УДК: 004.852 PACS: 07.05.Mh

ГЛУБОКИЕ НЕЙРОННЫЕ СЕТИ. ЗАРОЖДЕНИЕ. СТАНОВЛЕНИЕ. НАСТОЯЩЕЕ.

Макаренко А.В. [1, 2]

[1] Научно-исследовательская группа «Конструктивная Кибернетика»
101000, а/я 560, Москва, Россия

[2] Институт проблем управления им. В.А. Трапезникова РАН
117997, ул. Профсоюзная, д. 65, Москва, Россия

e-mail: avm.science@mail.ru

В обзоре рассмотрено эволюционное развитие искусственных нейронных сетей: от зарождения в виде нейрона Маккаллока-Питтса до современных глубоких архитектур. Перечислены основные «нейросетевые кризисы» и показаны причины их появления. Основное внимание в статье уделено нейронным архитектурам, обучающимся в режиме «обучения с учителем» по размеченной выборке данных. Приведены ссылки на оригинальные работы и основополагающие математические теоремы, формирующие теоретический фундамент под направлением искусственных нейронных сетей. Проанализированы причины затруднений на пути к формированию эффективных глубоких нейронных архитектур, рассмотрены пути разрешения возникших трудностей, выделены обстоятельства, способствующие успеху. В работе перечислены основные слои свёрточных и рекуррентных нейронных сетей, а также их архитектурные комбинации. В обзоре приведены примеры и ссылки на статьи, демонстрирующие эффективность глубоких нейронных сетей не только на данных, имеющих ярко выраженные структурные паттерны (изображения, голос, музыка и т.п.), но и на сигналах стохастического/хаотического характера. Выделено также одно из основных направлений развития свёрточных нейросетей – внедрение в слои обучаемых интегральных преобразований. На базовом уровне рассмотрена современная архитектура «Трансформер» – мейнстрим в задачах обработки последовательностей (в том числе в компьютерной лингвистике). Приведена ключевая проблематика современной теории искусственных нейронных сетей.

Keywords: глубокое обучение, свёрточные нейронные сети, рекуррентные нейронные сети.

Введение

Вычислительный интеллект как одна из ветвей *искусственного интеллекта* опирается на эвристические алгоритмы; в качестве основного математического инструментария применяется *машинное обучение по прецедентам*. Оно основано на выявлении общих закономерностей по частным эмпирическим (экспериментальным) данным и по факту относится к классу *индуктивного обучения*. Формально, задача машинного обучения ставится в следующем общем виде.

Дано:

X – описания объектов (характеристики, признаки, features);

R – решения алгоритма (ответы, метки, patterns, labels).

Существует, но неизвестна, целевая функция (target function):

$$G' : X \to R .$$

На основе анализа набора логических пар

$$\mathbf{d}_n^* = \left(\mathbf{x}_n, \mathbf{r}_n \right),$$

где $\mathbf{d}_n^*$ – составляет *n*-й прецедент, необходимо найти алгоритм (решающую функцию, decision function):

$$G : X \to R,$$

которая восстанавливает оценку G'.

Минимально выделяют два подмножества прецедентов – *обучающую* и *тестовую выборки* соответственно:

$$\mathrm{D}^{\mathrm{Tr}} = \left\{\mathbf{d}_n^*\right\}_{n=1}^{N_{\mathrm{Tr}}} - \textit{train set}, \quad \mathrm{D}^{\mathrm{Ts}} = \left\{\mathbf{d}_n^*\right\}_{n=1}^{N_{\mathrm{Ts}}} - \textit{test set}.$$

Отметим важное требование – исключение «*протечек данных*» (*leaked data*):

$$\mathrm{D}^{\mathrm{Tr}} \cap \mathrm{D}^{\mathrm{Ts}} \equiv \varnothing .$$

Введём в рассмотрение алгоритм

$$\mathrm{G}_i : \mathrm{X} \times \mathrm{W}_i \to \mathrm{R},$$

где W_i – множество допустимых значений $\mathbf{w}$ – вектора параметров алгоритма.

В этом случае выделяют два основных типа обучения:

параметрический – при фиксированном G_i ищется «оптимальное» значение $\tilde{\mathbf{w}}$, доставляющее минимум функционала ошибки

$$\mathrm{L}\left[\mathrm{G}_i(\mathrm{X}, \tilde{\mathbf{w}}), \mathrm{R}\right] \to \min .$$

структурный – в этом случае вначале осуществляется поиск «оптимального» представления G_i, а затем, «оптимальное» значение $\tilde{\mathbf{w}}$.

Выделяют три основных типа (режима) обучения:

с учителем – $\mathrm{D}^{\mathrm{Tr}} \neq \varnothing$ (это режим обучения по размеченной выборке);

без учителя – $\mathrm{D}^{\mathrm{Tr}} \equiv \varnothing$ (в этом режиме, как правило, решаются задачи кластеризации или понижения размерности набора данных X);

с подкреплением (*reinforcement learning*) – осуществляется поисковое взаимодействие обучаемого агента с внешней средой, обучение управляется системой поощрений и штрафов [1].

Из вышеприведённой формулировки задачи фактически следует, что алгоритм G_i, в зависимости от постановки задачи, может решать различные задачи из области управления: оценивания и прогнозирования процессов, идентификации систем и собственно управления.

Из числа подходов машинного обучения выделяют обширный класс методов *глубокого обучения* (*Deep Learning*), которые моделируют иерархические абстракции в данных, используя архитектуры, состоящие из каскадного множества нелинейных преобразований (фильтров). Пример иерархических абстракций в данных (распознавание изображений) приведён на рисунке 1а: пунктирная стрелка означает, что мета-признаки того или иного изображения помимо композиции (сцены) включают в себя также и все нижележащие (простые) иерархии, как-то: градиенты яркости, прямые линии, углы, контуры, текстуры.

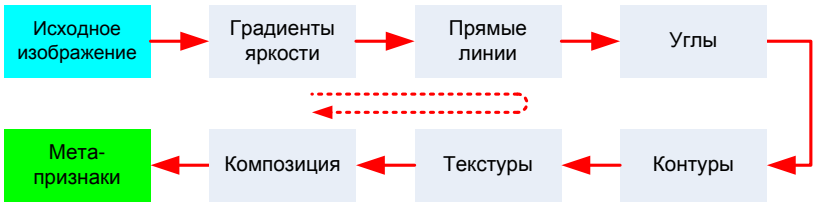

Рис. 1а. Иерархические абстракции в данных.

Архитектура, состоящая из каскадного множества нелинейных преобразований (фильтров), в общем виде показана на рисунке 1б.

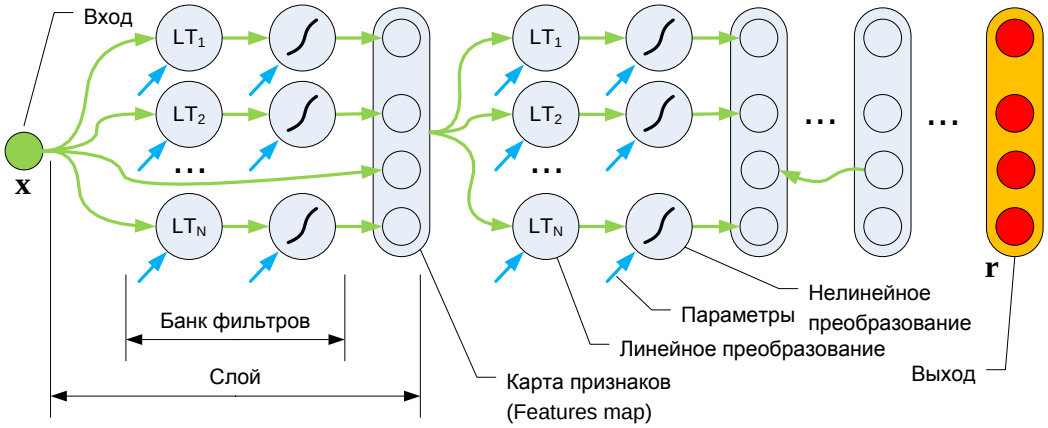

Рис. 1б. Обобщённая архитектура алгоритма глубокого обучения на основе каскадного множества нелинейных преобразований.

Уникальная особенность глубокого обучения заключается в том, что соответствующие алгоритмы работают с исходными данными (низкоуровневыми признаками) и самостоятельно извлекают (формируют) высокоуровневое признаковое описание объектов. То есть речь идёт о *метаобучении* – компьютерная программа самостоятельно учится, как лучше ей учиться. Сравнительные отличия с классическими *статистическими методами* и «плоским» машинным обучением даёт следующая диаграмма:

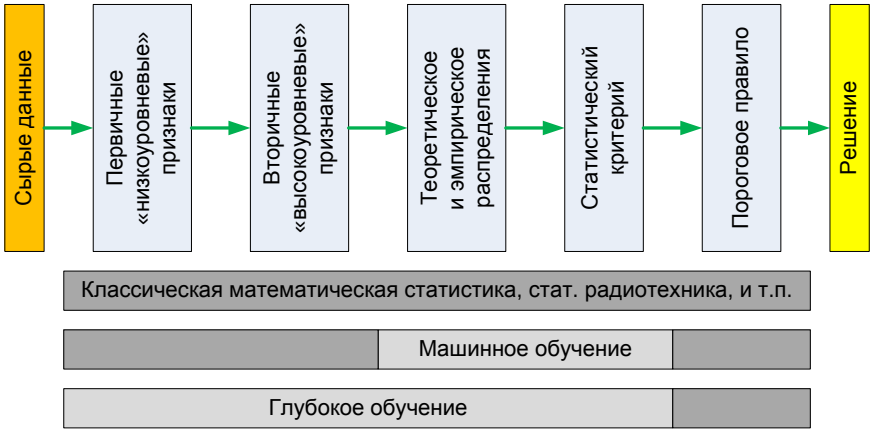

Рис. 2. Кодировка операций: тёмно-серый – выполняется человеком; светло-серый – автоматически, в рамках обучения модели.

Мейнстримом технической реализации концепции «Глубокое обучение» в настоящий момент являются глубокие *искусственные нейронные сети* (ИНС) [2]. Важно понимать, что глубокое обучение существенно шире по своей сути, нежели ИНС, и включает также исследования по *глубоким случайным лесам* (Deep Random Forest), по *глубоким байесовым сетям* (Deep Bayesian Networks) и некоторым другим подходам (в том числе и по исторически обусловленному *логическому интеллекту*).

Одна из особенностей глубоких нейронных сетей заключается в возможности реализации ими существенно адаптивного (в какой-то мере даже сверхадаптивного) управления.

Как будет показано далее, весь объём достижений глубоких нейросетей, с которыми читатель, возможно, сталкивается в повседневной жизни (распознавание номеров автомобилей, перевод текстов, распознавание слитной речи, синтез голоса и т.п.), объясняется хитроумными комбинациям всего трёх типов слоёв искусственных нейронов.

