходного слоя и *эволюцию* для определения весов скрытых слоев. Также возможно применение методов *универсального поиска* [64, 65].

Альтернативный подход к решению проблемы исчезающего градиента — использование безгессианной оптимизации (Hessian-free optimization) [66–70].

Очень глубокий обучатель, предложенный в работе 1992 года [71], обеспечивает решение проблемы исчезающего градиента и обучение сети глубиной до сотен слоев за счет использования предварительного обучения без учителя иерархии рекуррентных нейронных сетей. Каждая рекуррентная сеть обучается отдельно для того, чтобы предсказать следующее значение, которое поступит ей на вход [72, 73]. После обучения только ошибочно предсказанные значения передаются на более высокий уровень сети. Эта сеть работает уже на более медленной временной шкале, за счет чего информация сжимается и каждой последовательности сигналов соответствует набор все менее и менее избыточного кодирования на все более глубоких уровнях сети. Другое название такой архитектуры нейронной сети — *компрессор истории* (History Compressor), она может сжимать данные как в пространстве, так и во времени. Существует также и непрерывный вариант компрессора истории [74].

Проблему исчезающего градиента позволяет решить другая архитектура рекуррентной нейронной сети — *сеть долго–краткосрочной памяти* (Long Short-Term Memory) [75–77]. Такие сети содержат узлы специального типа, которые позволяют запоминать значения на длительный срок. Блок сети долго-краткосрочной памяти содержит специальный нейрон, используемый в качестве ячейки памяти (рис. 6). Выход нейрона соединен с его собственным входом с единичным весом. За счет этого значение в нейроне на каждом этапе перезаписывается и таким образом сохраняется. Управление нейроном выполняется с помощью трех вентилей: входного, выходного и вентиля забвения. При открытом входном вентиле значение на входе записывается в ячейку памяти. Если входной вентиль закрыт, то входные сигналы никак не влияют на содержимое ячейки. Открытый выходной вентиль позволяет прочитать значение из ячейки. Когда значение больше не нужно, его можно стереть с помощью вентиля забвения. Вентили подключаются к другим узлам нейронной сети, которые в процессе обучения определяют, когда необходимо открыть или закрыть тот или иной вентиль.

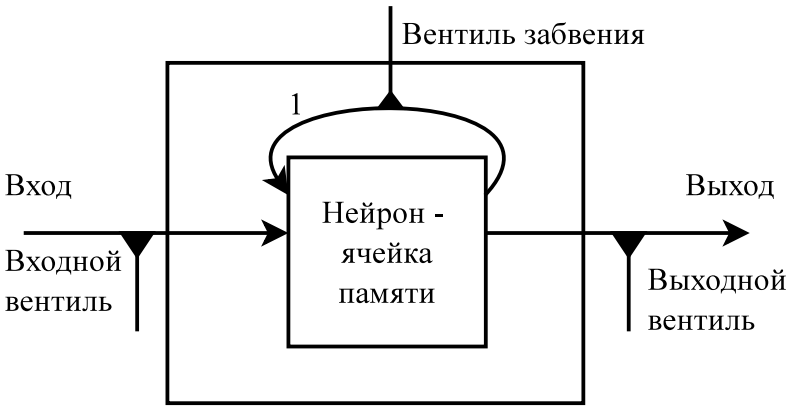

Рис. 6. Схема блока сети долго-краткосрочной памяти

Благодаря таким ячейкам сети долго-краткосрочной памяти могут определить важность событий, произошедших тысячи дискретных временных шагов назад, и запомнить эти события. Рекуррентные сети, которые использовались до этого, могли помнить о событии не дольше десяти временных шагов. Проблема исчезающего градиента в сетях долго-краткосрочной памяти решается за счет использования функции тождества

в качестве активационной и обратной связи с собственным входом с весом равным единице (рис. 6). Так как производная функции тождества равна единице, то ошибка при передаче через такие узлы не может исчезнуть.

В 1992 году появилась архитектура нейронных сетей *кресцептрон* (cresceptron) [78], основой для которой послужил неокогнитрон. В отличии от неокогнитрона, кресцептрон изменяет свою топологию во время обучения, по аналогии с сетями, использующими метод группового учета аргументов [29]. Важная идея, предложенная в кресцептроне — использование слоев *выбора максимального элемента* (max-pooling) вместо слоев подвыборки с усреднением. Более поздние и усовершенствованные версии Кресцептрона содержали также слои *размывания* (blurring) для уменьшения зависимости от положения объектов [79]. Слои максимального выбора сейчас широко применяются в сверточных нейронных сетях. Однако для обучения современных сверточных сетей используется алгоритм обратного распространения ошибки [80], что является более эффективным [81].

В 2006 году Джеффри Хинтон (Geoffrey Hinton) и Руслан Салахутдинов (Ruslan Salakhutdinov) предложили новую архитектуру нейронных сетей — *глубокие сети доверия* (Deep Belief Networks) [5]. В этой архитектуре для решения проблемы исчезающего градиента использовалась комбинация обучения с учителем и без учителя. Глубокая сеть доверия представляет собой стек *ограниченных машин Больцмана* [82, 83]. Каждая такая машина содержит только два слоя нейронов: входной и скрытый (рис. 7). Соединения есть только между нейронами разных слоев, в отличие от машины Больцмана высокого порядка [84], которая может содержать другие типы связей.

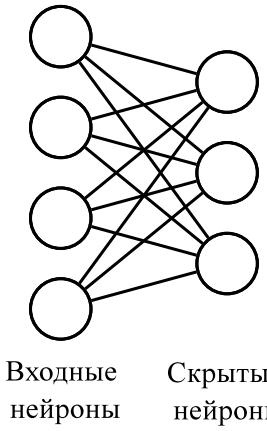

Рис. 7. Ограниченная машина Больцмана

Каждая ограниченная машина Больцмана получает сигналы о представлении шаблонов с предыдущего уровня и пытается закодировать их с использованием обучения без учителя [6] (рис. 7). После этого производится тонкая настройка всей сети с помощью обучения с учителем с использованием алгоритма обратного распространения ошибки. Обученная таким образом глубокая сеть доверия показала хорошие результаты на тесте распознавания рукописных цифр MNIST [5], распознавании фонем [85], поиске документов [86] и других задачах. Похожим альтернативным подходом, основанным на предварительном обучении без учителя и последующей точной настройке путем обучения с учителем, является использование стека автокодировщиков [48, 87–90].

В 2006–2007 годах также произошло развитие глубокого обучения сверточных сетей на основе обучения с учителем без предварительного обучения без учителя. В работе [80] впервые описано применение алгоритма обратного распространения ошибки для обучения

глубокой нейронной сети с архитектурой, подобной неокогнитрону и кресцептрону, состоящей из чередующихся слоев свертки и максимального выбора. Такая архитектура нейронных сетей активно используется по настоящее время [91–95].

Существенный вклад в развитие сверточных нейронных сетей внесло предложение использовать *полулинейную функцию активации* (rectified linear unit) [96, 97], которая задается следующим образом: $ReLU(x) = max(0, x)$. Такая функция активации позволяет избавиться от проблемы исчезающего градиента, т.к. при положительном значении сигнала не происходит его изменения, в отличие от сигмоидальных функций активации. Кроме того, полулинейная функция активации позволяет сократить время обучения нейронной сети [98]. Нейроны, выходной сигнал которых отрицательный, не участвуют в расчетах, а для остальных требуется выполнять только линейные вычисления.

В 2010 году Ксавье Глоро (Xavier Glorot) и Иошуа Бенджио (Yoshua Bengio) в работе [99] провели исследование влияния методов начальной инициализации весов и функций активации на распространение сигнала в сети как в прямом, так и в обратном направлении. Согласно исследованию, использование логистической сигмоидальной функции совместно с начальной инициализацией весов плохо подходит для создания глубоких нейронных сетей, т.к. приводит к быстрому насыщению. Функция активации гиперболический тангенс такой проблемой не обладает из-за симметричности (среднее значение функции 0, область значения — $(-1, 1)$). Глоро и Бенджио предложили новый метод инициализации весов нейронной сети, который они назвали *нормализованная инициализация*. Начальные значения весов сети W определяются по следующей формуле:

$$W \sim U[-\frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}, \frac{\sqrt{6}}{\sqrt{n_j + n_{j+1}}}], \tag{2}$$

где U — равномерное распределение на отрезке, n_j — количество нейронов на текущем слое сети, n_{j+1} — количество нейронов на следующем слое сети. Использование нормализованной инициализации приводит к снижению насыщения нейронов и сигнал об ошибке распространяется значительно лучше.

Метод нормализованной инициализации весов в 2015 году был адаптирован для полулинейной функции активации. В работе [100] предлагается определять начальные веса W следующим образом:

$$W \sim U[-\frac{2}{n_j}, \frac{2}{n_j}], \tag{3}$$

где U — равномерное распределение на отрезке, n_j — количество нейронов на текущем слое сети. Метод нормализованной инициализации весов позволил достигать качественного обучения глубоких нейронных сетей без необходимости использовать предварительное обучение без учителя.

Сергей Йоффе (Sergey Ioffe) и Кристиан Жегеды (Christian Szegedy) в 2015 году [101] предложили использовать в нейронных сетях специальные слои пакетной нормализации (batch normalization), которые позволяют повысить качество обучения глубокой нейронной сети. В работе [13] установлено, что алгоритм обратного распространения ошибки сходится быстрее, если входные данные нормализованы (имеют нулевое матожидание и единичную дисперсию). Однако при распространении сигнала по нейронной сети его матожидание и дисперсия меняются, причем иногда значительно, что негативно сказывается на процессе обучения. Сергей Йоффе и Кристиан Жегеды предложили выполнять нормализацию не только на входе в нейронную сеть, но и перед каждым слоем сети. Нормализация

выполняется отдельно для каждого *пакета данных* (mini-batch) метода стохастического градиентного спуска (или его модификаций). Пакет B содержит m элементов входных данных x_i: $B = \{x_1, ..., x_m\}$. Нормализованные значения $\hat{x}_i$ определяются по следующей формуле:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \tag{4}$$

где μ_B — среднее значение данных в пакете, σ_B^2 — дисперсия, ϵ — константа, введенная для стабильности метода. Затем, чтобы сохранить выразительность данных, выполняется сдвиг и масштабирование:

$$y_i = \gamma \hat{x}_i + \beta, \tag{5}$$

где y_i — результирующее значение, γ и β — параметры, которые определяются в процессе обучения.

