УДК 519.67

С. И. Хашин

ПРОГНОЗ ЯРКОСТИ СЛЕДУЮЩЕЙ ТОЧКИ С ПОМОЩЬЮ НЕЙРОСЕТИ

При сжатии изображений важным этапом в любом алгоритме является прогнозирование ярости точки на основе яркостей некоторых других, соседних точек. Например, в алгоритме png для прогноза яркости следующей точки через предыдущие используется один из пяти методов: «None», «Sub», «Up», «Average», «Paeth». В настоящей работе мы используем для прогноза яркости следующей точки методы нейронных сетей. Предлагается максимально универсальный способ описания архитектуры нейронных сетей и формата обучающих данных. Рассматриваются результаты большого количества численных экспериментов. Проводится сравнение погрешности нейросетей с погрешностью, полученной другими методами.

Ключевые слова: нейрон, нейросеть, активаторная функция, сжатие изображений.

S. I. Khashin

FORECASTING THE BRIGHTNESS OF THE NEXT POINT USING A NEURAL NETWORK

When compressing images, an important step in any algorithm is predicting the value of each byte (of the image data before filtering) based on the corresponding byte of the pixel to the left, the pixel above, and the pixel above and to the left or some combination thereof. For example, in the png algorithm for predicting the brightness of the next point through the previous one, one of five methods is used: «None», «Sub», «Up», «Average», «Paeth». In this paper, we use neural network methods to predict the brightness of the next point. We propose the most universal way to describe the architecture of neural networks and the format of training data. The results of a large number of numerical experiments are considered. The error of neural networks is compared with the error obtained by other methods.

Key words: neuron, neural network, activation function, image compression.

1. Введение

Нейронные сети в последнее время активно применяются во многих разделах компьютерной графики [2–5]. Одной из классических задач компьютерной графики является сжатие изображений, с потерями или без потерь, см., например, [1]. Если мы используем последовательное кодирование точек изображения, то важной частью алгоритма сжатия будет метод предсказания яркости текущей точки на основании яркостей предыдущих. В алгоритмах, применяемых в стандарте «PNG», используется один из пяти методов: «None», «Sub», «Up», «Average», «Paeth». В них яркость текущей точки предсказывается через яркость не более трех предыдущих, уже закодированных точек. Можно использовать и более сложные алгоритмы, например, полиномиальную аппроксимацию.

В настоящей работе мы используем для прогноза яркости следующей точки методы нейронных сетей. Рассматриваемая задача позволяет для произвольного изображения или набора изображений строить полтора десятка различных обучающих матриц, которые можно использовать в обучении искусственных нейронных сетей. При изучении нейронных сетей важным достоинством этой задачи является возможность сравнения полученного результата с достигаемыми другими методами, например, с помощью полиномиальной аппроксимации.

В разделе 2 рассматривается общий способ задания обучающих матриц в виде текстового csv-файла, пригодный как для задач регрессии, так и для классификации. Формат файла сделан максимально общим, позволяющим без преобразований использовать исходные данные (datasets) из разных источников. Элементы обучающей матрицы могут разделяться запятыми, точками с запятой, пробелами и табуляциями, быть окружены как одиночными, так и двойными кавычками. Так как в русифицированой версии MS Excel при сохранении данных в виде csv-файла разделителем целой и дробной части иногда оказывается не точка, а запятая, такая структура данных предусмотрена предлагаемым форматом. Все возможные варианты обучающей матрицы распознаются и обрабатывается автоматически, не требуя каких-либо дополнительных указаний от пользователя. Здесь же описывается алгоритм построения обучающих матриц на основе произвольного изображения или набора изображений. В следующем разделе приводится результат полиномиальной аппроксимации искомой целевой функции, эти результаты будут использованы в дальнейшем при сравнении с результатами нейронных сетей.

В разделе 4 описываются функции активации, применяемые в нашей системе. В разделе 5 рассмотрен метод задания архитектуры нейронных сетей. Метод достаточно универсален и может применяться для многослойных нейросетей с любым количеством слоев, для полносвязных нейросетей и многих других. В следующем разделе описываются результаты, полученные в результате большого количества численных экспериментов.

Все рассмотренные алгоритмы были реализованы на C++ (Visual Studio, [7]).

2. Обучающие матрицы

Мы применяем нейронные сети для решения двух типов задач: регрессии и классификации.