Одной из целей настоящего обзора является развенчивание широко распространённого мифа из мира ИНС: нейросети – это какая-то «магия» и сплошная «кустарщина», и «наука не понимает, как это всё работает». Также будет показано, что ошибочным является стереотип, что глубокие нейронные сети эффективно функционируют только на данных, имеющих ярко выраженные структурные паттерны (изображения, голос, музыка и т.п.) и не работают со случайными и/или хаотическими процессами.

Для формирования у читателя цельной картины эволюции классических нейросетей в глубокие, ниже приводится краткая хронология основных событий: с момента зарождения этого научного направления и до настоящего времени.

1. Эволюция полносвязных нейросетей прямого распространения

Официально старт нейросетевому направлению работ был дан в 1943 г. в статье У. Маккалока и У. Питтса [3]. Авторы ввели понятие *искусственной нейронной сети* (ИНС) и предложили формальную модель *искусственного нейрона*:

$$s = \mathbf{w} \cdot \mathbf{x} + b\,,\ z = \mathrm{g}(s)\,, \tag{1}$$

где: $\mathbf{x}$ – вектор *входных данных*, $\mathbf{x} \in \mathbb{Z}_{\{0,1\}}^{N}$; $\mathbf{w}$ – вектор *весов*; b – смещение; $\cdot$ – операция скалярного умножения; $\mathrm{g}(\circ)$ – *функция активации*; z – *выход*. Следует отметить, что исходно нейрон оперировал только двухуровневыми сигналами: $x_i = 0$ – логический нуль и $x_i = 1$ – логическая единица, а функция активации строилась по типу пороговой функции Хевисайда:

$$z = \mathrm{g}_{01}(s) = \begin{cases} 1, & \text{если } s > a, \\ 0 & \text{если } s \le a. \end{cases}, \tag{2}$$

где $a > 0$ – *порог активации*.

При формировании ИНС отдельные нейроны (1) объединяются в *нейросетевой слой*:

$$\mathbf{s} = \mathbf{W}\mathbf{x} + \mathbf{b}\,,\ \mathbf{z} = \mathrm{g}_{01}(\mathbf{s})\,, \tag{3}$$

где $\mathbf{W}$ – матрица весов, в общем случае прямоугольная.

В 1949 г. Д. Хебб в книге [4] изложил некоторые гипотезы относительно того, как нейроны человеческого мозга могут обучаться. Одна из основных концепций: обучение происходит в результате усиления связи (*синаптического веса*) между одновременно активными нейронами[1]. Исходя из этого часто используемые связи усиливаются, что объясняет феномен обучения путём многократного повторения одних и тех же входных стимулов (см. также обучение с подкреплением [1]).

В 1958 г. Ф. Розенблатт изобретает *перцептрон* с одним *скрытым слоем* [5]:

$$\mathbf{s}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1\,,\ \mathbf{y} = \mathrm{g}_{01}(\mathbf{s}_1)\,,\ \mathbf{s}_2 = \mathbf{W}_2\mathbf{y} + \mathbf{b}_2\,,\ \mathbf{z} = \mathrm{g}_{11}(\mathbf{s}_2)\,, \tag{4}$$

где $z = \mathrm{g}_{11}(s) = \operatorname{sign} \mathrm{s}$, $\mathbf{x} \in \mathbb{Z}_{\{0,1\}}^{N}$.

Следует отметить, что это первая ИНС, которая умела решать задачу классификации и активно применялась на практике.

Особенностью данной сети является необучаемость скрытого слоя (в терминологии Ф. Розенблатта [5] он именуется A-слоем) – элементы $\mathbf{W}_1$ и $\mathbf{b}_1$ исходно принимают случайные фиксированные значения $\{-1, 0, 1\}$, также фиксируется и пороги a в функции g_{01}. Как было позже показано, смысл этого слоя заключается в приведении несепарабельной задачи (линейно неразделимой) к сепарабельной (линейно разделимой). Здесь отчасти работает

Теорема 1 (*Ковер*, 1965 г. [6])**.** *Нелинейное проецирование в пространство более высокой размерности заданного набора данных, не являющихся сепарабельными, повышает вероятность их линейной разделимости.*

[1] *Впоследствии этот «алгоритм» стал называться «правило Хебба».*

Второй слой (в терминологии Ф. Розенблатта [5] он именуется R-слоем) обучается по *методу коррекции ошибки* [7] – формализованному правилу Хебба – по выходу нейросети $\mathbf{z}$.

В 1960 г Б. Уидроу и М. Хофф для обучения однослойной сети вида (1) предложили так называемое *дельта-правило* [8] (метод обучения ИНС градиентным спуском по поверхности ошибки) и назвали получившуюся систему ADALINE[2]. Данная ИНС сразу же начала использоваться для решения задач *адаптивного управления*. С одной стороны, относительно перцептрона Розенблатта это был шаг назад (отсутствие скрытого слоя и невозможность решения несепарабельных задач). С другой – был применён новый метод обучения на основе минимизации функции стоимости (функционала потерь), который заложил основу для разработки более совершенных алгоритмов машинного обучения[3] и собственно алгоритмов обучения ИНС. Ключевым моментом разработанного дельта-правила является вычисление ошибки модели и формирование корректирующих обновлений весов не по дискретному выходу нейросети $\mathbf{z}$, а по непрерывнозначному выходу сумматора $\mathbf{s}$ на основе квадратичной *функции потерь*:

$$\mathrm{L}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N_{\mathrm{Tr}}}(\mathbf{r}_n - \mathbf{s}_n)^2 \to \min_{\mathbf{w}}, \tag{5}$$

где $\mathbf{r}_n$ – истинная величина n-го обучающего прецедента, N_{Tr} – размер обучающей выборки. В результате сеть ADALINE стало возможно обучать высокоэффективным *методом градиентного спуска*:

$$\Delta\mathbf{w} = -\eta\nabla\mathrm{L}(\mathbf{w}), \tag{6}$$

где $\Delta\mathbf{w}$ – обновление весов сети (1), $\nabla\mathrm{L}(\mathbf{w})$ – градиент функции потерь, η – темп обучения. Как можно заметить из (5), обновление весов вычисляется по всем прецедентам из обучающей выборки (вместо инкрементного обновления веса после каждого образца), поэтому такой подход получил название «пакетный» (*batch*) градиентный спуск.

В 1969 г. М. Минский и С. Паперт опубликовали книгу [9], в которой содержался целый ряд критических замечаний о функциональных ограничениях перцептронов Розенблатта, тем самым вызвав существенное снижение интереса к тематике ИНС[4]. Следует отметить, что анализ был сделан для так называемого *элементарного перцептрона*, но название книги и формулировка выводов вызвали у читателей ощущение, что проблемы касаются всего направления ИНС[5]. Началась первая «*нейросетевая зима*», переведя фокус исследований в искусственном интеллекте на символьно-логические системы.

В 1986 г. Д.Е. Румельхарт переоткрывает заново *многослойный персептрон* (multilayer perceptron, MLP) в виде [10]

$$\mathbf{s}_1 = \mathbf{W}_1\mathbf{x} + \mathbf{b}_1,\ \mathbf{y} = g_\circ(\mathbf{s}_1),\ \mathbf{s}_2 = \mathbf{W}_2\mathbf{y} + \mathbf{b}_2,\ \mathbf{z} = g_\circ(\mathbf{s}_2), \tag{7}$$

где $\mathrm{g}_\circ(s) = \mathrm{g}_{\mathrm{sg}}(s) = \dfrac{1}{1+e^{-s}}$ – либо сигмоидальная функция, либо $\mathrm{g}_\circ(s) = \mathrm{g}_{\mathrm{th}}(s) = \mathrm{th}\, s$ – гиперболический тангенс. При этом первый слой становится также обучаемым, а вход сети непрерывнозначным $\mathbf{x} \in \mathbb{R}^N$. Сеть, как целое, учится по *методу обратного распространения ошибки*, который был впервые описан в 1974 г. в работах А.И. Галушкина [11] и П. Вербоса [12] и существенно развит в последующих работах [13, 14].

Следует отметить, что Д.Е. Румельхарт при публикации своих результатов по какой-то причине исказил определение перцептрона Розенблатта, представив его как ИНС без

---

[2] *ADAptive LInear NEuron, адаптивный линейный нейрон.*

[3] *В том числе логистической регрессии, метода опорных векторов и целого семейства регрессионных моделей.*

[4] *Интересно, что М. Минский был сокурсником Ф. Розенблатта.*

[5] *Ради справедливости стоит отметить, что в 1987 г. авторы выпустили третье издание книги, где многие критические замечания были сняты.*

скрытого слоя, тем самым породив методическую ошибку[6], что перцептрон Розенблатта не способен решать ряд элементарных несепарабельных задач, например вычислять булеву функцию XOR (исключающее «или»).

Тем не менее работа [10] запустила вторую волну массового интереса к ИНС. В 1988 г. Брумхед и Лоу предложили *сеть радиально-базисных функций* (Radial Basis Function Network, RBF) [15]. Это MLP с одним скрытым слоем вида

$$y_m = \exp\left[-\beta\sum_{i=1}^{N}\left(x_i - r_{i\,m}\right)^2\right],\ m=\overline{1,M}\,, \tag{8}$$

где M – число нейронов скрытого слоя, $\mathbf{r}_m$ – так называемый центральный вектор m-го скрытого нейрона (обучаемый параметр). Выходной слой имеет линейную (тождественную) функцию активации.

В 1989 г. были получены два важных результата. Во-первых, доказана важная

Теорема 2 (**Universal Approximation Theorem FFNN** [16], George Cybenko, 1989)**.** *Искусственная нейронная сеть прямого распространения с одним скрытым слоем может аппроксимировать любую непрерывную функцию многих переменных с любой точностью, при условии, что сеть имеет в скрытом слое достаточное количество нейронов N, имеющих сигмоидальную функцию активации* g_{sg}.

Теорема 2 является в определённом смысле специализированным аналогом теоремы А.Н. Колмогорова и В.И. Арнольда о представимости непрерывных функций нескольких переменных суперпозицией непрерывных функций одной переменной и существенно дополнила теорему о сходимости перцептрона [7]. В этом ключе стоит также отметить близкую теорему Хехт – Нильсена [17].