Пакетная нормализация реализуется в виде слоев пакетной нормализации, которые могут быть вставлены в необходимое место в нейронной сети, в том числе несколько раз. Дополнительным преимуществом использования пакетной нормализации является сокращение времени обучения и снижение переобучения.

Методы нормализованной инициализации весов и слои пакетной нормализации помогают на практике справиться с проблемой исчезающего градиента и обучать глубокие нейронные сети, состоящие из нескольких десятков слоев. Это позволило некоторым авторам утверждать, что проблема исчезающего градиента в настоящее время решена [11].

Компания Google в 2014 году предложила новую архитектуру сверточных нейронных сетей *Inception* [102]. В этой архитектуре сеть строится из набора блоков, содержащих комбинацию операций свертки и подвыборки разной размерности. Такой подход позволяет избежать переобучения, а также снизить количество параметров сети, которые необходимо обучать, что снижает время обучения сети. Первая версия блока сети архитектуры Inception показана на рис. 8. Сеть строится из нескольких повторяющихся блоков Inception, которые следуют друг за другом.

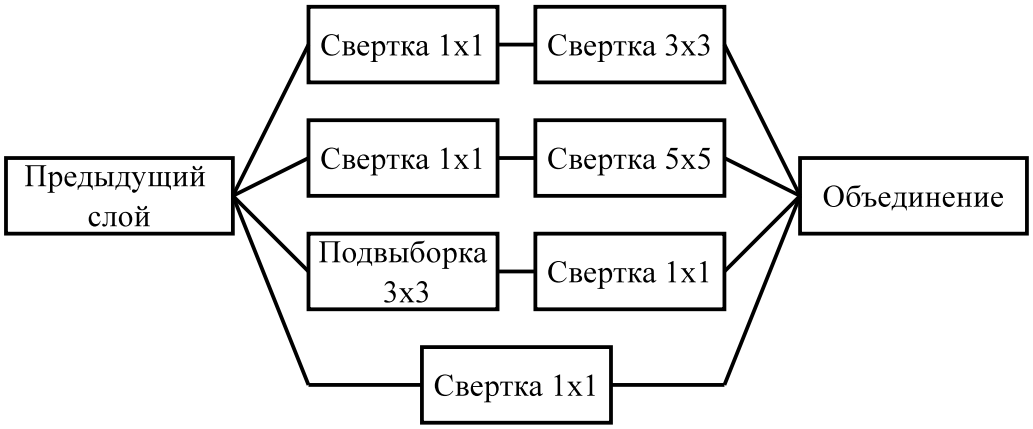

Рис. 8. Блок сети Google Inception [102]

На основе предложенной архитектуры была создана и успешно обучена сеть *GoogLeNet* [102], состоящая из 22 слоев. Сеть применяется для задач компьютерного зрения. Впоследствии Google предложила еще несколько вариантов блоков архитектуры Inception и сетей на ее основе [103, 104].

В 2014 году также была предложена новая архитектура рекуррентных нейронных сетей — управляемые рекуррентные нейронные сети (gated recurrent neural networks) [105, 106]. Такие сети похожи на сети долго-краткосрочной памяти, но в них не используются ячейки памяти. Схема блока управляемой рекуррентной сети показана на рис. 9.

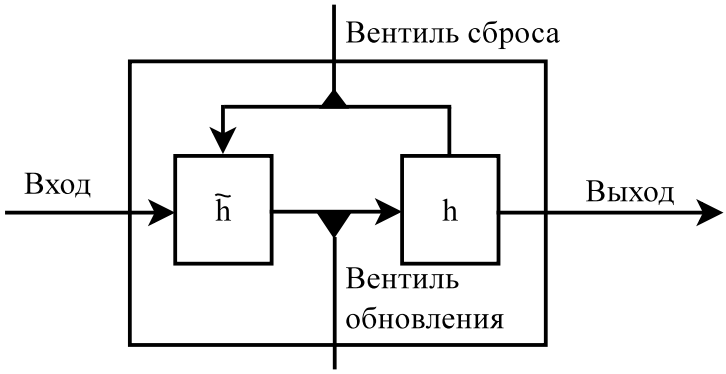

Рис. 9. Блок управляемой рекуррентной нейронной сети [105]: h — значение в блоке, $\widetilde{h}$ — новое значение

Блок управляемой нейронной сети, в отличие от сетей долго-краткосрочной памяти, не содержит выходного вентиля, поэтому значение в блоке всегда выдается наружу. Используется два типа вентилей: обновления и сброса. Вентиль обновления определяет, будет ли применяться новое значение $\tilde{h}$, а вентиль сброса задает, учитывается ли текущее значение h при расчете нового значения. Управляемые нейронные сети хорошо показали себя в задачах моделирования сигналов речи и полифонической музыки, а также для автоматического перевода [106, 107].

Увеличение количества слоев в нейронных сетях, даже с использованием полулинейных функций активации, нормализации начальных значений весов и слоев нормализации, не всегда приводит к увеличению качества обучения. При этом причина не в проблеме исчезающего градиента и не в переобучении, а в увеличении ошибки обучения сети при росте количества слоев [11, 108]. Для решения этой проблемы компания Microsoft в 2015 году предложила новую архитектуру и подход к обучению нейронных сетей — *остаточное обучение* (residual learning). Архитектура использует тот факт, что нейронную сеть всегда можно сделать более глубокой без снижения качества работы путем добавления нескольких слоев, которые не меняют сигнал. Сеть остаточного обучения, как и сеть Google Inception, строится из повторяющихся блоков. Схема блока показана на рис. 10. Блок включает два обычных слоя нейронной сети, веса в которых обучаются с помощью алгоритма обратного распространения ошибки, а также параллельный им тождественный слой, не изменяющий входных сигналов. Тождественный слой используется, если не получается обучить основные слои. Таким образом, сеть в процессе обучения сама определяет, сколько слоев нужно для решения задачи.

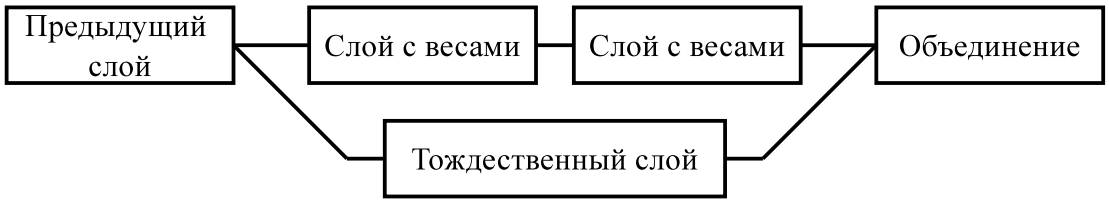

Рис. 10. Блок сети остаточного обучения [11]

Компании Microsoft удалось собрать из блоков сети остаточного обучения глубиной 34, 50, 101 и даже 152 слоя, успешно обучить их и применить для решения различных задач компьютерного зрения [11]. Компания Google также стала использовать подход остаточного обучения в сетях архитектуры Inception [104].

4. Программные системы обучения глубоких нейронных сетей

В настоящее время создано большое количество программных систем для обучения глубоких нейронных сетей [109–111]. Среди наиболее популярных из них можно отметить Caffe, Theano, TensorFlow, Torch и CNTK. Их основные характеристики приведены в табл.

Библиотека Caffe [109] — одна из самых первых популярных систем глубокого обучения. Ее разработали в центре компьютерного зрения и обучения в Беркли (Berkeley Vision and Learning Center), исходные коды стали открытыми в 2014 году. Caffe включает самое большое количество готовых к использованию предварительно обученных моделей. Система Theano [112] создана в Университете Монреаля, Канада. Theano разработана на Python, но обеспечивает высокую производительность за счет того, что программа на Python автоматически преобразуется в программу на C++, которая компилируется и затем выполняется. TensorFlow [113] создана компанией Google в 2015 году и включает системы эффективной работы с тензорами и потоковой обработки данных на графе. Библиотека Torch [114] разработана на языке Lua и предоставляет удобный высокоуровневый интерфейс для создания программ машинного обучения, аналогичный MATLAB. Высокая производительность обеспечивается, так же как и в Theano, за счет интеграции с языком C. Авторы Torch предпочли использовать Lua вместо Python из-за простоты интеграции C и Lua. Компания Microsoft разработала систему CNTK (Cognitive Toolkit) [115] и открыла ее исходные коды в 2016 году.

Все перечисленные системы глубокого обучения нейронных сетей могут использовать для ускорения обучения как многоядерные процессоры, так и ускорители вычислений GPU (включая оптимизированную библиотеку cuDNN). Причем существенным преимуществом является то, что нет необходимости переделывать программу, распараллеливание на CPU и GPU выполняется автоматически. Системы Caffe и Theano дополнительно поддерживают ускорители Intel Xeon Phi, которые также помогают существенно сократить время обучения глубоких нейронных сетей [116]. Почти все системы, кроме Theano, можно использовать для распределенного обучения нейронных сетей на вычислительных кластерах.

В дополнение к описанным выше системам можно отметить также библиотеку Keras [117], которая предоставляет удобный и простой в использовании программный интерфейс для обучения глубоких нейронных сетей. Keras не является самостоятельной системой, а работает поверх Theano, TensorFlow или CNTK. В 2016 году Keras включили в состав TensorFlow.