В первой задаче с помощью нейросети мы хотим получить максимально точную аппроксимацию функции $f\colon \mathbb{R}^n \to \mathbb{R}$, заданной таблично. Это значит, что мы имеем большое количество векторов $x_i \in \mathbb{R}^n$, $i = 1, \ldots, N$, и значения функции в них $y_i = f(x_i) \in \mathbb{R}$. Мы ищем функцию $g\colon \mathbb{R}^n \to \mathbb{R}$ такую, что сумма квадратов отклонений

$$S = \sum_{i=1}^{N} \big(g(x_i) - f(x_i)\big)^2$$

была бы минимальной. В случае нейронной сети эта функция является выходной функцией последнего нейрона.

Во второй задаче мы знаем не значение функции f в каждом из векторов x_i, а принадлежность точки к одному из конечного количества классов $Y_1, \ldots, Y_K$. Если мы имеем K различных классов, то нейронная сеть будет иметь K выходных нейронов, каждый из них соответствует своему классу. Будем считать, что вектор x принадлежит к тому классу, нейрон которого дает наибольшее значение на выходе.

В обоих случаях обучающая матрица задается в виде текстового csv-файла. Первая строка файла, заголовок, пропускается. Каждая следующая строка задает обучающий вектор x_i и значение функции на нем. В задаче регрессии это будет действительное число, в задаче классификации — некоторый идентификатор.

Элементы строк могут разделяться запятыми, точками с запятой, пробелами и табуляциями, а также быть заключенными в одинарные или двойные кавычки. Эти особенности файла наша система обработает автоматически, без дополнительных указаний пользователя. Из-за особенностей русификации MS Excel часто встречаются csv-файлы, в которых дробная часть в действительных числах отделена от целой не точкой, а запятой. Такие файлы также будут корректно обработаны, в качестве разделителя полей в этом случае должны использоваться точки с запятой, пробелы и табуляции. В каждой строке должно быть одно и тоже количество полей, значения типа «n/a», «null», «None» и т. п. недопустимы.

Примеры корректных обучающих матриц:

```
y, x00, ..., x53  Задача классификации
Cat, 1.150700, -1.182809,  0.244507,  0.297739, 0.491739
Dog, 0.560224, -0.707624, -0.650604,  0.500500, 0.435216
Dog, 1.227451,  1.135558, -0.640428, -0.125069,-0.063573
Cat, 0.323810, -0.432380, -0.405235,  0.363134, 0.000000
```

```
y, x00, ..., x53 задача регрессии
1.150700, -1.182809,  0.244507,  0.297739, 0.491739
0.560224, -0.707624, -0.650604,  0.500500, 0.435216
1.227451,  1.135558, -0.640428, -0.125069,-0.063573
0.323810, -0.432380, -0.405235,  0.363134, 0.000000
```

```
y, x00, x01  Задача классификации
Cat; 1.15; -1.18
Dog, 0.56 -0.70
Dog  1.22  1.13
Cat 0.32 -0.43
```

```
y, x00, x01  Задача классификации
Cat; 1,15; -1,18
Dog, 0.56 '-0,70'
Dog 1.227  1.13
Cat "0.32" -0.43
```

В настоящей работе мы применяем нейронные сети для решения задачи прогнозирования яркости точки изображения.

Пусть $a_{i,j}$ — матрица яркости либо одной из RGB-компонент полноцветного изображения, либо серого для черно-белого (серого) изображения. На основе этой матрицы будем строить обучающую матрицу для задачи регрессии для нейронной сети. Для этого яркость точки, отмеченной

символом «X» в следующей таблице, будем пытаться выразить через яркости предыдущих точек.

	-3	**-2**	**-1**	**0**	**1**	**2**	**3**
-3:			15	13	16		
-2:		10	7	5	8	11	
-1:	14	6	2	1	3	9	17
0:	12	4	0	X	-	-	-

Другими словами, мы хотим найти функцию f от 18 аргументов в полном виде такую, что

$$a_{i,j} \approx f(a_{p_0}, a_{p_1}, \ldots, a_{p_{17}}), \tag{1}$$

где точки p_i определяются как

$$\begin{aligned} p_0 &= (i-1,\, j),\\ p_1 &= (i,\, j-1),\\ p_2 &= (i-1,\, j-1),\\ p_3 &= (i-1,\, j+1),\\ &\ldots\\ p_{16} &= (i-3,\, j+1),\\ p_{17} &= (i-1,\, j+3). \end{aligned}$$

Если изображение имеет размеры $mx \times my$, то будем брать индексы i, j в пределах

$$\begin{aligned} 3 \leqslant i < my,\\ 3 \leqslant j < mx - 3. \end{aligned}$$

Таким образом, из каждой цветовой компоненты изображения размера $mx \times my$ мы получаем $(my-3)(mx-6)$ строк обучающей матрицы, в каждой строке в начале записывается яркость точки (i, j), а затем яркости предыдущих 18 точек в указанном выше порядке.