Во-вторых, Дж. Бридли вводит в обиход машинного обучения функцию активации SoftMax [18]:

$$z_i = \frac{e^{s_i}}{\sum_{i=1}^{M} e^{s_i}},\ i=\overline{1,M}\,, \tag{9}$$

где M – число нейронов выходного слоя. Функция (9), в отличие от других «интуитивных» (но, как правило, некорректных) подходов позволила на строгом теоретическом уровне обоснования решать задачу многоклассовой классификации (в режиме «один из многих»). При обучении ИНС с выходным *SoftMax*-слоем, как правило, используется функция потерь в виде *кросс-энтропии*:

$$L = -\frac{1}{N_{\mathrm{Tr}}}\sum_{n=1}^{N_{\mathrm{Tr}}}\sum_{i=1}^{M}(\mathbf{z}_n^*)_i \ln(\mathbf{z}_n)_i\,, \tag{10}$$

где N_{Tr} – размер обучающей выборки, M – число нейронов в выходом слое (число классов в решаемой задаче), $\mathbf{z}_n^*$ – вектор меток, ассоциированный с n-м прецедентом.

В 1991 г. К. Хорник обобщает теорему 2 на случай произвольных нелинейных активационных функций [19]. Становится ясно, что универсальные аппроксимационные свойства ИНС – это в большей мере свойство сетевой структуры.

Тем не менее, несмотря на успехи, исследователи очень скоро «упираются» в существенную ограниченность MLP с одним скрытым слоем – удаётся решать лишь ограниченное число практически важных задач. Такие насущные проблемы, как распознавание изображений, голоса, обработка текста – остаются за гранью приложимости MLP. Попытки добавления числа скрытых слоёв не приносят успеха – сети не обучаются, одна из сильнейших проблем – затухание градиента (*Vanishing Gradients*

[6] *Искусственное понятие «однослойный перцептрон» стало во главу целого ряда недоразумений, вошло в ряд монографий и учебников, в том числе и современных.*

Problem) $\left|\nabla \mathrm{L}(\mathbf{w})\right| \to 0$ по мере продвижения обучающих сигналов ко входу сети. Как следствие, во второй половине 90-х начинается вторая затяжная «нейросетевая зима».

Продемонстрируем практическую ограниченность MLP при классификации случайных сигналов[7]. Рассмотрим

Пример 1. Введём в рассмотрение три класса стохастических сигналов $\mathbf{x}$, различающихся функциями одноточечной плотности вероятности: $\mathcal{N}$ – нормальное; $\mathcal{U}$ – равномерное; $\mathcal{E}$ – экспоненциальное. Причём все сигналы являются δ-коррелированными и независимыми. Поставим задачу синтеза ИНС по типу MLP для классификации входящих сигналов по принадлежности к одному из классов: $\mathcal{N}$, $\mathcal{U}$ или $\mathcal{E}$. При этом потребуем стандартизацию входящих сигналов: нулевое математическое ожидание и единичная дисперсия. Таким образом, классические энергетические обнаружители функционировать не будут, а проблема классификации сдвигается в область распознавания структурных характеристик случайных процессов. Для определённости положим длину каждого временного ряда в 1024 отсчёта. Поставленная задача, как показано в работе [20], успешно решается MLP, если на вход ИНС подаются высокоуровневые информативные признаки – в данном случае статистические моменты (на рисунке 3 приведена диаграмма рассеивания изучаемых сигналов в координатах третьего и четвёртого статистических моментов). Если же на вход нейросети подать «сырые» сигналы $\mathbf{x}$, то она полностью теряет способность к классификации сигналов: значение меры качества F_1 [2] не поднимается выше 0.417 [21].

Подобные результаты в своё время как раз и вызвали «вторую нейросетевую зиму» и ограничили применимость «плоских» нейросетей к целому ряду важных прикладных областей, как-то: сверхширокополосная радиолокация, гидроакустическая шумопеленгация, инструментальная медицинская диагностика, техническая диагностика и т.п.

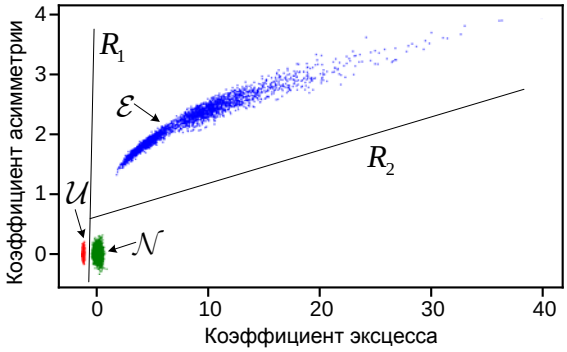

Рис. 3. Демонстрация линейной разделимости изучаемых сигналов x в пространстве третьего и четвёртого статистических моментов.

2. Зарождение «глубины» ИНС

Во второй половине 2000-х годов появляются работы, систематически направленные на разработку конструктивных методов обучения *многослойных нейронных сетей* (с числом скрытых слоёв более одного). В 2006 г. Дж. Хинтон с Р. Салахутдиновым предлагают *двухфазный подход* к обучению многослойных ИНС [22]. Первая фаза – послойное последовательное обучение *без учителя* скрытых слоёв (начиная с первого) внутренним представлениям[8]. На второй фазе выходной слой обучается и скрытые слои дообучаются посредством метода обратного распространения ошибок. Способ оказался работоспособным, но весьма затратным в плане вычислительных ресурсов, и, как оказалось в дальнейшем,

[7] *Задача по сути постановки близка к прикладной проблематике распознавания сигналов в пассивных акустических пеленгаторах.*

[8] *Фактически обучался энкодер в Автоэнкодере.*

весьма неустойчивым для сетей, имеющих более 3–5-ти скрытых слоёв. Похожая идея тех же авторов на основе ограниченной *машины Больцмана* [2] и *сетей доверия* [2] страдала теми же недостатками. Как выяснилось в дальнейшем, все эти ухищрения – существенно избыточны.

Оказалось, что для решения проблемы обучения глубоких нейронных сетей как единого целого (обучение всех слоёв сразу) необходимо было сделать два «простых шага».

Во-первых, потребовалось найти адекватную функцию активации, что и сделали Дж. Хинтон с соавтором, предложив в 2010 г. функцию ReLU (Rectified Linear Unit) [23][9]:

$$g_{RL}(s) = \max(0, s).$$

График этой функции приведён на рисунке 4. Для сравнения на рисунке 5 приведен график классической функции активации g_{sg}.

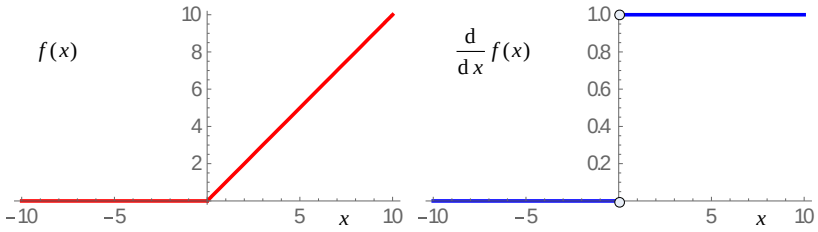

Рис. 4. График функции ReLU и её первой производной.

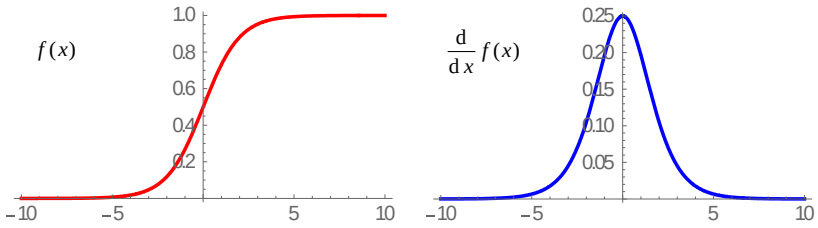

Рис. 5. График сигмоидальной функции и её первой производной.

Из сравнения графиков функций g_{sg} и g_{RL} видно, что ReLU имеет широкий рабочий отрезок (область, в которой первая производная существенно отлична от нуля). Кроме того, ReLU очень «дёшева» в вычислительном плане. Её недифференцируемость в нуле, как показала практика, никак себя негативно не проявила.

Во-вторых, потребовалось изменить схему *начальной инициализации весов* ИНС. Удачная конструкция получилась в том же 2010 г. у З. Глорота с соавтором [24]. Дисперсию инициализирующего шума (равномерного или нормального) было предложено находить по формуле

$$\mathrm{Var}(\mathbf{w}) = \frac{2}{N_{\text{in}} + N_{\text{out}}}, \tag{11}$$

где N_{in}, N_{out} – количества нейронов в предыдущем и последующем слоях соответственно.

К этому моменту уже более-менее выкристаллизовалось определение глубоких ИНС. К ним формально стали относить нейронные сети с числом скрытых слоёв более одного (точнее более двух – именно такие сети стали успешно извлекать сложные иерархические представления из «сырых» данных) и обучающихся как единое целое (обучение всех слоёв сразу).

Таким образом, к концу первой декады XXI столетия всё было готово для того, чтобы глубокие нейронные сети продемонстрировали прорывной результат. И его появление не заставило себя долго ждать.

---

[9] *Впоследствии появилось целое семейство ReLU-подобных функций* [2].

3. Глубокие свёрточные нейросети

В 1989 г. Ян Лекун с соавторами публикует работу [25], в которой описывает реальное приложение ИНС к практической задаче по распознаванию рукописных цифр в почтовом индексе. В статье рассматривается новая архитектура ИНС на основе принципа *разделения весов* (Weight Sharing). В данной работе фактически обобщён и переосмыслен ранний опыт по разработке К. Фукушимой *неокогнитрона* [26] и формализованы идеи *коннекционизма* М. Мозера [27]. К 1998 г. идеи Яна Лекуна окончательно вышлифовываются [28] в так называемые *свёрточные нейросети*. Представленная в работе [28] архитектура сети LeNet-5 стала фундаментальной на многие последующие годы, особенно для задач анализа изображений. Свёрточная нейросеть использовала последовательную комбинацию из двух типов слоёв[10]. Первый тип – *свёртка* (*Convolution Layer* [2]) – извлекает информативные признаки, имеющие структурную организацию, см. рисунок 4.36а. Второй тип – *субдескритизация* (*Pooling Layer* [2]) – за счёт пространственного сжатия данных обеспечивает инвариантность отклика слоя к малому смещению паттерна, см. рисунок 6б.

На выходе сети LeNet-5 применялся RBF слой (8), а в качестве функции потерь при обучении использовалась MSE (5).

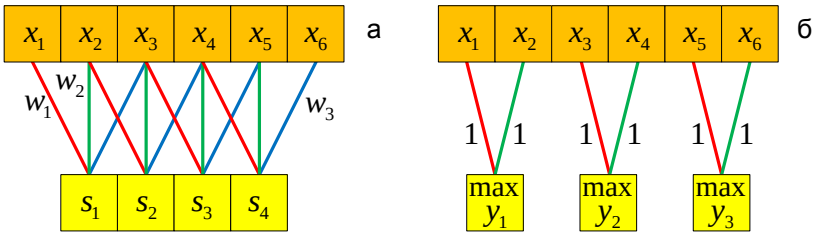

Рис. 6. Базовые слои свёрточной нейросети:

а) – 1D-свёртка, размер 3, шаг 1; б) – слой MaxPool размер 2, шаг 2.