Заслуживают внимания также и новые библиотеки глубокого обучения, созданные недавно, но набирающие популярность. Системы PaddlePaddle [118] (создана компанией Baidu) и MXNet [119] специально разработаны для обучения глубоких нейронных сетей на распределенных кластерах. Библиотека Neon [120] разрабатывалась компанией Nervana. После покупки Nervana компанией Intel, Neon стала одной из самых быстро развивающихся библиотек с поддержкой ускорителей GPU и Intel Xeon Phi, а также большим количеством

Таблица

Программные системы обучения глубоких нейронных сетей

Свойство	**Caffe**	**Theano**	**TensorFlow**	**Torch**	**CNTK**
Базовый язык	C++	Python	C++	Lua	C++
API	C++ Python	Python	C++ Python	Lua Python	C++, C# Python
Многоядерные CPU	+	+	+	+	+
GPU	+	+	+	+	+
Xeon Phi	+	+	−	−	−
Распределенное обучение	+	−	+	+	+
Разработчик	Центр компьютерного зрения и обучения Беркли	Университет Монреаля	Google	Ронан Коллаберт	Microsoft
Открытые коды	+	+	+	+	+
Обученные сети	+	−	+	+	+

встроенных предварительно обученных нейронных сетей. MXNet и Neon показывают хорошие результаты на тестах производительности [111, 121].

Заключение

С момента возникновения нейронных сетей произошло много изменений в их архитектуре и методах обучения. В настоящее время доминирующими являются два типа архитектур: сверточные сети, которые успешно применяются для задач компьютерного зрения, и рекуррентные сети, активно использующиеся для задач обработки естественного языка.

Ранние сверточные сети обучались путем комбинации обучения с учителем и без учителя с использованием автокодировщиков и глубоких сетей доверия. Современные методы, такие как остаточное обучение, позволяют использовать только обучение с учителем и отказаться от предобучения, что ускоряет и упрощает процесс обучения. Также важным направлением в развитии сверточных нейронных сетей является передача обучения (transfer learning) [122]. Этот подход предполагает использование нейронных сетей, обученных на одних данных, для решения других типов задач. При этом применяется тонкая настройка сети и дообучение на данных от интересующей нас задачи. В результате сокращается время обучения и расширяется область применения предварительно обученных нейронных сетей. Перспективным также является совместное использование сверточных и рекуррентных нейронных сетей с обучением с подкреплением [123].

Для задач обработки естественного языка, и более общего случая обработки последовательностей, в настоящее время используются рекуррентные нейронные сети.

Среди них наиболее эффективными являются сети долго-краткосрочной памяти и управляемые рекуррентные нейронные сети, т.к. они позволяют запоминать интересующие события на длительное время [124]. Дополнительным преимуществом рекуррентных сетей является возможность обучения без учителя и без предварительно размеченного набора данных.

Широкое распространение практического применения нейронных сетей является возможным благодаря наличию большого количества готовых решений для обучения глубоких нейронных сетей [109, 112–115, 117–120], в том числе с возможностью использования современных многоядерных процессоров, ускорителей вычислений GPU и Intel Xeon Phi, а также вычислительных кластеров с распределенной памятью.

Работа поддержана проектом УрО РАН № 15-7-1-8.

Литература

1. LeCun Y., Bengio Y., Hinton G. Deep Learning // Nature. 2015. Vol. 521. P. 436–444. DOI: 10.1038/nature14539.

2. Ravì D., Wong Ch., Deligianni F., et al. Deep Learning for Health Informatics // IEEE Journal of Biomedical and Health Informatics. 2017. Vol. 21, No. 1. P. 4–21. DOI: 10.1109/JBHI.2016.2636665.

3. Schmidhuber J. Deep Learning in Neural Networks: an Overview // Neural Networks. 2015. Vol. 1. P. 85–117, DOI: 10.1016/j.neunet.2014.09.003.

4. McCulloch W.S., Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity // The Bulletin of Mathematical Biophysics. 1943. Vol. 5, No. 4. P. 115–133. DOI: 10.1007/BF02478259.

5. Hinton G., Salakhutdinov R. Reducing the Dimensionality of Data with Neural Networks // Science. 2006. Vol. 313, No. 5786. P. 504–507. DOI: 10.1126/science.1127647.

6. Hinton G.E., Osindero S., Teh Y.-W. A Fast Learning Algorithm for Deep Belief Nets // Neural Computing. 2006. Vol. 18, No. 7. P. 1527–1554. DOI: 10.1162/neco.2006.18.7.1527.

7. Síma J. Loading Deep Networks Is Hard // Neural Computation. 1994. Vol. 6, No. 5. P. 842–850. DOI: 10.1162/neco.1994.6.5.842.

8. Windisch D. Loading Deep Networks Is Hard: The Pyramidal Case // Neural Computation. 2005. Vol. 17, No. 2. P. 487–502. DOI: 10.1162/0899766053011519.

9. Gomez F.J., Schmidhuber J. Co-Evolving Recurrent Neurons Learn Deep Memory POMDPs // Proc. of the 2005 Conference on Genetic and Evolutionary Computation (GECCO) (Washington, DC, USA, June 25–29, 2005), 2005. P. 491–498. DOI: 10.1145/1068009.1068092.

10. Ciresan D.C., Meier U., Gambardella L.M., Schmidhuber J. Deep, Big, Simple Neural Nets for Handwritten Digit Recognition // Neural Computation. 2010. Vol. 22, No. 12. P. 3207–3220. DOI: 10.1162/NECO_a_00052.

11. He K., Zhang X., Ren S., et al. Deep Residual Learning for Image Recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (Las Vegas, NV, USA, 27–30 June 2016), 2016. P. 770–778. DOI: 10.1109/CVPR.2016.90.

12. Rumelhart D.E., Hinton G.E., McClelland J.L. A General Framework for Parallel Distributed Processing // Parallel Distributed Processing: Explorations in the Microstructure of Cognition. 1986. Vol. 1, P. 45–76. DOI: 10.1016/B978-1-4832-1446-7.50010-8.

13. LeCun Y., Bottou L., Orr G.B. Efficient BackProp // Neural Networks: Tricks of the Trade. 1998. P. 9–50. DOI: 10.1007/3-540-49430-8_2.

14. Broomhead D.S., Lowe D. Multivariable Functional Interpolation and Adaptive Networks // Complex Systems. Vol. 2. P. 321–355. DOI: 10.1016/0167-6911(92)90025-N.

15. Stone M.N. The Generalized Weierstrass Approximation Theorem // Mathematics Magazine. 1948. Vol. 21, No. 4. P. 167–184. DOI: 10.2307/3029750.

16. Горбань А.Н., Дунин-Барковский В.Л., Кирдин А.Н. и др. Нейроинформатика. Новосибирск: Наука. 1998. С. 296.

17. Hornik K., Stinchcombe M., White H. Multilayer Feedforward Networks are Universal Approximators // Neural Networks. 1989. Vol. 2, No. 5. P. 359–366. DOI: 10.1016/0893-6080(89)90020-8.

18. Mhaskar H.N., Micchelli Ch.A. Approximation by Superposition of Sigmoidal and Radial Basis Functions // Advances in Applied Mathematics. 1992. Vol. 13, No. 13. P. 350–373. DOI: 10.1016/0196-8858(92)90016-P.

19. Hebb D.O. The Organization of Behavior. New York: Wiley. 1949. 335 p. DOI: 10.1016/S0361-9230(99)00182-3.

20. Novikoff A.B. On Convergence Proofs on Perceptrons // Symposium on the Mathematical Theory of Automata. 1962. Vol. 12. P. 615–622.

21. Rosenblatt F. The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain // Psychological Review. 1958. P. 65–386. DOI: 10.1037/h0042519.

22. Widrow B., Hoff M. Associative Storage and Retrieval of Digital Information in Networks of Adaptive Neurons // Biological Prototypes and Synthetic Systems. 1962. Vol. 1. 160 p. DOI: 10.1007/978-1-4684-1716-6_25.

23. Narendra K.S., Thathatchar M.A.L. Learning Automata – a Survey // IEEE Transactions on Systems, Man, and Cybernetics. 1974. Vol. 4. P. 323–334. DOI: 10.1109/tsmc.1974.5408453.

24. Rosenblatt F. Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms. 1962. Washington: Spartan Books. 616 p. DOI: 10.1007/978-3-642-70911-1_20.

25. Grossberg S. Some Networks That Can Learn, Remember, and Reproduce any Number of Complicated Space-Time Patterns // International Journal of Mathematics and Mechanics. 1969. Vol. 19. P. 53–91. DOI: 10.1512/iumj.1970.19.19007.

26. Kohonen T. Correlation Matrix Memories // IEEE Transactions on Computers. 1972. Vol. 100, No. 4. P. 353–359. DOI: 10.1109/tc.1972.5008975.

27. von der Malsburg C. Self-Organization of Orientation Sensitive Cells in the Striate Cortex // Kybernetik. 1973. Vol. 14, No. 2. P. 85–100. DOI: 10.1007/bf00288907.

28. Willshaw D.J., von der Malsburg C. How Patterned Neural Connections Can Be Set Up by Self-Organization // Proceedings of the Royal Society London B. 1976. Vol. 194. P. 431–445. DOI: 10.1098/rspb.1976.0087.

29. Ivakhnenko A.G. Heuristic Self-Organization in Problems of Engineering Cybernetics // Automatica. 1970. Vol. 6, No. 2. P. 207–219. DOI: 10.1016/0005-1098(70)90092-0.

30. Ivakhnenko A.G. Polynomial Theory of Complex Systems // IEEE Transactions on Systems, Man and Cybernetics. 1971. Vol. 4. P. 364–378. DOI: 10.1109/tsmc.1971.4308320.

31. Ikeda S., Ochiai M., Sawaragi Y. Sequential GMDH Algorithm and Its Application to River Flow Prediction // IEEE Transactions on Systems, Man and Cybernetics. 1976. Vol. 7. P. 473–479. DOI: 10.1109/tsmc.1976.4309532.

32. Witczak M, Korbicz J, Mrugalski M., et al. A GMDH Neural Network-Based Approach to Robust Fault Diagnosis: Application to the DAMADICS Benchmark Problem // Control Engineering Practice. 2006. Vol. 14, No. 6. P. 671–683. DOI: 10.1016/j.conengprac.2005.04.007.