Для применения в нейронных сетях сделаем еще одно преобразование. В каждой строке матрицы вычтем из всех ее элементов второй, равный $f(p_0) = f(i-1,\, j)$. На результатах работы нейронной сети это никак не скажется, но используемые числа будут меньше. В результате второй столбец обучающей матрицы будет состоять из одних нулей и его можно исключить. Так как нейронные сети ориентированы на работу с небольшими числами, дополнительно умножим все коэффициенты обучающей матрицы на 0.01.

Построенную таким образом обучающую матрицу будем подавать на вход нейронной сети. Эта матрица в виде текстового файла будет выглядеть так:

```
 y, x1, x2, x3, x4, x5, x6,    ... ,x16, x17
 0.04, 0.04, 0.00,-0.05, 0.08, ... ,-0.05,-0.03
 0.02, 0.03, 0.05, 0.01,-0.02, ... , 0.01, 0.03
-0.01, 0.02, 0.01, 0.03,-0.01, ... , 0.03, 0.04
 0.02, 0.00,-0.02, 0.00, 0.05, ... , 0.00,-0.01
 ...
```

Обучающую матрицу можно строить по каждой цветовой компоненте любого изображения. Поскольку все такие матрицы имеют одинаковое

количество столбцов, матрицы разных изображений можно объединять. Например, получить общую обучающую матрицу для всех цветовых компонент всех изображений из одной папки.

Аппроксимация яркости точки через 18 предыдущих часто является избыточной. Поэтому мы будем исследовать помимо аппроксимации вида (1) и аппроксимации через меньшее количество точек $K \geqslant 3$. Например, аппроксимация через три предыдущие точки означает поиск формулы вида

$$a_{i,\,j} \approx f_3(a_{i-1,\,j},\, a_{i,\,j-1},\, a_{i-1,\,j-1}) \tag{2}$$

или, с учетом описанного выше преобразования матриц,

$$a_{i,\,j} - a_{i-1,\,j} \approx g_3(a_{i,\,j-1} - a_{i-1,\,j},\, a_{i-1,\,j-1} - a_{i-1,\,j}) \tag{3}$$

Фактически это означает поиск с помощью нейросетей (или каким-то другим методом) функции $g_3(u_1, u_2)$ от двух аргументов, как можно более точно аппроксимирующей данную таблично заданную функцию.

В настоящей работе было взято стандартное изображение lena.bmp размера 512×512 точек, преобразованное к серой палитре. Таким образом была получена обучающая матрица, содержащая 257 554 обучающих векторов. Количество точек, на основе которых прогнозировалась яркость следующей точки, бралось в пределах от 3 до 18, то есть количество входных параметров Nin нейросети — от 2 до 17.

3. Полиномиальная аппроксимация

Для оценки результатов обучения нейронных сетей в описанной ситуации можно обычным методом наименьших квадратов построить полиномиальную аппроксимацию целевой функции многочленами небольшой степени.

Nin	$deg = 1$		$deg = 2$		$deg = 3$		$deg = 4$	
	Nw	S	Nw	S	Nw	S	Nw	S
2	3	0.004661	6	0.004660	10	0.004620	15	0.004591
3	4	0.004178	10	0.004165	20	0.003816	35	0.003792
4	5	0.004022	15	0.003993	35	0.003547	70	0.003510
5	6	0.003663	21	0.003595	56	0.003139	126	0.003087
6	7	0.003624	28	0.003543	84	0.003077	330	0.003008
7	8	0.003596	36	0.003502	120	0.003008	495	0.002922
8	9	0.003418	45	0.003321	165	0.002837	715	0.002747
9	10	0.003366	55	0.003259	220	0.002740	1001	-
10	11	0.003363	66	0.003252	286	0.002724	-	-
11	12	0.003340	78	0.003226	364	0.002693	-	-
12	13	0.003326	91	0.003205	455	0.002657	-	-
13	14	0.003326	105	0.003200	560	-	-	-
14	15	0.003323	120	0.003194	680	-	-	-
15	16	0.003323	136	0.003211	816	-	-	-
16	17	0.003320	153	0.003201	969	-	-	-
17	18	0.003313	171	0.003184	1140	-	-	-