В связи с отсутствием достаточных по качеству и размеру наборов данных, а также из-за медленного обучения на центральном процессоре (CPU) с 1998 г. по 2010 г. свёрточные нейросети пребывали в состоянии некоторой инкубации.

В 2010 г. были получены два результата, которые впоследствии оказали весьма существенное влияние на всю область глубоких нейросетей.

Во-первых, Д. Кирешан и Й. Шмидхубер опубликовали одну из первых реализаций свёрточной нейросети на графическом ускорителе (GPU) [29]. Реализация содержала 9 скрытых слоёв и оба прохода – прямой (расчёт) и обратный (обучение).

Во-вторых, М. Цейлер с коллегами предложили новый нейросетевой слой, фактически обратный операции свёртки [30], и назвали его «*слой деконволюции*»[11] (*Deconvolutional Network layer*):

$$\sum_{m=1}^{M} z_m \oplus f_{m,c} = x_c, \tag{12}$$

где: x – входные данные (изображение); $\oplus$ – операция свёртки; z – выход слоя (карты признаков, количеством M); $f_{m,c}$ – ядра свёрток (обучаемые послойно, без учителя, см. раздел 2), уникальные для каждого c – цветового канала изображения и для каждой карты признаков m. При этом, если входное изображение имеет размер $N_x \times N_y$, а ядро $N_k \times N_k$, то выход слоя имеет размер: $(N_x + N_k - 1) \times (N_y + N_k - 1)$.

[10] *Реализация слоёв в сети LeNet-5 отличалась от принятой в настоящее время.*

[11] Название слоя не совсем верное, поэтому в дальнейшем оно было изменено (примерно с 2015 г, см. ниже).

В 2012 г.А. Крижевский[12] в соревновании по распознаванию изображений ImageNet[13] применил подход на основе глубоких нейронных сетей. Его свёрточная сеть AlexNet победила с существенным отрывом от лучших решений, основанных на классических техниках компьютерного зрения и машинного обучения [31]. Эта работа фактически дала исходный толчок к буму Deep Learning, который мы наблюдаем и в настоящее время.

Отметим, что на выходе сети AlexNet стоял уже привычный SoftMax (9), в качестве функции потерь при обучении использовалась кросс-энтропия (10), а основной функцией активации являлась ReLU.

В том же 2012 г. тот же Дж. Хинтон с коллегами ввёл в рассмотрение технику *Dropout* [32] для борьбы с *переобучением*: на каждой итерации обучения часть нейронов скрытого слоя вместе с их входящими и исходящими весами исключается, а после завершения итерации – возвращается. После окончания обучения все веса умножаются на нормализующий коэффициент. Как впоследствии было показано, эта процедура эквивалентна порождению экспоненциально большого ансамбля ИНС и усреднения (*ансамблирования*) их решений, что усиливает инвариантность сети к ошибкам в данных.

В 2013 г. М. Лин с коллегами публикуют работу «*Network In Network*» («*Сеть внутри сети*») [33]. Статья содержала две ключевые идеи, которые впоследствии существенно развились. Во-первых, было предложено между слоями свёрток вставлять многослойные персептроны, которые усиливали обобщающие свойства свёрточных слоёв (эта идея в 2014 г легла в основу архитектуры модулей Inception, см. ниже). Во-вторых, был предложен новый слой: *глобальная усредняющая субдескритизация* (*Global Average Pooling*, GAP). Его структура приведена на рисунке 7. Применение этого слоя позволило конструировать *полносвёрточные нейросети* (*Fully Convolutional Networks*) без полносвязных слоёв в концевой части сети (эта идея в будущем привела к разработке целого направления: нейросетей, инвариантных к размеру входных данных, и позволяющих решать задачи локализации объектов и/или сегментации изображений, см. ниже).

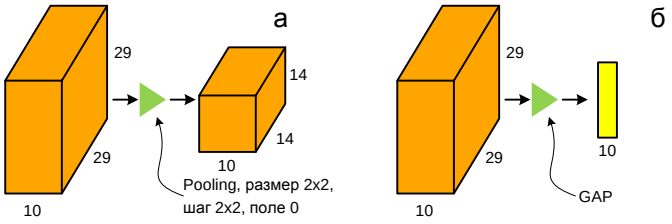

Рис. 7. Структуры слоёв субдескритизации: а – обычный AveragePooling; б – GAP.

В 2014 г. К. Симонян и А. Зиссерман публикуют так называемую VGG-сеть [34]. Она содержала 19 скрытых обучаемых слоёв. Её архитектура шла вразрез с рекомендациями авторов ранних сетей: использовать во входных слоях крупные свёртки размером не менее 5×5 пикселей (LeNet-5) и 11×11 пикселей (AlexNet). Оказалось, что последовательность мелких свёрточных ядер 3×3 эффективно эмулирует более крупные рецептивные поля (типа 9×9, 11×11) при явно меньшем числе настраиваемых параметров и с меньшим количеством затратных операций умножения.

Но оказалось, что 3×3 – это не предел. Осенью того же 2014 г. Кристиан Жегеди с коллегами публикует так называемую GoogLeNet [35], включающую в свой состав модули Inception (см. рисунок 8), в которых ключевую роль играют ядра размером 1×1. Эта работа во многом является творческим осмыслением ранее предложенного подхода Network-in-

[12] *Кстати, аспирант Дж. Хинтона.*

[13] *ImageNet – база данных аннотированных изображений, предназначенная для отработки и тестирования алгоритмов распознавания образов и машинного зрения. Для категоризации объектов на изображениях используется семантическая сеть WordNet. База данных определяет 1000 классов и по состоянию на 2016 год содержала около 10 млн изображений.*

Network [33], в котором применяются свёртки размером 1 × 1 (фактически пространственно ориентированные слои MLP) для увеличения комбинаторных свойств свёрточных слоёв.

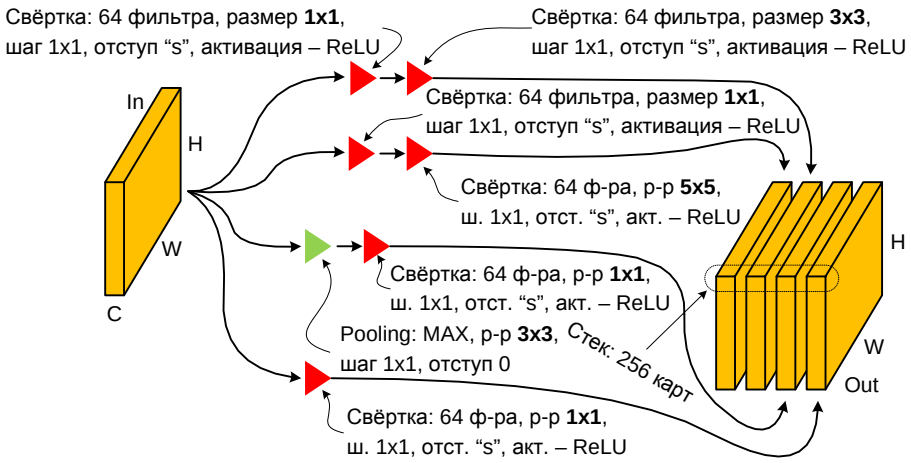

Рис. 8. Структура свёрточного модуля Inception.

Буквально одновременно с публикацией архитектуры Inception, Л. Сифре[14] защищает кандидатскую диссертацию [36], в которой вводит в рассмотрение так называемый *Depthwise Separable* свёрточный слой. Его архитектура приведена на рисунке 9б (в сравнении с архитектурой классического свёрточного слоя, изображённого на рисунке 9а).

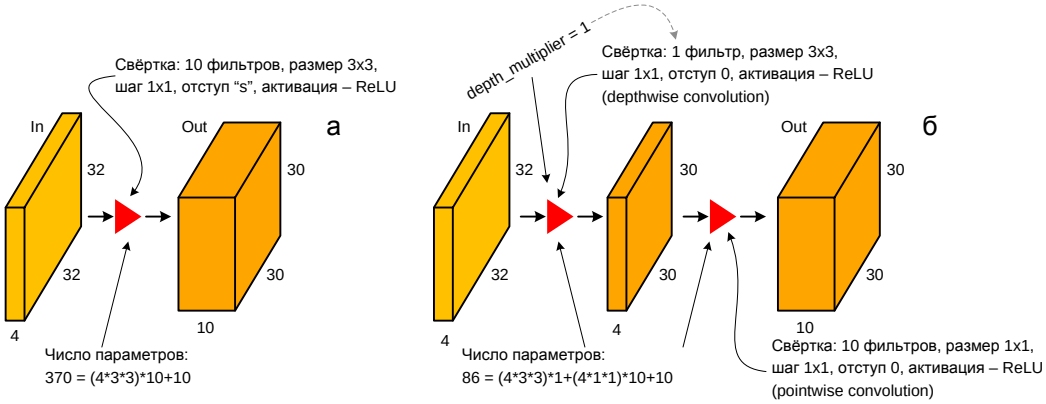

Рис. 9. Структуры свёрточных слоёв:

а – обычная регулярная свёртка; б – свёртка Depthwise Separable.

В ноябре 2014 г. Дж. Лонг с соавторами публикует препринт [37], в котором описывают так называемую «*полносвёрточную сеть*» (*Fully Convolutional Networks*), направленную на решение задачи семантической сегментации в режиме «pix2pix»[15]. Работа, с одной стороны, развила концепцию статьи [33] в части построения сетей без полносвязных слоёв, с другой, ввела в рассмотрение новый слой «*UpSampling*» – операция пространственного расширения карты признаков. Смысл операции иллюстрируют следующие формулы (в случае 2D данных, с размером ядра 2 и шагом 2):

[14] Кстати аспирант Стефана Маллата, который известен теоретическими исследованиями причин эффективности свёрточных нейронных сетей и анализом их эквивалентности банкам вейвлет-фильтров.

[15] Альтернативное название Image-to-Image, т.е. изображение на входе нейросети преобразуется на её выходе в некое другое изображение (зависит от задачи), но совпадающее по размеру с исходным.

$$\mathbf{Z} = \mathrm{UpS}(\mathbf{X}):\ \mathbf{X} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix},\ \mathbf{Z}' = \begin{bmatrix} x_{11} & x_{11} & x_{12} & x_{12} \\ x_{11} & x_{11} & x_{12} & x_{12} \\ x_{21} & x_{21} & x_{22} & x_{22} \\ x_{21} & x_{21} & x_{22} & x_{22} \end{bmatrix},\ \mathbf{Z} = \mathrm{Flt}(\mathbf{Z}'), \tag{13}$$

где: $\mathbf{X}$ – матрица входных данных; $\mathbf{Z}$ – выход слоя; Flt – операция фильтрации (она либо отсутствует – тождественное преобразование $\mathbf{Z} = \mathbf{Z}'$, либо применяется билинейная интерполяция, как в работе [37]). Следует отметить, что этот слой является упрощённой версией слоя деконволюции (12), который в современной трактовке называется «*транспонированная свёртка*» (T*ransposed convolution*) [38]. Весьма наглядно, разница между слоями UpSampling и Transposed convolution продемонстрирована в работе [39].