33. Kondo T., Ueno J. Multi-Layered GMDH-type Neural Network Self-Selecting Optimum Neural Network Architecture and Its Application to 3-Dimensional Medical Image Recognition of Blood Vessels // International Journal of Innovative Computing, Information and Control. 2008. Vol. 4, No. 1. P. 175–187.

34. Linnainmaa S. The Representation of the Cumulative Rounding Error of an Algorithm as a Taylor Expansion of the Local Rounding Errors. University of Helsinki. 1970.

35. Linnainmaa S. Taylor Expansion of the Accumulated Rounding Error // BIT Numerical Mathematics. 1976. Vol. 16, No. 2. P. 146–160. DOI: 10.1007/bf01931367.

36. Werbos P.J. Applications of Advances in Nonlinear Sensitivity Analysis // Lecture Notes in Control and Information Sciences. 1981. Vol. 38, P. 762–770. DOI: 10.1007/BFb0006203.

37. Parker D.B. Learning Logic. Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA. 1985.

38. LeCun Y. A Theoretical Framework for Back-Propagation // Proceedings of the 1988 Connectionist Models Summer School (Pittsburgh, Pennsylvania, USA, June 17–26, 1988), 1988. P. 21–28.

39. Rumelhart D.E., Hinton G.E., Williams R.J. Learning Internal Representations by Error Propagation // Parallel Distributed Processing. 1986. Vol. 1. P. 318–362. DOI: 10.1016/b978-1-4832-1446-7.50035-2.

40. Qian N. On the Momentum Term in Gradient Descent Learning Algorithms // Neural Networks: The Official Journal of the International Neural Network Society. 1999. Vol. 12, No. 1. P. 145–151. DOI: 10.1016/s0893-6080(98)00116-6.

41. Duchi J., Hazan E., Singer Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization // Journal of Machine Learning Research. 2011. Vol. 12. P. 2121–2159.

42. Kingma D.P., Ba J.L. Adam: a Method for Stochastic Optimization // International Conference on Learning Representations (San Diego, USA, May 7-9, 2015), 2015. P. 1–13.

43. Fukushima K. Neocognitron: a Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position // Biological Cybernetics. 1980. Vol. 36, No. 4. P. 193–202. DOI: 10.1007/BF00344251.

44. Wiesel D.H., Hubel T.N. Receptive Fields of Single Neurones in the Cat's Striate Cortex // The Journal of Physiology. 1959. Vol. 148, No. 3. P. 574–591. DOI: 10.1113/jphysiol.1959.sp006308.

45. Fukushima K. Artificial Vision by Multi-Layered Neural Networks: Neocognitron and its Advances // Neural Networks. 2013. Vol. 37. P. 103–119. DOI: 10.1016/j.neunet.2012.09.016.

46. Fukushima K. Training Multi-Layered Neural Network Neocognitron // Neural Networks. 2013. Vol. 40. P. 18–31. DOI: 10.1016/j.neunet.2013.01.001.

47. Fukushima K. Increasing Robustness Against Background Noise: Visual Pattern Recognition by a Neocognitron // Neural Networks. 2011. Vol. 24, No. 7. P. 767–778. DOI: 10.1016/j.neunet.2011.03.017.

48. Ballard D.H. Modular Learning in Neural Networks // Proceedings of the Sixth National Conference on Artificial Intelligence (Seattle, Washington, USA, July 13–17, 1987), 1987. Vol. 1. P. 279–284.

49. Hinton G.E., McClelland J.L. Learning Representations by Recirculation // Neural Information Processing Systems. 1998. American Institute of Physics. P. 358–366.

50. Wolpert D.H. Stacked Generalization // Neural Networks. 1992. Vol. 5, No. 2. P. 241–259. DOI: 10.1016/s0893-6080(05)80023-1.

51. Ting K.M., Witten I.H. Stacked Generalization: When Does It Work? // Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) (Nagoya, Japan, August 23–29, 1997), 1997. P. 866–871.

52. LeCun Y., Boser B., Denker J.S., et al. Back-Propagation Applied to Handwritten Zip Code Recognition // Neural Computation. 1998. Vol. 1, No. 4. P. 541–551. DOI: 10.1162/neco.1989.1.4.541.

53. LeCun Y., Boser B., Denker J.S., et al. Handwritten Digit Recognition with a Back-Propagation Network // Advances in Neural Information Processing Systems 2. Morgan Kaufmann. 1990. P. 396–404.

54. Baldi P., Chauvin Y. Neural Networks for Fingerprint Recognition // Neural Computation. 1993. Vol. 5, No. 3. P. 402–418. DOI: 10.1007/978-3-642-76153-9_35.

55. Elman J.L. Finding Structure in Time // Cognitive Science. 1990. Vol. 14, No. 2. P. 179–211. DOI: 10.1207/s15516709cog1402_1.

56. Jordan M.I. Serial Order: a Parallel Distributed Processing Approach. Institute for Cognitive Science, University of California, San Diego. ICS Report 8604. 1986. P. 40.

57. Jordan M.I. Serial Order: a Parallel Distributed Processing Approach // Advances in Psychology. 1997. Vol. 121. P. 471–495. DOI: 10.1016/s0166-4115(97)80111-2.

58. Hochreiter S. Untersuchungen zu Dynamischen Neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer. Technische Universität München, 1991.

59. Hochreiter S., Bengio Y., Frasconi P., et al. Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies // A Field Guide to Dynamical Recurrent Neural Networks. Wiley-IEEE Press. 2001. P. 237–243. DOI: 10.1109/9780470544037.ch14.

60. Bengio Y., Simard P., Frasconi P. Learning Long-Term Dependencies with Gradient Descent is Difficult // IEEE Transactions on Neural Networks. 1994. Vol. 5, No. 2. P. 157–166. DOI: 10.1109/72.279181.

61. Tiňo P., Hammer B. Architectural Bias in Recurrent Neural Networks: Fractal Analysis // Neural Computation. 2004. Vol. 15, No. 8. P. 1931–1957. DOI: 10.1162/08997660360675099.

62. Hochreiter S., Schmidhuber J. Bridging Long Time Lags by Weight Guessing and "Long Short-Term Memory" // Spatiotemporal Models in Biological and Artificial Systems. 1996. Vol. 37. P. 65–72.

63. Schmidhuber J., Wierstra D., Gagliolo M., et al. Training Recurrent Networks by Evolino. // Neural Computation. 2007. Vol. 19, No. 3. P. 757–779. DOI: 10.1162/neco.2007.19.3.757.

64. Levin L.A. Universal Sequential Search Problems // Problems of Information Transmission. 1997. Vol. 9, No. 3. P. 265–266.

65. Schmidhuber J. Discovering Neural Nets with Low Kolmogorov Complexity and High Generalization Capability // Neural Networks. 1997. Vol. 10, No. 5. P. 857–873. DOI: 10.1016/s0893-6080(96)00127-x.

66. Møller, M.F. Exact Calculation of the Product of the Hessian Matrix of Feed-Forward Network Error Functions and a Vector in O(N) Time. Computer Science Department, Aarhus University, Denmark. 1993. No. PB-432. DOI: 10.7146/dpb.v22i432.6748.

67. Pearlmutter B.A. Fast Exact Multiplication by the Hessian // Neural Computation. 1994. Vol. 6, No. 1. P. 147–160. DOI: 10.1162/neco.1994.6.1.147.

68. Schraudolph N.N. Fast Curvature Matrix-Vector Products for Second-Order Gradient Descent // Neural Computation. 2002. Vol. 14, No. 7. P. 1723–1738. DOI: 10.1162/08997660260028683.

69. Martens J. Deep Learning via Hessian-Free Optimization // Proceedings of the 27th International Conference on Machine Learning (ICML-10) (Haifa, Israel, June 21–24, 2010), 2010. P. 735–742.

70. Martens J., Sutskever I. Learning Recurrent Neural Networks with Hessian-Free Optimization // Proceedings of the 28th International Conference on Machine Learning (ICML-11) (Bellevue, Washington, USA, June 28 – July 02, 2011), 2011. P. 1033–1040.

71. Schmidhuber J. Learning Complex, Extended Sequences Using the Principle of History Compression // Neural Computation. 1992. Vol. 4, No. 2. P. 234–242. DOI: 10.1162/neco.1992.4.2.234.

72. Connor J., Martin D.R., Atlas L.E. Recurrent Neural Networks and Robust Time Series Prediction // IEEE Transactions on Neural Networks. 1994. Vol. 5, No. 2. P. 240–254. DOI: 10.1109/72.279188.

73. Dorffner G. Neural Networks for Time Series Processing // Neural Network World. 1996. Vol. 6. P. 447–468.

74. Schmidhuber J., Mozer M.C., Prelinger D. Continuous History Compression // Proceedings of International Workshop on Neural Networks (Aachen, Germany, 1993), 1993. P. 87–95.

75. Hochreiter S., Schmidhuber J. Long Short-Term Memory // Neural Computation. 1997. Vol. 9, No. 8. P. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

76. Gers F.A., Schmidhuber J., Cummins F. Learning to Forget: Continual Prediction with LSTM // Neural Computation. 2000. Vol. 12, No. 10. P. 2451–2471. DOI: 10.1162/089976600300015015.

77. Pérez-Ortiz J.A., Gers F.A., Eck D., et al. Kalman Filters Improve LSTM Network Performance in Problems Unsolvable by Traditional Recurrent Nets // Neural Networks. 2003. Vol. 16, No. 2. P. 241–250. DOI: 10.1016/s0893-6080(02)00219-8.

78. Weng J., Ahuja N., Huang T.S. Cresceptron: a Self-Organizing Neural Network Which Grows Adaptively // International Joint Conference on Neural Networks (IJCNN) (Baltimore, MD, USA, 7–11 June 1992). 1992. Vol. 1. P. 576–581. DOI: 10.1109/ijcnn.1992.287150.

79. Weng J.J., Ahuja N., Huang T.S. Learning Recognition and Segmentation Using the Cresceptron // International Journal of Computer Vision. 1997. Vol. 25, No. 2. P. 109–143. DOI: 10.1023/a:1007967800668.