В таблице, приведенной выше, для различного количества входных точек (от 3 до 18; $Nin = 2, 3, \ldots, 17$) приведено количество коэффици-

ентов многочлена (Nw) и средний квадрат отклонения целевой функции от прогноза. Именно с этим числом следует сравнивать результат работы нейросети при сравнимом количестве коэффициентов. Если количество коэффициентов многочлена больше 500, то результаты уже получаются неустойчивые, поэтому они и не приведены в таблице.

Из таблицы можно сделать вывод, что достаточно полной полиномиальной аппроксимацией является аппроксимация многочленами степени 3 при 11 входных точках. Увеличение количества точек или увеличение степени многочлена существенно усложняет вычисления, но практически не дает заметного улучшения результата.

4. Используемые функции активации

Каждый нейрон является функцией $\mathbb{R}^n \to \mathbb{R}$ вида

$$(x_1, \ldots, x_n) \to f(w_0 + w_1 x_1 + \ldots + w_n x_n),$$

где $(x_1, \ldots, x_n)$ — входные аргументы нейрона, $(w_0, w_1, \ldots, w_n)$ — его внутренние параметры и $f\colon \mathbb{R} \to \mathbb{R}$ — некоторая активаторная функция.

Активаторные функции, вообще говоря, могут быть самыми различными, в настоящей работе мы ограничиваемся лишь следующими наиболее распространенными вариантами (см. [6]):

$$
\begin{aligned}
&f_0(x) = x \\
&f_1(x) = (x < 0 \ ? \ 0 : x) && \text{(ReLU)} \\
&f_2(x) = 1/(1 + \exp(-x)) && \text{(сигмоид)} \\
&f_3(x) = (x < 0 \ ? \ -1 : 1) && \text{(sign)} \\
&f_4(x) = (2 \operatorname{arctg} x)/\pi \\
&f_5(x) = x/(1 + |x|) && \text{(softSign)} \\
&f_6(x) = \operatorname{sign} x * x^2/(1 + x^2) && \text{(softSign2)} \\
&f_7(x) = \operatorname{th} x && \text{(тангенс гиперболический)} \\
&f_8(x) = \begin{cases} x, & |x| \leqslant 1 \\ \operatorname{sign} x, & |x| > 1 \end{cases} && \text{(кусочно линейная 1)} \\
&f_9(x) = \begin{cases} -1, & x \leqslant -3 \\ (x-1)/4, & -3 < x < -1 \\ x/2, & -1 \leqslant x \leqslant 1 \\ (x+1)/4, & 1 < x < 3 \\ 1, & 3 \leqslant x \end{cases} && \text{(кусочно линейная 2)}
\end{aligned}
$$

Отметим, что все они, кроме первых трех, являются нечетными.

5. Описание архитектур рассматриваемых нейросетей

Рассмотрим способ задания архитектур используемых нами нейронных сетей [6].

Пусть нейросеть имеет k входных параметров, которые будем обозначать $x_1, \ldots, x_k$. Первый нейрон получает на вход числа $x_1, \ldots, x_k$ и вычисляет результат (обозначим его x_{k+1}) по формуле

$$x_{k+1} = f_{k+1}(w_{k+1,0} + w_{k+1,1} x_1 + \ldots + w_{k+1,k} x_k)$$

Второй нейрон получает на вход числа $x_1, \ldots, x_k, x_{k+1}$, вычисляет результат x_{k+2} по формуле

$$x_{k+2} = f_{k+2}(w_{k+2,0} + w_{k+2,1}x_1 + \ldots + w_{k+2,k}x_k + w_{k+2,k+1}x_{k+1})$$

Таким образом, значение каждого нейрона x_i вычисляется через входные данные нейросети $x_1, \ldots, x_k$, а также через значения всех предыдущих нейронов $x_{k+1}, \ldots, x_{i-1}$ с помощью внутренних параметров $w_{i,0}, \ldots, w_{i,i-1}$.