К концу 2014 г. происходит некий «фазовый переход»: интернет-гиганты, в том числе Google, признают высокую эффективность глубоких свёрточных ИНС в задачах, связанных с распознаванием изображений и голоса, и начинается активное их внедрение в соответствующие бизнес-процессы.

В 2015 г. С. Иоффе с коллегой предлагают *стандартизовывать* данные внутри нейросети при их передаче между скрытыми слоями [40]. Техника была названа *Batch Normalization*:

$$\begin{gathered} \mathcal{B} = \{x_1, x_2, \ldots, x_M\},\ y_i = \mathrm{BN}_{\gamma,\beta}(x_i),\ \mu_{\mathcal{B}} = \mathrm{M}[\mathcal{B}], \\ \sigma_{\mathcal{B}}^2 = \mathrm{D}[\mathcal{B}],\ \hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}} + \varepsilon},\ \varepsilon \to 0,\ \mathrm{BN}_{\gamma,\beta}: y_i = \gamma\,\hat{x}_i + \beta, \end{gathered} \tag{14}$$

где: $\mathcal{B}$ – мини-батч данных; γ, β – обучаемые параметры.

Предложенный подход существенно облегчил обучение глубоких структур, так как стандартизация приводила к тому, что последующий слой не тратил свои степени свободы на сдвиг и масштабирование входящих данных, а занимался только оцениванием их структурных свойств. Как следствие: ускорение сходимости процесса обучения, работа с более сложными данными, возможность использовать более высокие значения Learning Rate – параметр η в (6). Следует отметить, что в некоторых литературных источниках *Batch Normalization* объявляют более эффективной заменой *Dropout*, но на самом деле у них разный принцип действия и разное назначение.

Весной 2015 г. О. Роннебергер с коллегами предложил весьма оригинальную архитектуру полносвёрточной нейросети решающей задачу сегментации изображений [41]. Сеть состоит из двух частей: входной – сжимающей (свёрточные слои и слои субдискретизации) и выходной – расширяющей (в основе слои UpSampling, после каждого следует свёрточный слой). Ключевой момент – наличие прямых связей между сжимающей и расширяющей частями на одинаковых пространственных масштабах. Подобная архитектура, в отличии от, например, ранее рассмотренной [37] требует меньшее число примеров для обучения и при этом порождает более точную сегментацию.

Ещё одна разработка 2015 г. – *Разряженная свёртка* (*Dilation convolution*), которую предложили Ф. Юу и В. Котлин в работе [42]. Изменяя коэффициент дилатации D, возможно гибко управлять размером рецептивного поля без изменения числа обучаемых параметров (пример 1D свёртки размером 3):

$$\begin{aligned} &D = 0:\ w_0 x_0 + w_1 x_1 + w_2 x_2, \\ &D = 1:\ w_0 x_0 + w_1 x_2 + w_2 x_4, \\ &D = 2:\ w_0 x_0 + w_1 x_3 + w_2 x_6, \end{aligned} \tag{15}$$

где: $\mathbf{x}$ – входные данные; $\mathbf{w}$ – ядро свёртки. При $D = 0$ разряженная свёртка эквивалентна обычной. Следует отметить, что при комбинировании слоёв разряженных свёрток с $D > 0$, область видимости подобных нейронов растёт весьма быстро, при сохранении малого числа настраиваемых параметров.

В самом конце 2015 г. выходит весьма нетривиальная (и, как оказалось впоследствии, революционная) работа [43] сотрудников одного из исследовательских центров Microsoft. В статье описываются так называемые Residual Networks (ResNet: *свёрточная нейросеть с остаточными блоками*), см. рисунок 10. Основная идея ResNet: неизменённые входные данные суммируются с нелинейно преобразованными. Это сразу же привело к стабильной обучаемости сетей глубиной в 100, а в последствии и в 1000 слоёв[16] на достаточно сложных данных.

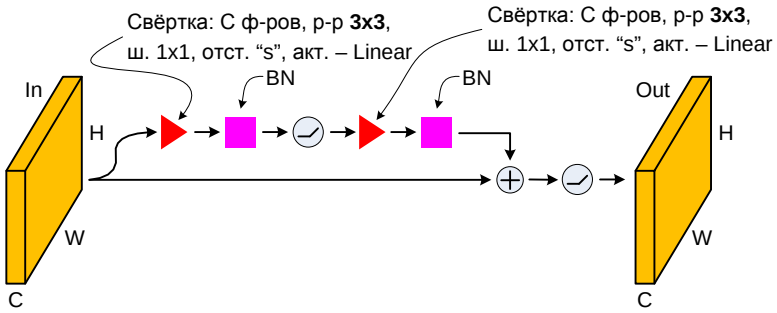

Рис. 10. Структура свёрточного модуля ResNet;

блоки BN – это Batch Normalization, см. (14).

На протяжении 2016–2019 гг. исследователи в основном экспериментировали с различными вариациями и комбинациями Inception и ResNet и их приложениями к реальным задачам. Но, помимо этого, были также инициированы исследования по использованию различного рода интегральных преобразований в свёрточных слоях.

В 2018 г. Х. Кхан с коллегами предложил [44] слой *вейвлет-деконволюции*[17] (*wavelet deconvolution*) в качестве эффективной адаптивной альтернативы предварительного спектрального разложения входных данных (временных рядов):

$$\psi_{s,b}(t)=\frac{1}{\sqrt{s}}\psi^{*}\left(\frac{t-b}{s}\right),\ z=x\oplus\psi_{s,b}, \tag{16}$$

где: x – входные данные (дискретная последовательность); $\oplus$ – операция свёртки; z – выход слоя; ψ^{*} – материнский вейвлет для непрерывного вейвлет-преобразования; b – параметр масштаба, адаптируемая (обучаемая) величина, которая изменяется при обучении на величину $\Delta s=-\eta\frac{\partial \mathrm{L}(s)}{\partial s}$.

В том же 2018 г. С. Фуджиеда с коллегами ввёл в рассмотрение [45] вейвлет аналог слоя субдискретизации (см. рисунок 6б), который позволяет эффективно извлекать информацию о характере текстур на 2D данных (изображениях) на разных пространственных масштабах посредством аналога дискретного вейвлет-преобразования. Недостатком реализованного авторами подхода является фиксированное число масштабов разложения, задаваемого посредством числа слоёв в нейросети.

В 2019 г. П. Лю с коллегами предложили встроить дискретное прямое и обратное вейвлет преобразование на основе вейвлета Хаара в свёрточные слои [46]. При этом архитектурно, полносвёрточная сеть получилась подобна рассмотренной выше сети U-Net [41].

Следует отметить, что подходы, подобные изложенным в работах [44, 45, 46], позволили существенно уменьшить число обучаемых параметров свёрточных нейросетей, а

[16] *Практического смысла в столь глубокой сети нет никакого* (*по крайней мере на исследовавшихся задачах*), *но есть смысл методический: «можем обучать».*

[17] Исходя из математического описания, более корректное название статьи и самой операции: вейвлет-декомпозиция (wavelet decomposition).

также улучшить их характеристики на ряде задач относительно типовых архитектур (AlexNet, VGG, и т.п.).

Продемонстрируем на задаче из примера 1 богатые функциональные возможности элементарной свёрточной нейронной сети при классификации случайных сигналов[18]. Рассмотрим

Пример 2. Сформируем ИНС с одним скрытым слоем из одного свёрточного ядра размером 1 (см. рисунок 11) и на вход подадим «сырые данные» **x**. Как показано в работе [21], подобная ИНС весьма успешно решает задачу классификации случайных сигналов имеющих идентичную энергию, и различающихся только функциями плотности вероятности: $\mathcal{N}$, $\mathcal{U}$, $\mathcal{E}$. Мера качества классификации $F_1 = 1$.

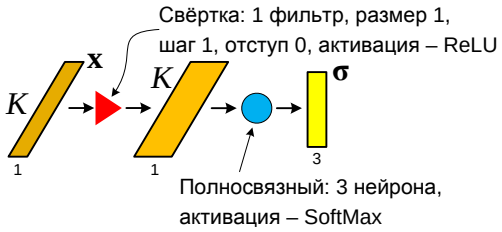

Рис. 11. Структура минимальной свёрточной ИНС, успешно решающей задачу классификации случайных сигналов с функциями плотности вероятности $\mathcal{N}$, $\mathcal{U}$, $\mathcal{E}$.

В работе [21] также проведён анализ структуры обученных свёрточных сетей и механизмов их функционирования при принятии решения в случае представленной задачи. Исследована их устойчивость к загрязнению входных данных по модели запирания канала/сенсора и возможность обнаружения сетью преобладающего сигнала в смеси сигналов $\mathcal{N}$, $\mathcal{U}$, $\mathcal{E}$ в условиях априорной неопределенности. Таким образом, показано, что глубокие свёрточные нейронные сети могут эффективно работать не только с сигналами, имеющими ярко выраженные паттерны, но и с реализациями узко- или широкополосных случайных процессов, и таким образом решать ряд задач по обработке сигналов. Решение подобной сложной прикладной задачи изложено в работе [47], посвящённой разработке на основе глубокой свёрточной ИНС первичного классификатора сигналов для квантовой волоконно-оптической системы охраны магистральных трубопроводов.

Следует отметить, что свёрточные нейронные сети отлично справляются и с проблематикой идентификации хаоса, а также прямого оценивания показателя Ляпунова в дискретных динамических системах по их наблюдаемым траекториям в расширенном пространстве состояний (см., например, работу [48]).

4. Глубокие рекуррентные нейросети

В 1990 г. Дж. Элман предложил *рекуррентную ИНС* с одним скрытым слоем по типу MLP [49]:

$$\mathbf{h}_k = \mathrm{g}_\mathrm{h}\left(\mathbf{W}_\mathrm{h}\,\mathbf{x}_k + \mathbf{U}_\mathrm{h}\,\mathbf{h}_{k-1} + \mathbf{b}_\mathrm{h}\right),\ \mathbf{y}_k = \mathrm{g}_\mathrm{y}\left(\mathbf{W}_\mathrm{y}\,\mathbf{h}_k + \mathbf{b}_\mathrm{y}\right), \tag{17}$$

где: k – дискретное время; $\mathbf{h}_k$ – вектор скрытого состояния сети в момент времени k. Как видно из (17), слагаемое $\mathbf{U}_\mathrm{h}\,\mathbf{h}_{k-1}$ задаёт обратную связь и отвечает за временной контекст. Этот контекст «одношаговый» по времени, подобные сети классифицируют как SimpleRNN (Recurrent Neural Network), в противовес им многошаговые сети называются FullyRNN.