80. Ranzato M.A., Huang F.J., Boureau Y.L., et al. Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition // IEEE Conference on Computer Vision and Pattern Recognition (Minneapolis, MN, USA, 17–22 June 2007), 2007. P. 1–8. DOI: 10.1109/cvpr.2007.383157.

81. Scherer D., Müller A., Behnke S. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition // Lecture Notes in Computer Science. 2010. Vol. 6354, P. 92–101. DOI: 10.1007/978-3-642-15825-4_10.

82. Smolensky P. Information Processing in Dynamical Systems: Foundations of Harmony Theory // Parallel Distributed Processing: Explorations in the Microstructure of Cognition. 1986. Vol. 1. P. 194–281.

83. Hinton G.E., Sejnowski T.E. Learning and Relearning in Boltzmann Machines // Parallel Distributed Processing. 1986. Vol. 1. P. 282–317.

84. Memisevic R., Hinton G.E. Learning to Represent Spatial Transformations with Factored Higher-Order Boltzmann Machines // Neural Computation. 2010. Vol. 22, No. 6. P. 1473–1492. DOI: 10.1162/neco.2010.01-09-953.

85. Mohamed A., Hinton G.E. Phone Recognition Using Restricted Boltzmann Machines // IEEE International Conference on Acoustics, Speech and Signal Processing (Dallas, TX, USA, 14–19 March 2010), 2010. P. 4354–4357. DOI: 10.1109/icassp.2010.5495651.

86. Salakhutdinov R., Hinton G. Semantic Hashing // International Journal of Approximate Reasoning. 2009. Vol. 50, No. 7. P. 969–978. DOI: 10.1016/j.ijar.2008.11.006.

87. Bengio Y., Lamblin P., Popovici D., et al. Greedy Layer-Wise Training of Deep Networks // Advances in Neural Information Processing Systems 19. 2007. P. 153–160.

88. Vincent P., Hugo L., Bengio Y., et al. Extracting and Composing Robust Features with Denoising Autoencoders // Proceedings of the 25th international Conference on Machine learning (Helsinki, Finland, July 05–09, 2008). 2008. P. 1096–1103. DOI: 10.1145/1390156.1390294.

89. Erhan D., Bengio Y., Courville A., et al. Why Does Unsupervised Pre-Training Help Deep Learning? // Journal of Machine Learning Research. 2010. Vol. 11. P. 625–660.

90. Arel I., Rose D.C., Karnowski T.P. Deep Machine Learning – a New Frontier in Artificial Intelligence Research // Computational Intelligence Magazine, IEEE. 2010. Vol. 5, No. 4. P. 13–18. DOI: 10.1109/mci.2010.938364.

91. Viren J., Sebastian S. Natural Image Denoising with Convolutional Networks // Advances in Neural Information Processing Systems (NIPS) 21. 2009. P. 769–776.

92. Razavian A.Sh., Azizpour H., Sullivan J., at al. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition // Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (Washington, DC, USA, June 23–28, 2014), 2014. P. 512–519. DOI: 10.1109/cvprw.2014.131.

93. Ruochen W., Zhe X. A Pedestrian and Vehicle Rapid Identification Model Based on Convolutional Neural Network // Proceedings of the 7th International Conference on Internet Multimedia Computing and Service (ICIMCS '15) (Zhangjiajie, China, August 19–21, 2015), 2015. P. 32:1–32:4. DOI: 10.1145/2808492.2808524.

94. Boominathan L., Kruthiventi S.S., Babu R.V. CrowdNet: A Deep Convolutional Network for Dense Crowd Counting // Proceedings of the 2016 ACM on Multimedia Conference (Amsterdam, The Netherlands, October 15–19, 2016), 2016. P. 640–644. DOI: 10.1145/2964284.2967300.

95. Kinnikar A., Husain M., Meena S.M. Face Recognition Using Gabor Filter And Convolutional Neural Network // Proceedings of the International Conference on Informatics and Analytics (Pondicherry, India, August 25–26, 2016), 2016. P. 113:1–113:4. DOI: 10.1145/2980258.2982104.

96. Hahnloser R.H.R., Sarpeshkar R., Mahowald M.A., et al. Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit // Nature. 2000. Vol. 405. P. 947–951. DOI: 10.1038/35016072.

97. Hahnloser R.H.R., Seung H.S., Slotine J.J. Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks // Neural Computation. 2003. Vol. 15, No. 3. P. 621–638. DOI: 10.1162/089976603321192103.

98. Glorot X., Bordes A., Bengio Y. Deep Sparse Rectifier Neural Networks // Journal of Machine Learning Research. 2011. Vol. 15. P. 315–323.

99. Glorot X., Bengio Y. Understanding the Difficulty of Training Deep Feedforward Neural Networks // Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10) (Sardinia, Italy, May 13–15, 2010). Society for Artificial Intelligence and Statistics. 2010. P. 249–256.

100. He K., Zhang X., Ren Sh. et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification // Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV) (Santiago, Chile, December 7–13, 2015), 2015. P. 1026–1034. DOI: 10.1109/ICCV.2015.123.

101. Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift // JMLR Workshop and Conference Proceedings. Proceedings of the 32nd International Conference on Machine Learning (Lille, France, July 06–11, 2015), 2015. Vol. 37. P. 448–456.

102. Szegedy C., Liu W, Jia Y. et al. Going Deeper with Convolutions // IEEE Conference on Computer Vision and Pattern Recognition (Boston, MA, USA, June 7–12, 2015), 2015. P. 1–9. DOI: 10.1109/CVPR.2015.7298594.

103. Szegedy C., Vanhoucke V., Ioffe S., et al. Rethinking the Inception Architecture for Computer Vision // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Seattle, WA, USA, Jun 27–30, 2016), 2016. P. 2818–2826. DOI: 10.1109/cvpr.2016.308.

104. Szegedy C., Ioffe S., Vanhoucke V., et al. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning // Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) (San Francisco, California, USA, February 4–9, 2017), 2017. P. 4278–4284.

105. Cho K., van Merrienboer B., Gulcehre C., et al. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation // Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Doha, Qatar, October 25–29, 2014), 2014. P. 1724–1734. DOI: 10.3115/v1/d14-1179.

106. Cho K., van Merrienboer B., Bahdanau D., et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches // Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (Doha, Qatar, October 25, 2014), 2014. P. 103–111. DOI: 10.3115/v1/w14-4012.

107. Chung, J., Gulcehre, C., Cho, K., et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling // NIPS 2014 Workshop on Deep Learning (Montreal, Canada, December 12, 2014), 2014. P. 1–9.

108. He K., Sun J. Convolutional Neural Networks at Constrained Time Cost // 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Boston, MA, USA, June 07–12, 2015), 2015. P. 5353–5360. DOI: 10.1109/CVPR.2015.7299173.

109. Jia Y., Shelhamer E., Donahue J., et al. Caffe: Convolutional Architecture for Fast Feature Embedding // Proceedings of the 22nd ACM International Conference on Multimedia (Orlando, FL, USA, November 03–07, 2014), 2014. P. 675–678. DOI: 10.1145/2647868.2654889

110. Kruchinin D., Dolotov E., Kornyakov K. et al. Comparison of Deep Learning Libraries on the Problem of Handwritten Digit Classification // Analysis of Images, Social Networks and Texts. Communications in Computer and Information Science. 2015. Vol. 542. P. 399–411. DOI: 10.1007/978-3-319-26123-2_38.

111. Bahrampour S., Ramakrishnan N., Schott L., et al. Comparative Study of Deep Learning Software Frameworks. URL: `https://arxiv.org/abs/1511.06435` (дата обращения: 02.07.2017).

112. Bergstra J., Breuleux O., Bastien F., et al. Theano: a CPU and GPU Math Expression Compiler // Proceedings of the Python for Scientific Computing Conference (SciPy) (Austin, TX, USA, June 28 – July 3, 2010), 2010. P. 3–10.

113. Abadi M., Agarwal A. Barham P. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems // Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16) (Savannah, GA, USA, November, 2–4, 2016), 2016. P. 265–283.

114. Collobert R., Kavukcuoglu K., Farabet C. Torch7: a Matlab-like Environment for Machine Learning // BigLearn, NIPS Workshop (Granada, Spain, December 12–17, 2011), 2011.

115. Seide F., Agarwal A. CNTK: Microsoft's Open-Source Deep-Learning Toolkit // Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16) (San Francisco, California, USA, August 13–17, 2016), 2016. P. 2135–2135. DOI: 10.1145/2939672.2945397.

116. Viebke A., Pllana S. The Potential of the Intel(r) Xeon Phi for Supervised Deep Learning // IEEE 17th International Conference on High Performance Computing and

Communications (HPCC) (New York, USA, August 24–26, 2015), 2015. P. 758–765. DOI: 10.1109/hpcc-css-icess.2015.45.

117. Chollet. F., et al. Keras. 2015. URL: https://github.com/fchollet/keras (дата обращения: 02.07.2017).

118. PadlePadle: PArallel Distributed Deep LEarning. URL: http://www.paddlepaddle.org/ (дата обращения: 02.07.2017).

119. Chen T., Li M., Li Y. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. URL: https://arxiv.org/abs/1512.01274 (дата обращения: 02.07.2017).

120. Intel Nervana Reference Deep Learning Framework Committed to Best Performance on all Hardware. URL: https://www.intelnervana.com/neon/ (дата обращения: 02.07.2017).

121. Shi Sh., Wang Q., Xu P. Benchmarking State-of-the-Art Deep Learning Software Tools. URL: https://arxiv.org/abs/1608.07249 (дата обращения: 02.07.2017).

122. Weiss K., Khoshgoftaar T.M., Wang D. A Survey of Transfer Learning // Journal of Big Data. 2016. Vol. 3, No. 1. P. 1–9. DOI: 10.1186/s40537-016-0043-6

123. Ba J., Mnih V., Kavukcuoglu K. Multiple Object Recognition with Visual Attention // Proceedings of the International Conference on Learning Representations (ICLR) (San Diego, USA,May 7–9, 2015), 2015. P. 1–10.