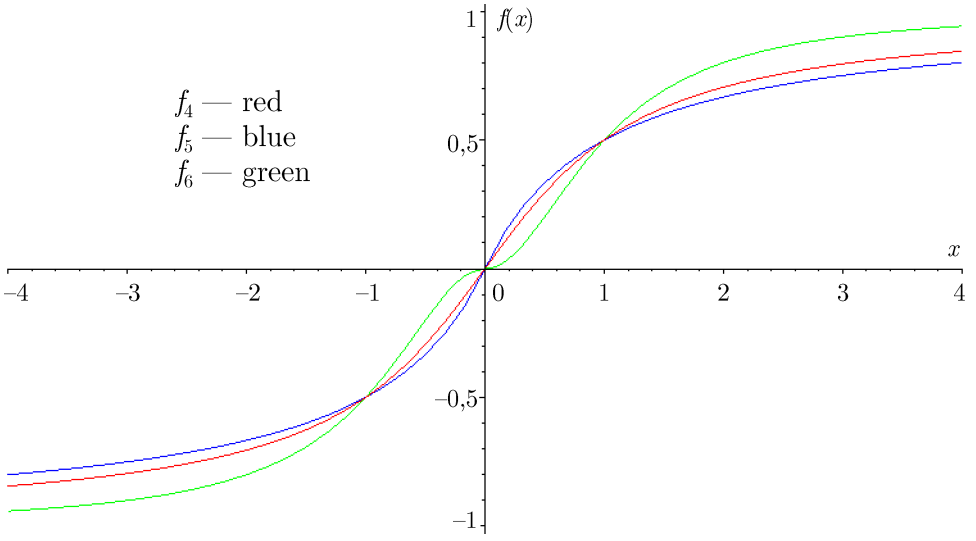

Рис. 1. Функции f_4, f_5, f_6 при $-4 < x < 4$

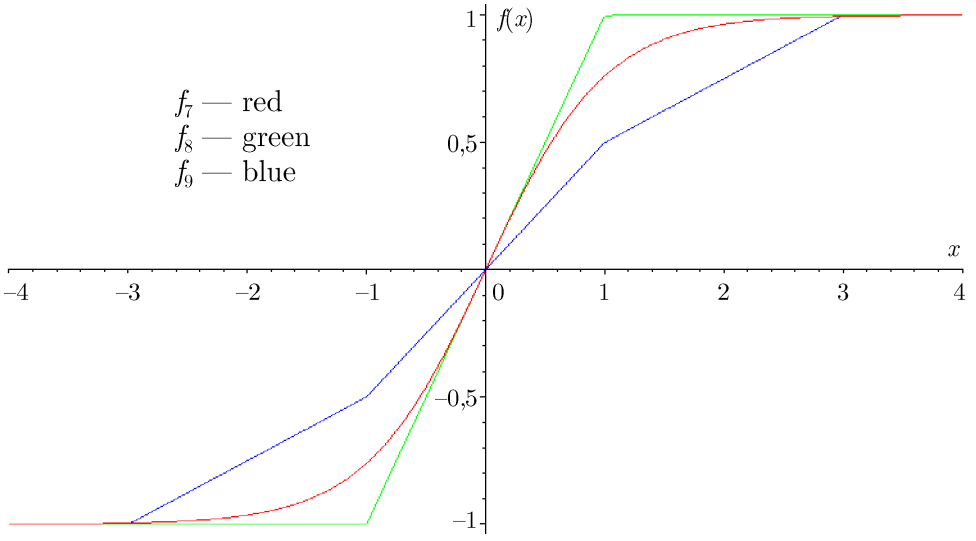

Рис. 2. Функции f_7, f_8, f_9 при $-4 < x < 4$

Нейрон не обязательно зависит от всех предыдущих, часть коэффициентов может быть равна 0. Мы будем явно указывать, от каких из предыдущих зависит каждый нейрон.