[18] Следует отметить, что *задача по сути своей постановки близка к прикладной проблематике распознавания сигналов в пассивных акустических пеленгаторах.*

В 1991 г. Х. Зигельманн и Е. Сонтаг доказали следующий результат.

Теорема 3 (О полной тьюринговости RNN [50]**).** *Любые машины Тьюринга могут моделироваться полностью связанными рекуррентными сетями, созданными из нейронов с сигмоидальными функциями активации, при условии, что сеть имеет достаточное количество нейронов в скрытом слое* M *и достаточное число шагов временной памяти* K.

В 1992 г. К. Фунахаши и И. Накамура доказали следующий результат.

Теорема 4 (Универсальная аппроксимационная теорема RNN [51]**).** *Любая нелинейная динамическая система класса*

$$\frac{\mathrm{d}}{\mathrm{d}t}s(t) = \mathrm{f}(s),\ s(t=0) \in \mathrm{S}_0, \tag{18}$$

может быть аппроксимирована рекуррентной нейронной сетью с любой точностью, без ограничений на компактность пространства состояний системы, при условии, что сеть имеет достаточное количество нейронов в скрытом слое M.

Из теоремы 4 автоматически вытекает следствие, что любая непрерывная кривая (динамический процесс, временной ряд) может быть аппроксимирована с любой точностью выходом RNN (при соблюдении ряда условий, основными из которых являются достаточное количество нейронов рекуррентного слоя M и достаточное число шагов его временной памяти K). Таким образом, открываются возможности применения RNN для высокоэффективного решения ряда задач управления[19], в том числе оценивания и прогнозирования динамических сигналов, идентификации систем управления и т.п.; причём в классе адаптивных и сверхадаптивных систем управления.

В 1997 г. М. Джордан предложил модификацию сети Элмана (17) [52]:

$$\mathbf{h}_k = \mathrm{g}_\mathrm{h}\left(\mathbf{W}_\mathrm{h}\,\mathbf{x}_k + \mathbf{U}_\mathrm{h}\,\mathbf{y}_{k-1} + \mathbf{b}_\mathrm{h}\right),\ \mathbf{y}_k = \mathrm{g}_\mathrm{y}\left(\mathbf{W}_\mathrm{y}\,\mathbf{h}_k + \mathbf{b}_\mathrm{y}\right). \tag{19}$$

Из сравнения (17) и (19), видно, что в случае *сети Джордана* контекст решения определяется выходом сети, а не скрытым слоем.

Теоремы 3 и 4 вызвали активные исследования применимости сетей Элмана и Джордана в самых различных областях, но очень скоро выяснились фатальные недостатки SimpleRNN:

– фактически сети оперируют очень короткими динамическими контекстами, забывание «прошлого» идёт с экспоненциальной скоростью;

– в рамках одной сети очень сложно совмещать процессы различных масштабов, в том числе «быстрое» и «медленное» время, а также обрабатывать пропуски данных;

– рекуррентные сети, построенные по типу MLP, очень сложно обучать (используется алгоритм Backpropagation Through Time) при больших значениях K: градиент либо затухает, либо испытывает взрывной рост.

В 1997 г. для решения означенных проблем С. Хохрейтер с коллегами предложил принципиально иную архитектуру RNN, названную LSTM – Long Short-term Memory (*долгая краткосрочная память*) [53]. Элементарная ячейка скрытого слоя сети приведена на рисунке 12.

[19] *Естественно, что эти возможности относятся к глубоким RNN, в том числе имеющих в своём составе ячейки LSTM* (*описание см. ниже*).

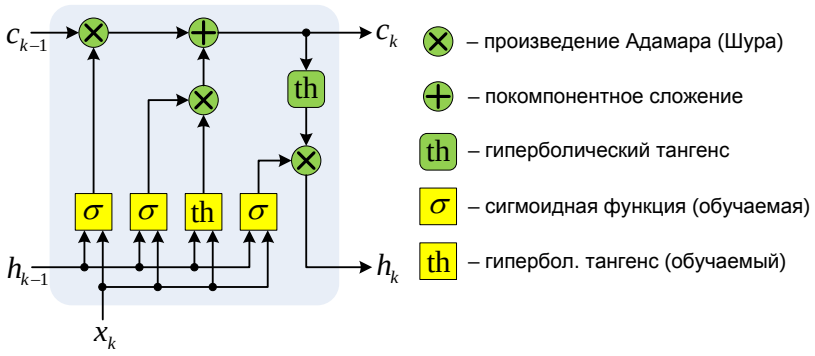

Рис. 12. *Элементарная ячейка LSTM.*

Ячейка LSTM содержит *конвейер состояния ячейки* $c_{k-1} \to c_k$, который включает в себя только линейные (!) операции. Модификация информации управляется вентилями (*gates*): $s' = s\sigma(\circ)$, причём $\sigma(\circ) \in [0,1]$. Стандартная ячейка LSTM состоит из четырёх вентилей:

«Forget gate»:

$$\mathbf{f}_k = \sigma\left(\mathbf{W}_\mathrm{f}\,\mathbf{x}_k + \mathbf{U}_\mathrm{f}\,\mathbf{h}_{k-1} + \mathbf{b}_\mathrm{f}\right). \tag{20a}$$

Интерпретация: если тема (сцена) изменяется, то информация о старой теме (сцене) стирается.

«Input gate & activation»:

$$\mathbf{i}_k = \sigma\left(\mathbf{W}_\mathrm{i}\,\mathbf{x}_k + \mathbf{U}_\mathrm{i}\,\mathbf{h}_{k-1} + \mathbf{b}_\mathrm{i}\right),\ \tilde{\mathbf{c}}_k = \mathrm{th}\left(\mathbf{W}_\mathrm{c}\,\mathbf{x}_k + \mathbf{U}_\mathrm{c}\,\mathbf{h}_{k-1} + \mathbf{b}_\mathrm{c}\right). \tag{20b}$$

Интерпретация: определяется какие значения будут обновляться и создаётся вектор кандидатов на $\tilde{\mathbf{c}}_k$, которые предполагается добавить в состояние ячейки.

«Internal state»:

$$\mathbf{c}_k = \mathbf{f}_k * \tilde{\mathbf{c}}_{k-1} + \mathbf{i}_k * \tilde{\mathbf{c}}_k\,. \tag{20c}$$

Интерпретация: формируется новое состояние ячейки $\mathbf{c}_k$.

«Output gate & value»:

$$\mathbf{o}_k = \sigma\left(\mathbf{W}_\mathrm{o}\,\mathbf{x}_k + \mathbf{U}_\mathrm{o}\,\mathbf{h}_{k-1} + \mathbf{b}_\mathrm{o}\right),\ \mathbf{h}_k = \mathbf{o}_k * \mathrm{th}\,\mathbf{c}_k\,. \tag{20d}$$

Интерпретация: формируется новый выход ячейки $\mathbf{h}_k$.

Количество настраиваемых во время обучения параметров в слое LSTM:

$$4\left(M\,N + M^2 + M\right),$$

где N – число признаков во входном векторе $\mathbf{x}$, M – число нейронов в рекуррентном слое, эту же размерность имеют вектора $\mathbf{c}$ и $\mathbf{h}$.

Последующие исследования LSTM-сетей показали, что для них выполняются теоремы 3 и 4, но при этом LSTM-сети свободны от большинства проблем SimpleRNN.

В 2000 г. Т. Чао и Х. Ли расширили теорему 4 на неавтономные нелинейные обыкновенные дифференциальные уравнения [54]:

$$\frac{\mathrm{d}}{\mathrm{d}\,t}s(t) = \mathrm{f}(s) + \mathrm{g}(t),\ s(t=0) \in \mathrm{S}_0\,. \tag{21}$$

В 2005 г. Ф. Морин и Б. Йошуа предложили иерархическое обобщение Softmax слоя (9) [55], что сделало возможным устойчивое решение задач классификации (в первую очередь в компьютерной лингвистике) размером свыше 20 тысяч классов.

В этом же году А. Гравес и Ю. Шмидхубер предлагают двунаправленное обобщение LSTM – BiDirectional LSTM [56]. Основная мотивация: «Настоящее зависит не только от прошлого, но и от будущего». Впоследствии это позволило получить более устойчивые и качественные решения ряда задач, так как для формирования выхода z_i сеть использовала

информацию не только из левой части временного ряда: […, z_{i-2}, z_{i-1}, z_i], но также и из правой: [z_{i+1}, z_{i+2}, …].

Таким образом, к концу 2005 г. у исследователей формируется уверенность в перспективности применения LSTM-сетей в области компьютерной лингвистики, распознавания и синтеза слитной речи, онлайн распознавания слитных рукописных текстов и т.п. Интенсивность исследований в области рекуррентных ИНС существенно возрастает.

В 2013 г. А. Гравес предлагает первую дифференцируемую реализацию[20] *механизма внимания* (*Attention Layer*) [57]. Структурная схема предложенного слоя сети приведена на рисунке 13.

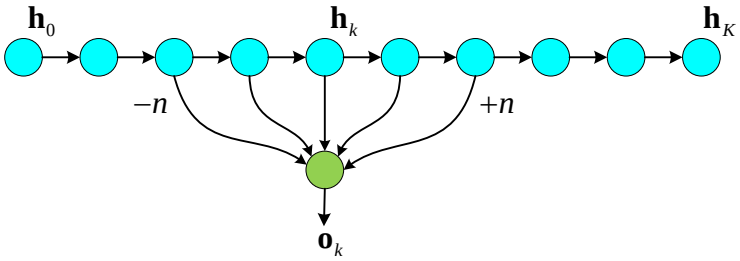

Рис. 13. Схема дифференцируемого слоя внимания.

Выход слоя внимания формируется как взвешенное скользящее среднее от выхода рекуррентного слоя:

$$o_k = \sum_{i=0}^{2n} a_i \, h_{k+(i-n)}, \quad \sum_{i=0}^{2n} a_i \equiv 1, \tag{22}$$

где **a** – вектор обучаемых параметров. Интерпретация (22): при формировании выхода слоя учитываются настраиваемые локальные во времени структурные связи – так называемый контекст.

Наконец, в 2015 г. выходит работа, которую давно ожидали: С. Ши с коллегами предлагает Convolutional LSTM-сеть [58], предназначенную для обработки *пространственно-временных зависимостей*. Обобщённая структура данной сети, демонстрирующая идею, приведена на рисунке 14. Основное отличие Convolutional LSTM-сети от обычной LSTM в том, что её внутренняя MLP-подобная структура (см. рисунок 13) заменена на свёрточную.