124. Graves A., Mohamed A.R., Hinton G. Speech Recognition with Deep Recurrent Neural Networks // IEEE International Conference on Acoustics, Speech and Signal Processing (Vancouver, Canada, May 26–31, 2013), 2013. P. 6645–6649. DOI: 10.1109/ICASSP.2013.6638947.

Созыкин Андрей Владимирович, к.т.н., зав. отделом вычислительной техники, Институт математики и механики им. Н.Н.Красовского УрО РАН, зав. кафедрой, высокопроизводительных компьютерных технологий, Уральский федеральный университет (Екатеринбург, Российская Федерация)

DOI: 10.14529/cmse170303

AN OVERVIEW OF METHODS FOR DEEP LEARNING IN NEURAL NETWORKS

© 2017 A.V. Sozykin

N.N. Krasovskii Institute of Mathematics and Mechanics (S.Kovalevskaya str. 16, Yekaterinburg, 620990 Russia),

Ural Federal University (Mira str. 19, Yekaterinburg, 620002 Russia)

E-mail: avs@imm.uran.ru

Received: 12.04.2017

At present, deep learning is becoming one of the most popular approach to creation of the artificial intelligences systems such as speech recognition, natural language processing, computer vision and so on. The paper presents a historical overview of deep learning in neural networks. The model of the artificial neural network is described as well as the learning algorithms for neural networks including the error backpropagation algorithm,

which is used to train deep neural networks. The development of neural networks architectures is presented including neocognitron, autoencoders, convolutional neural networks, restricted Boltzmann machine, deep belief networks, long short-term memory, gated recurrent neural networks, and residual networks. Training deep neural networks with many hidden layers is impeded by the vanishing gradient problem. The paper describes the approaches to solve this problem that provide the ability to train neural networks with more than hundred layers. An overview of popular deep learning libraries is presented. Nowadays, for computer vision tasks convolutional neural networks are utilized, while for sequence processing, including natural language processing, recurrent networks are preferred solution, primarily long short-term memory networks and gated recurrent neural networks.

Keywords: deep learning, neural networks, machine learning.

FOR CITATION

Sozykin A.V. An Overview of Methods for Deep Learning in Neural Networks.. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2017. vol. 6, no. 3. pp. 28–59. (in Russian) DOI: 10.14529/cmse170303.

References

1. LeCun Y., Bengio Y., Hinton G. Deep Learning. Nature. 2015. vol. 521. pp. 436–444. DOI: 10.1038/nature14539.
2. Ravì D., Wong Ch., Deligianni F., et al. Deep Learning for Health Informatics. IEEE Journal of Biomedical and Health Informatics. 2017. vol. 21, no. 1. pp. 4–21. DOI: 10.1109/JBHI.2016.2636665.
3. Schmidhuber J. Deep Learning in Neural Networks: an Overview. Neural Networks. 2015. vol. 1. pp. 85–117, DOI: 10.1016/j.neunet.2014.09.003.
4. McCulloch W.S., Pitts W. A Logical Calculus of the Ideas Immanent in Nervous Activity. The Bulletin of Mathematical Biophysics. 1943. vol. 5, no. 4. pp. 115–133. DOI: 10.1007/BF02478259.
5. Hinton G., Salakhutdinov R. Reducing the Dimensionality of Data with Neural Networks. Science. 2006. vol. 313, no. 5786. pp. 504–507. DOI: 10.1126/science.1127647.
6. Hinton G.E., Osindero S., Teh Y.-W. A Fast Learning Algorithm for Deep Belief Nets. Neural Computing. 2006. vol. 18, no. 7. pp. 1527–1554. DOI: 10.1162/neco.2006.18.7.1527.
7. Síma J. Loading Deep Networks Is Hard. Neural Computation. 1994. vol. 6, no. 5. pp. 842–850. DOI: 10.1162/neco.1994.6.5.842.
8. Windisch D. Loading Deep Networks Is Hard: The Pyramidal Case. Neural Computation. 2005. vol. 17, no. 2. pp. 487–502. DOI: 10.1162/0899766053011519.
9. Gomez F.J., Schmidhuber J. Co-Evolving Recurrent Neurons Learn Deep Memory POMDPs. Proc. of the 2005 Conference on Genetic and Evolutionary Computation (GECCO) (Washington, DC, USA, June 25–29, 2005), 2005. pp. 491–498. DOI: 10.1145/1068009.1068092.
10. Ciresan D.C., Meier U., Gambardella L.M., Schmidhuber J. Deep, Big, Simple Neural Nets for Handwritten Digit Recognition. Neural Computation. 2010. vol. 22, no. 12. pp. 3207–3220. DOI: 10.1162/NECO_a_00052.

11. He K., Zhang X., Ren S., et al. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (Las Vegas, NV, USA, 27–30 June 2016), 2016. pp. 770–778. DOI: 10.1109/CVPR.2016.90.

12. Rumelhart D.E., Hinton G.E., McClelland J.L. A General Framework for Parallel Distributed Processing. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. 1986. vol. 1, pp. 45–76. DOI: 10.1016/B978-1-4832-1446-7.50010-8.

13. LeCun Y., Bottou L., Orr G.B. Efficient BackProp. Neural Networks: Tricks of the Trade. 1998. pp. 9–50. DOI: 10.1007/3-540-49430-8_2.

14. Broomhead D.S., Lowe D. Multivariable Functional Interpolation and Adaptive Networks. Complex Systems. vol. 2. pp. 321–355. DOI: 10.1016/0167-6911(92)90025-N.

15. Stone M.N. The Generalized Weierstrass Approximation Theorem. Mathematics Magazine. 1948. vol. 21, no. 4. pp. 167–184. DOI: 10.2307/3029750.

16. Gorban A.N., Dudin-Barkovsky V.L., Kirdin A.N., et al. *Nejroinformatika* [Neuroinformatics]. Novosibirsk, Science, 1998. 296 p.

17. Hornik K., Stinchcombe M., White H. Multilayer Feedforward Networks are Universal Approximators. Neural Networks. 1989. vol. 2, no. 5. pp. 359–366. DOI: 10.1016/0893-6080(89)90020-8.

18. Mhaskar H.N., Micchelli Ch.A. Approximation by Superposition of Sigmoidal and Radial Basis Functions. Advances in Applied Mathematics. 1992. vol. 13, no. 13. pp. 350–373. DOI: 10.1016/0196-8858(92)90016-P.

19. Hebb D.O. The Organization of Behavior. New York: Wiley. 1949. 335 p. DOI: 10.1016/S0361-9230(99)00182-3.

20. Novikoff A.B. On Convergence Proofs on Perceptrons. Symposium on the Mathematical Theory of Automata. 1962. vol. 12. pp. 615–622.

21. Rosenblatt F. The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain. Psychological Review. 1958. pp. 65–386. DOI: 10.1037/h0042519.

22. Widrow B., Hoff M. Associative Storage and Retrieval of Digital Information in Networks of Adaptive Neurons. Biological Prototypes and Synthetic Systems. 1962. vol. 1. 160 p. DOI: 10.1007/978-1-4684-1716-6_25.

23. Narendra K.S., Thathatchar M.A.L. Learning Automata – a Survey. IEEE Transactions on Systems, Man, and Cybernetics. 1974. vol. 4. pp. 323–334. DOI: 10.1109/tsmc.1974.5408453.

24. Rosenblatt F. Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms. 1962. Washington: Spartan Books. 616 p. DOI: 10.1007/978-3-642-70911-1_20.

25. Grossberg S. Some Networks That Can Learn, Remember, and Reproduce any Number of Complicated Space-Time Patterns. International Journal of Mathematics and Mechanics. 1969. vol. 19. pp. 53–91. DOI: 10.1512/iumj.1970.19.19007.

26. Kohonen T. Correlation Matrix Memories. IEEE Transactions on Computers. 1972. vol. 100, no. 4. pp. 353–359. DOI: 10.1109/tc.1972.5008975.

27. von der Malsburg C. Self-Organization of Orientation Sensitive Cells in the Striate Cortex. Kybernetik. 1973. vol. 14, no. 2. pp. 85–100. DOI: 10.1007/bf00288907.

28. Willshaw D.J., von der Malsburg C. How Patterned Neural Connections Can Be Set Up by Self-Organization. Proceedings of the Royal Society London B. 1976. vol. 194. pp. 431–445. DOI: 10.1098/rspb.1976.0087.

29. Ivakhnenko A.G. Heuristic Self-Organization in Problems of Engineering Cybernetics. Automatica. 1970. vol. 6, no. 2. pp. 207–219. DOI: 10.1016/0005-1098(70)90092-0.

30. Ivakhnenko A.G. Polynomial Theory of Complex Systems. IEEE Transactions on Systems, Man and Cybernetics. 1971. vol. 4. pp. 364–378. DOI: 10.1109/tsmc.1971.4308320.

31. Ikeda S., Ochiai M., Sawaragi Y. Sequential GMDH Algorithm and Its Application to River Flow Prediction. IEEE Transactions on Systems, Man and Cybernetics. 1976. vol. 7. pp. 473–479. DOI: 10.1109/tsmc.1976.4309532.

32. Witczak M, Korbicz J, Mrugalski M., et al. A GMDH Neural Network-Based Approach to Robust Fault Diagnosis: Application to the DAMADICS Benchmark Problem. Control Engineering Practice. 2006. vol. 14, no. 6. pp. 671–683. DOI: 10.1016/j.conengprac.2005.04.007.

33. Kondo T., Ueno J. Multi-Layered GMDH-type Neural Network Self-Selecting Optimum Neural Network Architecture and Its Application to 3-Dimensional Medical Image Recognition of Blood Vessels. International Journal of Innovative Computing, Information and Control. 2008. vol. 4, no. 1. pp. 175–187.

34. Linnainmaa S. The Representation of the Cumulative Rounding Error of an Algorithm as a Taylor Expansion of the Local Rounding Errors. University of Helsinki. 1970.