В целом, нейронная сеть описывается списком этих зависимостей, набором коэффициентов (w_{ij}) и указанием номера функции активации $(0, \ldots, 9)$ каждого нейрона. Информацию о списках зависимостей и но-

мерах функций активации будем называть архитектурой нейронной сети и задавать в текстовом файле с расширением «.arch». Например, файл содержащий текст:

```
NeiroArch, 3 Layers
 IO 2, 1
 #Layer 1:
   3) 6:   1,  2
   4) 6:   1,  2
   5) 6:   1,  2
 #Layer 2:
   6) 7:   3,  4,  5
   7) 7:   3,  4,  5
   8) 7:   3,  4,  5
   9) 7:   3,  4,  5
 #Layer 3:
  10) 0:   6,  7,  8,  9
```

описывает трехслойную нейронную сеть с двумя входными параметрами. Первая строка файла должна начинаться с символов `NeuroArch` (сигнатура файла). Вторая строка

```
 IO 2, 1
```

задает количество входных и выходных параметров неронной сети. Строки, начинающиеся со знака «#», являются комментариями.

В данном случае входных параметров два, будем их обозначать x_1, x_2, сами нейроны нумеруются, начиная с 3. Нейроны первого слоя (3, 4, 5) имеют функцию активации номер 6 и зависят только от входных параметров x_1, x_2. Нейроны второго слоя (6, 7, 8, 9) имеют функцию активации номер 7 и зависят только от значений нейронов первого слоя (x_3, x_4, x_5). В третьем слое у нас только один нейрон (10) с тривиальной функцией активации $f_0(x) = x$, он зависит только от нейронов второго слоя (x_6, x_7, x_8, x_9).

Предложенный способ записи позволяет описывать нейронные сети любой архитектуры.

В настоящей работе мы рассматриваем задачу регрессии, поэтому все нейросети будут иметь лишь один выходной нейрон, его функцию активации будем полагать тривиальной: $f_0(x) = x$. В качестве функции активации всех остальных нейронов выберем f_6 — softSign2, так как в результате довольно большого количества экспериментов она оказалась в данной задаче наиболее эффективной, хотя ее преимущество перед некоторыми другими невелико.

В дальнейшем в настоящей работе мы будем подробно рассматривать нейросети двух типов архитектур:

1) полносвязная архитектура, когда каждый нейрон зависит от всех входных параметров и всех предыдущих нейронов;

2) трехслойная нейросеть: нейроны первого слоя зависят только от входных параметров, нейроны второго слоя — только от нейронов первого; так как мы рассматриваем задачу регрессии, то в третьем, выходном, слое лишь один нейрон.

Выше был приведен пример файла с описанием трехслойной архитектуры. Приведем также пример файла с описанием полносвязной архитектуры с двумя входными параметрами и 5-ю нейронами:

```
NeiroArch, Full net with 5 neurons
IO 2, 1
3) 6: 1,  2
4) 6: 1,  2,  3
5) 6: 1,  2,  3,  4
6) 6: 1,  2,  3,  4,  5
7) 0: 1,  2,  3,  4,  5,  6
```

Функции активации у всех нейронов — f_6, кроме последнего, выходного, нейрона, у которого она тривиальна: $f_0(x) = x$.

6. Результаты обучения нейросетей

Рассмотрим результаты, полученные при обучении нейросетей. Так как обучение нейросети начинается с выбора случайной точки, для каждой архитектуры сделаем несколько попыток и выберем из них наилучший результат.

Сначала приведем результаты, полученные для полносвязных архитектур при 3, ..., 18 предыдущих точках ($Nin = 2, \ldots, 17$) и при количестве нейронов от 3 до 6. В таблице ниже Nw — количество параметров нейросети, $S(3)$, $S(4)$, $S(5)$, $S(6)$ — минимальные достигнутые значения целевой функции при 3, 4, 5, 6 нейронах соответственно. Первые столбцы в таблице: nPt — количество точек, на основе которых строится прогноз, Nin — количество входных параметров нейросети, $Nin = nPt - 1$.

nPt	Nin	Nw	$S(3)$	Nw	$S(4)$	Nw	$S(5)$	Nw	$S(6)$
3	2	12	0.004524	18	0.004512	25	0.004505	33	0.004487
4	3	15	0.003898	22	0.003858	30	0.003859	39	0.003694
5	4	18	0.003712	26	0.003549	35	-	45	-
6	5	21	0.003269	30	-	40	-	51	-
7	6	24	0.003227	34	0.003104	45	0.002965	57	0.002917
8	7	27	0.003158	38	0.003076	50	0.002960	63	0.002864
9	8	30	0.003013	42	0.002934	55	0.002869	69	0.002802
10	9	33	0.002960	46	0.002875	60	0.002808	75	0.002792
11	10	36	0.002963	50	0.002887	65	0.002827	81	0.002757
13	12	39	0.002916	54	0.002853	70	0.002865	87	0.002854
14	13	42	0.002901	58	0.002835	75	0.002823	93	0.002713
15	14	45	0.002892	62	0.002834	80	0.002780	99	0.002735
18	17	54	0.002965	74	0.002977	95	0.002978	117	0.002974