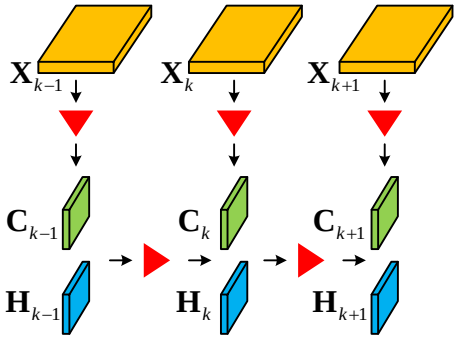

Рис. 14. Схема Convolutional LSTM-сети, треугольник – операция свёртки,

остальные обозначения аналогичны рис. 12.

В работе [58] продемонстрирована высокая эффективность ConvLSTM-сети при обработке данных с погодного радара.

[20] *Дифференцируемый здесь означает поддержку нейросетевым слоем режима обучения алгоритмом обратного распространения ошибки.*

Следом за работой [58] в том же году Н. Кальчбреннер с коллегами предлагает решётчатую LSTM-сеть [59]. Основная идея: расширение LSTM функционала с одной «временной оси» на все N осей входных данных, где $N = \dim \mathbf{X}$.

В конце 2015 г. выходят две работы [60, 61], которые демонстрируют противоречивые результаты применения техники Batch Normalization (14) к сетям RNN. Так в работе [60] применение пакетной нормализации фактически никак не повлияло на показатели качества исследуемых рекуррентных нейросетях, но в ряде случаев ускоряло их обучение. В работе [61], напротив, Batch Normalization являлся центральным элементом, применение которого позволило получить рекуррентную нейросеть высокого качества. Такое положение вещей практически моментально привело к появлению альтернативных решений.

Летом 2016 г. Дж. ЛеиБа с коллегами предлагают технику «*нормализация слоя*» (*Layer Normalization*) [62] для применения в RNN вместо пакетной нормализации. Предложенная альтернатива (нормализация слоя) стандартизует каждый сэмпл данных по всем нейронам слоя, в отличие от стандартизации мини-батча в целом, но для каждого нейрона индивидуально (пакетная нормализация). Как показали авторы [62], на их примерах (длинные последовательности и небольшие мини-батчи) Layer Normalization, относительно альтернатив, существенно положительно влиял на скорость обучения рекуррентных сетей.

Следует отметить, что результаты работ [60-62] относятся к задачам обработки текстов рекуррентными нейросетями, а эти задачи имеют одну примечательную особенность: предложения в наборах данных, как правило, имеют существенную вариативность по длине (числу слов, букв – *токенов*).

На протяжении 2016-2019 г.г. исследователи в основном экспериментировали с различными вариациями и комбинациями LSTM-сетей и их приложениями к реальным задачам. Но, помимо этого, активно исследовали и механизм внимания, как самостоятельную структурную единицу глубоких нейросетей.

Здесь следует заметить, что многие задачи компьютерной лингвистики (перевод, аннотирование текста, распознавание слитной речи и т.п.) в указанный период начали решаться в парадигме *sequence-to-sequence*, т.е. предложение (фрагмент текста) целиком поступает на вход нейросети (например, на английском языке), предложение (фрагмент текста) формируется на её выходе (к, примеру, с переводом на русский). Входная часть нейросети называется *Энкодер* (она обычно реализуется либо свёрточными, либо рекуррентными слоями), выходная часть – это *Декодер* (как правило, реализуется рекуррентными слоями). Между этими двумя частями включается слой внимания (22). Это была классическая – достаточно эффективная конструкция[21].

Но в 2017 г. А. Васвани с коллегами[22] предложили так называемую архитектуру *Transformer* [63] – целиком и полностью состоящую из слоёв иерархически организованных нелинейных банков ячеек внимания (!)[23], названных в оригинальной работе «*Multi-head attention*». Основная операция $\mathrm{Att}(\circ)$ – *attention* (*внимание*) выражается в виде:

$$\mathrm{Att}(Q, K, V) = \mathrm{Sm}\left(\frac{Q K^{\mathrm{T}}}{\sqrt{d}}\right) V, \tag{23}$$

где: $\mathrm{Sm}(\circ)$ – функции активации SoftMax (9); d – число столбцов Q, K, V (фактически размерность *эмбеддинга* (*embeddings*) – вложения); Q – запрос; K – ключ; V – значения эмбеддинга (как правило, векторное представление /кодирование/ обрабатываемого токена).

[21] Весьма схожая по архитектуре с автоэнкодерами [2].
[22] Все сотрудники различных подразделений Google.
[23] В состав сети дополнительно входят аналоги полносвязных слоёв, ResNet блоков и функции активации SoftMax, и, что примечательно, также Layer Normalization [62].

Предложенная конструкция существенно улучшила качество машинного перевода[24], став ведущей моделью компьютерной лингвистики. Её развитие и улучшение не заставили себя долго ждать, ибо базовая архитектура обладает существенным недостатком – ограниченная длина операционного контекста (не более нескольких десятков токенов).

Летом 2018 г. М. Дегани с коллегами обобщают архитектуру *Transformer*, включая в её состав рекуррентные цепочки [64]. Осенью того же года, Я. Девлин с коллегами предложили архитектуру BERT (Bidirectional Encoder Representations from Transformers) [65]. Существенное преимущество BERT перед LSTM сетями – это длина операционного контекста: десятки токенов у LSTM, против двух-трёх сотен у BERT.

Наконец, в 2019 г. З.Даи с коллегами предлагает новую архитектуру – так называемый Transformer-XL [66]. Длина генерируемых последовательностей согласованных токенов достигла нескольких тысяч[25].

Заключение

Итак, «естественный отбор» эффективных в прикладном плане архитектур ИНС привёл к тому, что к началу 2019 г. мейнстримом в глубоких нейросетях являются всего три качественно различающихся типа слоёв:

– *полносвязные* (по типу персептрона Румельхарта);

– *сверточные* (со всем многообразием их модификаций);

– *рекуррентные* (в основном LSTM и GRU [2]).

При этом полносвязные слои почти потеряли всякую самостоятельность. Они стоят, как правило, на выходе сети, представляя собой линейный классификатор (регрессор) сепарабельной задачи (см. теорему 1), которую формируют более выразительные (имеющие лучшие обобщающие свойства) свёрточные и рекуррентные слои.

Мощность (выразительность, обобщающая способность) свёрточных и рекуррентных слоёв по большей части объясняется тем, что они, в отличие от полносвязных архитектур, построенных по типу MLP, максимально задействуют при построении информативных признаков явную и/или скрытую структуру данных. Наглядный пример: распознавание изображений, голоса, построение языковых моделей. Примеры 1 и 2, приведённые выше, также демонстрируют этот аспект.

Следует отметить, что колоссальная «обобщающая способность» глубоких нейросетей формируется в основном за счёт широкой структурно-статистической вариативности и, как следствие, большого объёма обучающих данных. В этом вопросе, к сожалению, прогресс с 60-х годов XX века пока весьма слаб [7]. Поэтому такая процедура как *data augmentation* [2] является весьма востребованной в практике обучения глубоких ИНС.

Еще раз подчеркнем, что все приведённые выше в настоящем разделе теоремы формируют строгий математический фундамент теории ИНС – гарантируют решение задачи, но не являются конструктивными: они не дают путей решения задачи. И именно здесь начинается «*нейросетевое искусство*». Попытки формализовать этот процесс активно предпринимаются в рамках такой научной дисциплины как «Теория статистического обучения», ряд существенных результатов в которой был получен в своё время учёными ИПУ РАН В. Вапником и А. Червоненкинсом при разработке ими теории так называемой VC-размерности [67]. Правда, эти результаты к оцениванию характеристик глубоких нейронных сетей имеют очень ограниченную применимость.

[24] На паре «Английский → Немецкий», метрика качества BLEU превысила величину 28, что более чем на два с лишним пункта лучше предыдущего результата (сеть SliceNet /CNN/).

[25] Порождается весьма согласованный и достаточно осмысленный текст.

Из приведённого выше материала становится отчётливо видно, что успех глубоких ИНС объясняется не только алгоритмическими и математическими прорывами[26]. Есть ещё кое-что.

Во-первых, экспоненциальный рост вычислительных мощностей (наглядно демонстрируемый сравнением двух вариантов: top-1 суперкомпьютера начала XXI века и массивно-параллельного ускорителя (GPU) наших дней), обеспечил возможность приемлемой длительности обучения весьма сложных нейросетей.

ASCI White (X / 2000 г. – VI / 2002 г.)	GPU NVIDIA V100 (3-й квартал 2017 г.)
110 млн. $ США, 12.3 ТФЛОПС	1 700 $ США, 15.0 ТФЛОПС
6 МВт, 106 тонн	300 Вт, 370 г

Во-вторых, существенное снижение стоимости хранения цифровых данных и широкое распространение социальных сетей обеспечило экспоненциальный рост размеров датасетов различной природы (текст, фото, видео, звук, музыка и т.п.), на которых возможно тренировать нейросети.

В-третьих, с программно-алгоритмическим обеспечением в области Deep Learning сложилась уникальная ситуация, кардинально отличающаяся от принятых «правил игры» в других научно-технических областях: подавляющее большинство библиотек и фреймворков – бесплатно; исходный код основных библиотек и фреймворков – открыт; обучающие материалы – бесплатны и свободно доступны; функционируют широкие и отзывчивые группы поддержки – от уровня новичка и до топовой проблематики.

Следует заметить, что в данном обзоре, вне рассмотрения остался ряд важных вопросов, фактически формирующих ключевую проблематику современной теории искусственных нейронных сетей:

– два мощных и современных направления глубоких ИНС: *автоэнкодеры* – осуществляющие сжатие входных данных для представления их в *Latent-Space* (*скрытое пространство признаков и состояний*) и GAN – Generative Adversarial Network (генеративно-состязательные сети) – осуществляющие порождение данных посредством комбинации двух сетей: генератора и дискриминатора (осуществляет оценку правильности генерации). Заинтересованный читатель сможет найти о них подробную информацию в книге [2];

– обработка глубокими нейросетями нерегулярных сложных данных, к примеру, взвешенных графов произвольной структуры;

– обучение сетей, эффективные функции потерь и оптимизаторы (здесь можно порекомендовать читателю достаточно регулярно обновляемый пост С. Рудера [68]);

– машинный синтез эффективных нейроструктур, согласованных с решаемой задачей;

– интерпретация решений ИНС;

– сохранение эффективности нейросети вне домена обучающих данных и защита от имитационных атак (adversarial attack).

Список литературы

1. Sutton R.S.; Barto A.G. Reinforcement Learning: An Introduction. MIT Press, 1998.
2. Бенджио И., Гудфеллоу Я., Курвилль А. *Глубокое обучение*. ДМК-Пресс, 2018. – 652 с.
3. McCulloch W., Pitts W. *A Logical Calculus of the Ideas Immanent in Nervous Activity* // Bull. Math. Biophys. – 1943. – Vol. 5. – P. 115–133.
4. Hebb D. *The Organization of Behavior*. – John Wiley & Sons, New York, 1949. – 335 p.