35. Linnainmaa S. Taylor Expansion of the Accumulated Rounding Error. BIT Numerical Mathematics. 1976. vol. 16, no. 2. pp. 146–160. DOI: 10.1007/bf01931367.

36. Werbos P.J. Applications of Advances in Nonlinear Sensitivity Analysis. Lecture Notes in Control and Information Sciences. 1981. vol. 38, pp. 762–770. DOI: 10.1007/BFb0006203.

37. Parker D.B. Learning Logic. Technical Report TR-47, Center for Computational Research in Economics and Management Science, Massachusetts Institute of Technology, Cambridge, MA. 1985.

38. LeCun Y. A Theoretical Framework for Back-Propagation. Proceedings of the 1988 Connectionist Models Summer School (Pittsburgh, Pennsylvania, USA, June 17–26, 1988), 1988. P. 21–28.

39. Rumelhart D.E., Hinton G.E., Williams R.J. Learning Internal Representations by Error Propagation. Parallel Distributed Processing. 1986. vol. 1. pp. 318–362. DOI: 10.1016/b978-1-4832-1446-7.50035-2.

40. Qian N. On the Momentum Term in Gradient Descent Learning Algorithms. Neural Networks: The Official Journal of the International Neural Network Society. 1999. vol. 12, no. 1. pp. 145–151. DOI: 10.1016/s0893-6080(98)00116-6.

41. Duchi J., Hazan E., Singer Y. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. Journal of Machine Learning Research. 2011. vol. 12. pp. 2121–2159.

42. Kingma D.P., Ba J.L. Adam: a Method for Stochastic Optimization. International Conference on Learning Representations (San Diego, USA, May 7-9, 2015), 2015. pp. 1–13.

43. Fukushima K. Neocognitron: a Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position. Biological Cybernetics. 1980. vol. 36, no. 4. pp. 193–202. DOI: 10.1007/BF00344251.

44. Wiesel D.H., Hubel T.N. Receptive Fields of Single Neurones in the Cat's Striate Cortex. The Journal of Physiology. 1959. vol. 148, no. 3. pp. 574–591. DOI: 10.1113/jphysiol.1959.sp006308.

45. Fukushima K. Artificial Vision by Multi-Layered Neural Networks: Neocognitron and its Advances. Neural Networks. 2013. vol. 37. pp. 103–119. DOI: 10.1016/j.neunet.2012.09.016.

46. Fukushima K. Training Multi-Layered Neural Network Neocognitron. Neural Networks. 2013. vol. 40. pp. 18–31. DOI: 10.1016/j.neunet.2013.01.001.

47. Fukushima K. Increasing Robustness Against Background Noise: Visual Pattern Recognition by a Neocognitron. Neural Networks. 2011. vol. 24, no. 7. pp. 767–778. DOI: 10.1016/j.neunet.2011.03.017.

48. Ballard D.H. Modular Learning in Neural Networks. Proceedings of the Sixth National Conference on Artificial Intelligence (Seattle, Washington, USA, July 13–17, 1987), 1987. vol. 1. pp. 279–284.

49. Hinton G.E., McClelland J.L. Learning Representations by Recirculation. Neural Information Processing Systems. 1998. American Institute of Physics. pp. 358–366.

50. Wolpert D.H. Stacked Generalization. Neural Networks. 1992. vol. 5, no. 2. pp. 241–259. DOI: 10.1016/s0893-6080(05)80023-1.

51. Ting K.M., Witten I.H. Stacked Generalization: When Does It Work? Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) (Nagoya, Japan, August 23–29, 1997), 1997. pp. 866–871.

52. LeCun Y., Boser B., Denker J.S., et al. Back-Propagation Applied to Handwritten Zip Code Recognition. Neural Computation. 1998. vol. 1, no. 4. pp. 541–551. DOI: 10.1162/neco.1989.1.4.541.

53. LeCun Y., Boser B., Denker J.S., et al. Handwritten Digit Recognition with a Back-Propagation Network. Advances in Neural Information Processing Systems 2. Morgan Kaufmann. 1990. pp. 396–404.

54. Baldi P., Chauvin Y. Neural Networks for Fingerprint Recognition. Neural Computation. 1993. vol. 5, no. 3. pp. 402–418. DOI: 10.1007/978-3-642-76153-9_35.

55. Elman J.L. Finding Structure in Time. Cognitive Science. 1990. vol. 14, no. 2. pp. 179–211. DOI: 10.1207/s15516709cog1402_1.

56. Jordan M.I. Serial Order: a Parallel Distributed Processing Approach. Institute for Cognitive Science, University of California, San Diego. ICS Report 8604. 1986. pp. 40.

57. Jordan M.I. Serial Order: a Parallel Distributed Processing Approach. Advances in Psychology. 1997. vol. 121. pp. 471–495. DOI: 10.1016/s0166-4115(97)80111-2.

58. Hochreiter S. Untersuchungen zu Dynamischen Neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer. Technische Universität München, 1991.

59. Hochreiter S., Bengio Y., Frasconi P., et al. Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies. A Field Guide to Dynamical Recurrent Neural Networks. Wiley-IEEE Press. 2001. pp. 237–243. DOI: 10.1109/9780470544037.ch14.

60. Bengio Y., Simard P., Frasconi P. Learning Long-Term Dependencies with Gradient Descent is Difficult. IEEE Transactions on Neural Networks. 1994. vol. 5, no. 2. pp. 157–166. DOI: 10.1109/72.279181.

61. Tiňo P., Hammer B. Architectural Bias in Recurrent Neural Networks: Fractal Analysis. Neural Computation. 2004. vol. 15, no. 8. pp. 1931–1957. DOI: 10.1162/08997660360675099.

62. Hochreiter S., Schmidhuber J. Bridging Long Time Lags by Weight Guessing and "Long Short-Term Memory". Spatiotemporal Models in Biological and Artificial Systems. 1996. vol. 37. pp. 65–72.

63. Schmidhuber J., Wierstra D., Gagliolo M., et al. Training Recurrent Networks by Evolino.. Neural Computation. 2007. vol. 19, no. 3. pp. 757–779. DOI: 10.1162/neco.2007.19.3.757.

64. Levin L.A. Universal Sequential Search Problems. Problems of Information Transmission. 1997. vol. 9, no. 3. pp. 265–266.

65. Schmidhuber J. Discovering Neural Nets with Low Kolmogorov Complexity and High Generalization Capability. Neural Networks. 1997. vol. 10, no. 5. pp. 857–873. DOI: 10.1016/s0893-6080(96)00127-x.

66. Møller, M.F. Exact Calculation of the Product of the Hessian Matrix of Feed-Forward Network Error Functions and a Vector in O(N) Time. Computer Science Department, Aarhus University, Denmark. 1993. no. PB-432. DOI: 10.7146/dpb.v22i432.6748.

67. Pearlmutter B.A. Fast Exact Multiplication by the Hessian. Neural Computation. 1994. vol. 6, no. 1. pp. 147–160. DOI: 10.1162/neco.1994.6.1.147.

68. Schraudolph N.N. Fast Curvature Matrix-Vector Products for Second-Order Gradient Descent. Neural Computation. 2002. vol. 14, no. 7. pp. 1723–1738. DOI: 10.1162/08997660260028683.

69. Martens J. Deep Learning via Hessian-Free Optimization. Proceedings of the 27th International Conference on Machine Learning (ICML-10) (Haifa, Israel, June 21–24, 2010), 2010. pp. 735–742.

70. Martens J., Sutskever I. Learning Recurrent Neural Networks with Hessian-Free Optimization. Proceedings of the 28th International Conference on Machine Learning (ICML-11) (Bellevue, Washington, USA, June 28 – July 02, 2011), 2011. pp. 1033–1040.

71. Schmidhuber J. Learning Complex, Extended Sequences Using the Principle of History Compression. Neural Computation. 1992. vol. 4, no. 2. pp. 234–242. DOI: 10.1162/neco.1992.4.2.234.

72. Connor J., Martin D.R., Atlas L.E. Recurrent Neural Networks and Robust Time Series Prediction. IEEE Transactions on Neural Networks. 1994. vol. 5, no. 2. pp. 240–254. DOI: 10.1109/72.279188.

73. Dorffner G. Neural Networks for Time Series Processing. Neural Network World. 1996. vol. 6. pp. 447–468.

74. Schmidhuber J., Mozer M.C., Prelinger D. Continuous History Compression. Proceedings of International Workshop on Neural Networks (Aachen, Germany, 1993), 1993. pp. 87–95.

75. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997. vol. 9, no. 8. pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

76. Gers F.A., Schmidhuber J., Cummins F. Learning to Forget: Continual Prediction with LSTM. Neural Computation. 2000. vol. 12, no. 10. pp. 2451–2471. DOI: 10.1162/089976600300015015.

77. Pérez-Ortiz J.A., Gers F.A., Eck D., et al. Kalman Filters Improve LSTM Network Performance in Problems Unsolvable by Traditional Recurrent Nets. Neural Networks. 2003. vol. 16, no. 2. pp. 241–250. DOI: 10.1016/s0893-6080(02)00219-8.

78. Weng J., Ahuja N., Huang T.S. Cresceptron: a Self-Organizing Neural Network Which Grows Adaptively. International Joint Conference on Neural Networks (IJCNN) (Baltimore, MD, USA, 7–11 June 1992). 1992. vol. 1. pp. 576–581. DOI: 10.1109/ijcnn.1992.287150.

79. Weng J.J., Ahuja N., Huang T.S. Learning Recognition and Segmentation Using the Cresceptron. International Journal of Computer Vision. 1997. vol. 25, no. 2. pp. 109–143. DOI: 10.1023/a:1007967800668.

80. Ranzato M.A., Huang F.J., Boureau Y.L., et al. Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition. IEEE Conference on Computer Vision and Pattern Recognition (Minneapolis, MN, USA, 17–22 June 2007), 2007. pp. 1–8. DOI: 10.1109/cvpr.2007.383157.

81. Scherer D., Müller A., Behnke S. Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. Lecture Notes in Computer Science. 2010. vol. 6354, pp. 92–101. DOI: 10.1007/978-3-642-15825-4_10.