В следующей таблице показаны наилучшие результаты, полученные с помощью 3-слойных нейросетей. В первом и втором слое у них бралось по одинаковому количеству нейронов (Nne) с функцией активации f_6, в третьем, выходном, слое был один нейрон с тривиальной функцией активации.

nPt	*Nin*	*Nne*	*Nw*	*S*	*Nne*	*Nw*	*S*
6	5	15	49	0.003030	27	181	0.002861
		17	66	0.002950	31	241	0.002840
		19	85	0.002915	37	346	0.002821
		23	129	0.002864			
7	6	16	53	0.002989	28	191	0.002782
		18	71	0.002927	32	253	0.002776
		20	91	0.002863	38	361	0.002768
		24	137	0.002794			
8	7	17	57	0.002928	29	201	0.002689
		19	76	0.002861	33	265	0.002674
		21	97	0.002819	39	376	0.002665
		25	145	0.002736			
9	8	18	61	0.002796	30	211	0.002612
		20	81	0.002728	34	277	0.002573
		22	103	0.002705	40	391	0.002591
10	9	19	65	0.002767	31	221	0.002549
		21	86	0.002675	35	289	0.002532
		23	109	0.002650	41	406	0.002532
		27	161	0.002597			
11	10	20	69	0.002737	32	231	0.002530
		22	91	0.002665	36	301	0.002537
		24	115	0.002612	42	421	0.002557
		28	169	0.002598			
12	11	21	73	0.002718	29	177	0.002581
		23	96	0.002678	33	241	0.002563
		25	121	0.002623	43	436	0.002543
18	17	27	97	0.002685	39	301	0.002528
		29	126	0.002649	43	385	0.002554
		31	157	0.002629	49	526	0.002598
		35	225	0.002595			

7. Заключение

Таким образом, для одних и тех же обучающих матриц были рассмотрены следующие варианты решения задачи регрессии:

1) полиномиальная функция от своих аргументов (*Polynom*);

2) полносвязная нейронная сеть (*Full net*);

3) трехслойная нейронная сеть (3 *Layers*).

Выпишем для сравнения кратко общие результаты, полученные с помощью этих трех методов при различных уровнях сложности *Nw* (количестве настраиваемых параметров):

Type	$Nw < 50$	$Nw < 100$	$Nw < 200$	$Nw < 500$
Polynom	0.003321	0.003077	0.002837	0.002657
Full Net	0.002887	0.002735	-	-
3 *Layers*	0.003030	0.002675	0.002581	0.002532

На основе полученных данных можно сделать следующие выводы.

1. Нейросети позволяют получить ту же величину погрешности, что и полиномиальные формулы при сложности (количестве параметров, Nw) в 3 — 5 раз меньше.

2. При сравнительно небольшой сложности ($Nw < 50$) наиболее эффективными оказываются полносвязные сети.

3. При большей сложности трехслойные сети наиболее эффективны среди всех рассмотренных вариантов.

Библиографический список

1. *Ватолин Д., Ратушняк А., Смирнов М., Юдин В.* Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. М.: Диалог-МИФИ, 2002. 384 с.
2. *Галушкин А. И.* Нейрокомпьютеры: учебное пособие. М.: Альянс, 2014. 528 с.
3. *Гафаров Ф. М., Галимянов А. Ф.* Искусственные нейронные сети и их приложения. Казань: Изд-во Казанского ун-та, 2018. 121 с.
4. *Головко В. А., Краснопрошин В. В.* Нейросетевые технологии обработки данных : учеб. пособие. Минск: БГУ, 2017. 263 с.
5. *Николенко С., Кадурин А., Архангельская Е.* Глубокое обучение // СПб.: Питер, 2018. 480 с.
6. *Хашин С. И.* Сравнение эффективности передаточных функций нейросети // Вестник Ивановского государственного университета. Сер.: Естественные, общественные науки. 2019. Вып. 1/2. С. 54–58.
7. Центр загрузки Microsoft. URL: `https://www.microsoft.com/ru-ru/download` (дата обращения: 20.11.2020).