[26] *Действительно, большинство теоретических результатов получено в конце XX века, а активный научный и прикладной рост области начался после 2012 г.*

5. Rosenblatt F. *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain* // Cornell Aeronautical Laboratory / Psychological Review. – 1958. – Vol. 65, No. 6. – P. 386–408.
6. Cover T. *Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recognition* // IEEE Transactions on Electronic Computers. – 1965. – EC-14. – P. 326–334.
7. Розенблатт Ф. *Принципы нейродинамики: Перцептроны и теория механизмов мозга*. – М.: Мир, 1965. – 478 с.
8. Widrow B. *Pattern Recognition and Adaptive Control* // Proc. of the IRE-AIEE Joint Automatic Control Conference. – August 1962. – P. 19–26.
9. Minsky M., Papert S. *Perceptrons: An Introduction to Computational Geometry*. – The MIT Press, Cambridge MA, 1969.
10. *Parallel Distributed Processing: Explorations in the Microstructures of Cognition* / Ed. by Rumelhart D.E., Mcclelland J.L. – Cambridge, MA: MIT Press, 1986.
11. Галушкин А. *Синтез многослойных систем распознавания образов*. – М.: Энергия, 1974. – 368 с.
12. Werbos P. *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. – PhD thesis, Harward University, 1974.
13. Барцев С., Охонин В. *Адаптивные сети обработки информации*. – 1986. – Препринт №59Б. – Красноярск: Ин-т физики СО АН СССР. – 20 с.
14. Rumelhart D., Hinton G., Williams R. *Learning Internal Representations by Error Propagation* // In: Parallel Distributed Processing. – Vol. 1. – P. 318-362. – Cambridge, MA, MIT Press, 1986.
15. Broomhead D., Lowe D. *Radial basis functions, multivariable functional interpolation and adaptive networks* // Technical report RSRE. – 1988. – 4148.
16. Cybenko G. *Approximation by Superpositions of a Sigmoidal function* // Mathematics of Control Signals and Systems. – 1989. – Vol. 2, No. 4. – P. 303–314.
17. Nielsen R. *Kolmogorov's Mapping Neural Network Existence Theorem* // Proc. of the IEEE First Int. Conference on Neural Networks (San Diego, CA). – 1987. – Vol. 3. – P. 11–13.
18. Bridle J. *Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition* // In: Neurocomputing. NATO ASI Series (Series F: Computer and Systems Sciences) / Soulie F.F., Herault J. (eds). – Vol. 68. – Springer, Berlin, Heidelberg, 1989. – 227-236 p.
19. Hornik K. *Approximation Capabilities of Multilayer Feedforward Networks* // Neural Networks. – 1991. – Vol. 4, No. 2. P. 251–257.
20. Tiwari S., Singh A., Shukla V. *Statistical moments based noise classification using feed forward back propagation neural network* // Int. J. of Computer Applications. – 2011. – Vol. 18, No. 2. – P. 36–40.
21. Portsev R., Makarenko A. *Convolutional neural networks for noise signal recognition* // IEEE 28th Int. Workshop on MLSP, Aalborg. – 2018. – P. 1–6.
22. Hinton G., Salakhutdinov R. *Reducing the Dimensionality of Data with Neural Networks* // Science. – 2006. – Vol. 313, No. 5786. – P. 504–507.
23. Nair V., Hinton G. *Rectified Linear Units Improve Restricted Boltzmann Machines* // Proc. of the Int. Conference on Machine Learning. – 2010. – P. 807–814.
24. Glorot X., Bengio Y. *Understanding the difficulty of training deep feedforward neural networks* // Proc. of the Thirteenth Int. Conference on Artificial Intelligence and Statistics. – 2010. – Vol. 9. – P. 249–256.
25. Lecun Y., Boser B., Denker J., Henderson D., Howard R., Hubbard W., Jackel L. *Backpropagation Applied to Handwritten Zip Code Recognition* // Neural Computation. – 1989. – Vol. 1, No. 4. – P. 541–551.
26. 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.
27. Mozer M. *Early parallel processing in reading: A connectionist ap-proach* // In: Attention and performance 12: The psychology of reading / M. Coltheart (Ed.) – 1987. – P. 83–104.
28. Lecun Y., Bottou L., Bengio Y., Haffner P. *Gradient-Based Learning Applied to Document Recognition* // IEEE Intelligent Signal Processing. – 1998. – P. 306–351.

29. Ciresan D., Meier U., Gambardella L., Schmidhuber J. *Deep Big Simple Neural Nets Excel on Handwritten Digit Recognition* // ArXiv: 1003.0358.
30. Zeiler M.D., Krishnan D., Taylor G.W., Fergus R. Deconvolutional networks // Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Francisco, CA, 2010. P. 2528-2535.
31. Krizhevsky A., Sutskever I., Hinton G. *ImageNet Classification with Deep Convolutional Neural Networks* // Proc. of the 25th Int. Conference NIPS. – 2012. – Vol. 1. – P. 1097–1105.
32. Hinton G. et al. *Improving neural networks by preventing co-adaptation of feature detectors* // ArXiv: 1207.0580.[27]
33. Lin M., Chen Q., Yan S. *Network In Network* // ArXiv: 1312.4400.
34. Simonyan K., Zisserman A. *Very Deep Convolutional Networks for Large-Scale Image Recognition* // ArXiv: 1409.1556.
35. Szegedy C., Liu W., Jia Y., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A. *Going Deeper with Convolutions* // ArXiv: 1409.4842.
36. Sifre L. Rigid-motion scattering for image classification // Ph.D. thesis. Ecole Polytechnique, CMAP. Defended October 6th, 2014.[28]
37. Long J., Shelhamer E., Darrell T. Fully Convolutional Networks for Semantic Segmentation // ArXiv: 1411.4038.
38. Dumoulin V., Visin F. A guide to convolution arithmetic for deep learning // ArXiv: 1603.07285. URL: https://github.com/vdumoulin/conv_arithmetic.
39. Odena A., et al. Deconvolution and Checkerboard Artifacts // Distill, 2016. DOI:10.23915/distill.00003.
40. Ioffe S., Szegedy C. *Batch Normalization: Accelerating Deep Net-work Training by Reducing Internal Covariate Shift* // ArXiv: 1502.03167.
41. Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation // ArXiv: 1505.04597.
42. Yu F., Koltun V. Multi-Scale Context Aggregation by Dilated Convolutions // ArXiv: 1511.07122.
43. He K., Zhang X., Ren S., Sun J. *Deep Residual Learning for Image Recognition* // ArXiv: 1512.03385.
44. Khan H, Yener B. Learning filter widths of spectral decompositions with wavelets // Proc. of the NIPS Conference. – 2018. – P. 4601 – 4612.
45. Fujieda S., Takayama K., Hachisuka T. *Wavelet Convolutional Neural Networks* // ArXiv: 1805.08620.
46. Liu P., Zhang H., Lian W., Zuo W. Multi-level Wavelet Convolutional Neural Networks // ArXiv: 1907.03128.
47. Makarenko A. *Deep Learning Algorithms for Signal Recognition in Long Perimeter Monitoring Distributed Fiber Optic Sensors* // IEEE 26th Int. Workshop on MLSP. – 2016. – Vietri sul Mare, IIASS. – P. 1–6.
48. Makarenko A.V. Deep learning algorithms for estimating Lyapunov exponents from observed time series in discrete dynamic systems // Proc. of the STAB Conference. – 2018. – P. 1–4.
49. Elman J. *Finding Structure in Time* // Cognitive Science. – 1990. – Vol. 14, No. 2. – P. 179–211.
50. Siegelmann H., Sontag E. *Turing computability with neural nets* // Appl. Math. Lett. – 1991. – Vol. 4, No. 6. – P. 77–80.
51. Funahashi K., Nakamura Y. *Approximation of dynamical systems by continuous time recurrent neural networks* // Neural Networks. – 1993. – Vol. 6, No. 6. – P. 801–806.
52. Jordan M. *Serial Order: A Parallel Distributed Processing Approach* // Advances in Psychology. – 1997. – No. 121. – P. 471–495.
53. Hochreiter S., Schmidhuber J. *Long-Short Term Memory* // Neural Computation. – 1997. – Vol. 9, No. 8. – P. 1735–1780.

[27] Дополнительно: Srivastava N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks fromOverfitting // Journal of Machine Learning Research 15 (2014) 1929-1958.
[28] Дополнительно: Sifre L., Mallat S. Rigid-Motion Scattering for Texture Classification // ArXiv: 1403.1687.

54. Chow T., Li X. *Modeling of continuous time dynamical systems with input by recurrent neural networks* // IEEE Trans. on Circuits and Systems I: Fundamental Theory and Applications. – 2000. – Vol. 47, No. 4. – P. 575–578.
55. Morin F., Bengio Y. *Hierarchical probabilistic neural network language model* // Proc. of AISTATS, 2005. – P. 246–252.
56. Graves A., Schmidhuber J. *Framewise Phoneme Classification with Bidirectional LSTM Networks* // Int. Joint Conference on Neural Networks. – 2005. – P. 2047–2052.
57. Graves A. *Generating Sequences With Recurrent Neural Networks* // ArXiv: 1308.0850.
58. Shi X., Chen Z., Wang H., Yeung D., Wong W., Woo W. *Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting* // ArXiv: 1506.04214.
59. Kalchbrenner N., Danihelka I., Graves A. *Grid Long Short-Term Memory* // ArXiv: 1507.01526.
60. Laurent C., Pereyra G., Brakel P., Zhang Y., Bengio Y. Batch Normalized Recurrent Neural Networks // ArXiv: 1510.01378.
61. Amodei D., et al. Deep Speech 2: End-to-End Speech Recognition in English and Mandarin // ArXiv: 1512.02595.
62. Lei Ba J., Kiros J.R., Hinton G.E. Layer Normalization // ArXiv: 1607.06450.
63. Vaswani A., et al. Attention Is All You Need // ArXiv: 1706.03762.
64. Dehghani M., et al. Universal Transformers // ArXiv: 1807.03819.
65. Devlin J., Chang M.W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding // ArXiv: 1810.04805.
66. Dai Z., Yang Z., Yang Y., Carbonell J., Le Q.V., Salakhutdinov R. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context // ArXiv: 1901.02860.
67. Вапник В.Н., Червоненкис А.Я. *О равномерной сходимости частот появления событий к их вероятностям* // Теория вероятн. и ее прим. – 1971. – Т. 16, В. 2. – С. 264–279.
1. Ruder S. *An overview of gradient descent optimization algorithms.* – URL: https://ruder.io/optimizing-gradient-descent.