82. Smolensky P. Information Processing in Dynamical Systems: Foundations of Harmony Theory. Parallel Distributed Processing: Explorations in the Microstructure of Cognition. 1986. vol. 1. pp. 194–281.

83. Hinton G.E., Sejnowski T.E. Learning and Relearning in Boltzmann Machines. Parallel Distributed Processing. 1986. vol. 1. pp. 282–317.

84. Memisevic R., Hinton G.E. Learning to Represent Spatial Transformations with Factored Higher-Order Boltzmann Machines. Neural Computation. 2010. vol. 22, no. 6. pp. 1473–1492. DOI: 10.1162/neco.2010.01-09-953.

85. Mohamed A., Hinton G.E. Phone Recognition Using Restricted Boltzmann Machines. IEEE International Conference on Acoustics, Speech and Signal Processing (Dallas, TX, USA, 14–19 March 2010), 2010. pp. 4354–4357. DOI: 10.1109/icassp.2010.5495651.

86. Salakhutdinov R., Hinton G. Semantic Hashing. International Journal of Approximate Reasoning. 2009. vol. 50, no. 7. pp. 969–978. DOI: 10.1016/j.ijar.2008.11.006.

87. Bengio Y., Lamblin P., Popovici D., et al. Greedy Layer-Wise Training of Deep Networks. Advances in Neural Information Processing Systems 19. 2007. pp. 153–160.

88. Vincent P., Hugo L., Bengio Y., et al. Extracting and Composing Robust Features with Denoising Autoencoders. Proceedings of the 25th international Conference on Machine learning (Helsinki, Finland, July 05–09, 2008). 2008. pp. 1096–1103. DOI: 10.1145/1390156.1390294.

89. Erhan D., Bengio Y., Courville A., et al. Why Does Unsupervised Pre-Training Help Deep Learning?. Journal of Machine Learning Research. 2010. vol. 11. pp. 625–660.

90. Arel I., Rose D.C., Karnowski T.P. Deep Machine Learning – a New Frontier in Artificial Intelligence Research. Computational Intelligence Magazine, IEEE. 2010. vol. 5, no. 4. pp. 13–18. DOI: 10.1109/mci.2010.938364.

91. Viren J., Sebastian S. Natural Image Denoising with Convolutional Networks. Advances in Neural Information Processing Systems (NIPS) 21. 2009. pp. 769–776.

92. Razavian A.Sh., Azizpour H., Sullivan J., at al. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (Washington, DC, USA, June 23–28, 2014), 2014. pp. 512–519. DOI: 10.1109/cvprw.2014.131.

93. Ruochen W., Zhe X. A Pedestrian and Vehicle Rapid Identification Model Based on Convolutional Neural Network. Proceedings of the 7th International Conference on Internet Multimedia Computing and Service (ICIMCS '15) (Zhangjiajie, China, August 19–21, 2015), 2015. pp. 32:1–32:4. DOI: 10.1145/2808492.2808524.

94. Boominathan L., Kruthiventi S.S., Babu R.V. CrowdNet: A Deep Convolutional Network for Dense Crowd Counting. Proceedings of the 2016 ACM on Multimedia Conference (Amsterdam, The Netherlands, October 15–19, 2016), 2016. pp. 640–644. DOI: 10.1145/2964284.2967300.

95. Kinnikar A., Husain M., Meena S.M. Face Recognition Using Gabor Filter And Convolutional Neural Network. Proceedings of the International Conference on Informatics and Analytics (Pondicherry, India, August 25–26, 2016), 2016. pp. 113:1–113:4. DOI: 10.1145/2980258.2982104.

96. Hahnloser R.H.R., Sarpeshkar R., Mahowald M.A., et al. Digital Selection and Analogue Amplification Coexist in a Cortex-Inspired Silicon Circuit. Nature. 2000. vol. 405. pp. 947–951. DOI: 10.1038/35016072.

97. Hahnloser R.H.R., Seung H.S., Slotine J.J. Permitted and Forbidden Sets in Symmetric Threshold-Linear Networks. Neural Computation. 2003. vol. 15, no. 3. pp. 621–638. DOI: 10.1162/089976603321192103.

98. Glorot X., Bordes A., Bengio Y. Deep Sparse Rectifier Neural Networks. Journal of Machine Learning Research. 2011. vol. 15. pp. 315–323.

99. Glorot X., Bengio Y. Understanding the Difficulty of Training Deep Feedforward Neural Networks. Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10) (Sardinia, Italy, May 13–15, 2010). Society for Artificial Intelligence and Statistics. 2010. pp. 249–256.

100. He K., Zhang X., Ren Sh. et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV) (Santiago, Chile, December 7–13, 2015), 2015. pp. 1026–1034. DOI: 10.1109/ICCV.2015.123.

101. Ioffe S., Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. JMLR Workshop and Conference Proceedings. Proceedings of the 32nd International Conference on Machine Learning (Lille, France, July 06–11, 2015), 2015. vol. 37. pp. 448–456.

102. Szegedy C., Liu W, Jia Y. et al. Going Deeper with Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (Boston, MA, USA, June 7–12, 2015), 2015. pp. 1–9. DOI: 10.1109/CVPR.2015.7298594.

103. Szegedy C., Vanhoucke V., Ioffe S., et al. Rethinking the Inception Architecture for Computer Vision. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (Seattle, WA, USA, Jun 27–30, 2016), 2016. pp. 2818–2826. DOI: 10.1109/cvpr.2016.308.

104. Szegedy C., Ioffe S., Vanhoucke V., et al. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17) (San Francisco, California, USA, February 4–9, 2017), 2017. pp. 4278–4284.

105. Cho K., van Merrienboer B., Gulcehre C., et al. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Doha, Qatar, October 25–29, 2014), 2014. pp. 1724–1734. DOI: 10.3115/v1/d14-1179.

106. Cho K., van Merrienboer B., Bahdanau D., et al. On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (Doha, Qatar, October 25, 2014), 2014. pp. 103–111. DOI: 10.3115/v1/w14-4012.

107. Chung, J., Gulcehre, C., Cho, K., et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. NIPS 2014 Workshop on Deep Learning (Montreal, Canada, December 12, 2014), 2014. pp. 1–9.

108. He K., Sun J. Convolutional Neural Networks at Constrained Time Cost. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Boston, MA, USA, June 07–12, 2015), 2015. pp. 5353–5360. DOI: 10.1109/CVPR.2015.7299173.

109. Jia Y., Shelhamer E., Donahue J., et al. Caffe: Convolutional Architecture for Fast Feature Embedding. Proceedings of the 22nd ACM International Conference on Multimedia (Orlando, FL, USA, November 03–07, 2014), 2014. pp. 675–678. DOI: 10.1145/2647868.2654889.

110. Kruchinin D., Dolotov E., Kornyakov K. et al. Comparison of Deep Learning Libraries on the Problem of Handwritten Digit Classification. Analysis of Images, Social Networks and Texts. Communications in Computer and Information Science. 2015. vol. 542. pp. 399–411. DOI: 10.1007/978-3-319-26123-2_38.

111. Bahrampour S., Ramakrishnan N., Schott L., et al. Comparative Study of Deep Learning Software Frameworks. Available at: `https://arxiv.org/abs/1511.06435` (accessed: 02.07.2017).

112. Bergstra J., Breuleux O., Bastien F., et al. Theano: a CPU and GPU Math Expression Compiler. Proceedings of the Python for Scientific Computing Conference (SciPy) (Austin, TX, USA, June 28 – July 3, 2010), 2010. pp. 3–10.

113. Abadi M., Agarwal A. Barham P. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16) (Savannah, GA, USA, November, 2–4, 2016), 2016. pp. 265–283.

114. Collobert R., Kavukcuoglu K., Farabet C. Torch7: a Matlab-like Environment for Machine Learning. BigLearn, NIPS Workshop (Granada, Spain, December 12–17, 2011), 2011.

115. Seide F., Agarwal A. CNTK: Microsoft's Open-Source Deep-Learning Toolkit. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16) (San Francisco, California, USA, August 13–17, 2016), 2016. pp. 2135–2135. DOI: 10.1145/2939672.2945397.

116. Viebke A., Pllana S. The Potential of the Intel(r) Xeon Phi for Supervised Deep Learning. IEEE 17th International Conference on High Performance Computing and Communications (HPCC) (New York, USA, August 24–26, 2015), 2015. pp. 758–765. DOI: 10.1109/hpcc-css-icess.2015.45.

117. Chollet. F., et al. Keras. 2015. Available at: `https://github.com/fchollet/keras` (accessed: 02.07.2017).

118. PadlePadle: PArallel Distributed Deep LEarning. Available at: `http://www.paddlepaddle.org/` (accessed: 02.07.2017).

119. Chen T., Li M., Li Y. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. Available at: `https://arxiv.org/abs/1512.01274` (accessed: 02.07.2017).

120. Intel Nervana Reference Deep Learning Framework Committed to Best Performance on all Hardware. Available at: `https://www.intelnervana.com/neon/` (accessed: 02.07.2017).

121. Shi Sh., Wang Q., Xu P. Benchmarking State-of-the-Art Deep Learning Software Tools. Available at: `https://arxiv.org/abs/1608.07249` (accessed: 02.07.2017).

122. Weiss K., Khoshgoftaar T.M., Wang D. A Survey of Transfer Learning. Journal of Big Data. 2016. vol. 3, no. 1. pp. 1–9. DOI: 10.1186/s40537-016-0043-6.

123. Ba J., Mnih V., Kavukcuoglu K. Multiple Object Recognition with Visual Attention. Proceedings of the International Conference on Learning Representations (ICLR) (San Diego, USA, May 7–9, 2015), 2015. pp. 1–10.

124. Graves A., Mohamed A.R., Hinton G. Speech Recognition with Deep Recurrent Neural Networks. IEEE International Conference on Acoustics, Speech and Signal Processing (Vancouver, Canada, May 26–31, 2013), 2013. pp. 6645–6649. DOI: 10.1109/ICASSP.2013.6638947.