ИССЛЕДОВАНИЕ СТРУКТУРЫ РЕЦЕПТИВНОГО ПОЛЯ В ВИЗУАЛЬНОМ МЕТОДЕ РЕШЕНИЯ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ*

Н.А. Ольховский

Южно-Уральский государственный университет
(национальный исследовательский университет)

В статье описан метод определения проекции произвольного вектора на гиперплоскость в многомерном пространстве путем взаимодействия суперкомпьютера и искусственной нейронной сети. Суперкомпьютер строит визуальное представление гиперплоскости в окрестности некоторой точки. Визуальный образ обрабатывается нейронной сетью прямого распространения. В качестве ответа нейронная сеть возвращает направление проекции заданного вектора на гиперплоскость. В статье описано построение обучающего множества, проектирование, оптимизация и обучение нейронной сети. Представлены результаты вычислительных экспериментов с различными конфигурациями нейронных сетей.

Ключевые слова: линейное программирование, нестационарные задачи, режим реального времени, метод поверхностного движения, итерационный метод, теорема сходимости, искусственная нейронная сеть, глубокое обучение, параллельный алгоритм, высокопроизводительные вычисления

1. Введение

Одной из важнейших фундаментальных задач современной прикладной математики является задача линейного программирования (ЛП) с большим числом параметров [1]. Оптимизационные модели, основанные на многопараметрической (многомерной) задаче ЛП встречаются в системах поддержки принятия решений в экономике [2,3], в системах управления беспилотными летательными аппаратами [4], в управлении технологическими процессами [5–7], при построении логистических цепочек [8–10], в оперативном управлении и планировании [11, 12].

В работе [13] предложено оригинальное решение обозначенной проблемы: построить визуальный образ многогранника и при помощи искусственной нейронной сети (ИНС) прямого распространения определить направление максимального увеличения целевой функции. Однако при определении направления движения в пересечении гиперплоскостей, базовой задачей является с помощью ИНС определить направление в отдельно взятой гиперплоскости. Обзор литературы показал, что исследования в данном направлении не проводились. В данной публикации описывается попытка разработать и обучить нейросетевую модель, определяющую для произвольной гиперплоскости, расположенной в пространстве высокой размерности, направление, соответствующее увеличению некоторой целевой функции.

Структура статьи следующая. В разделе 2 изложены теоретические основы, определяющие основы работы нейросетевой модели. Раздел 3 раскрывает процедуру генерации обучающего множества и построение нейросетевой модели. В разделе 4 приводятся результаты вычислительных экспериментов, полученные в результате обучения нейронной сети. Наконец, в разделе 5 приведены выводы по результатам экспериментов и предложены идеи относительно дальнейших исследований.

*Исследование выполнено при финансовой поддержке РНФ (проект № 23-21-00356).

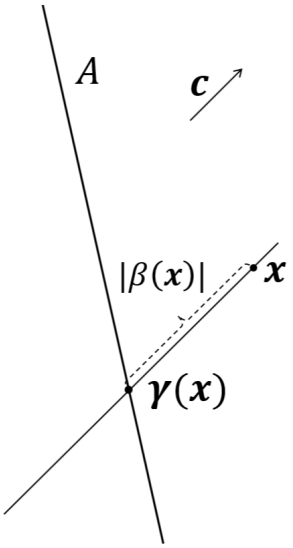


Рис. 1: Целевая проекция точки $\boldsymbol{x}$ на гиперплоскость $H_{\boldsymbol{a}}$.

2. Теоретические основы

Без потери общности мы можем ограничиться рассмотрением гиперплоскости, проходящей через начало координат. Зафиксируем вектор $\boldsymbol{a}$, коэффициенты которого являются случайными величинами, соответствующими нормальному распределению с коэффициентами $\mu = 0$, $\sigma^2 = 1$:

$$\boldsymbol{a} = \{a_i = |\mathcal{N}(0,1)| \mid i \in \{1, \ldots, n\}\}. \tag{1}$$

Обозначим $H_{\boldsymbol{a}}$ – такую гиперплоскость в пространстве $\mathbb{R}^n$, для которой вектор $\boldsymbol{a}$ является нормалью

$$H_{\boldsymbol{a}} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \langle \boldsymbol{a}, \boldsymbol{x} \rangle = 0\}. \tag{2}$$

Построим случайный *целевой вектор* $\boldsymbol{c} \in \mathbb{R}^n$ с тем же распределением

$$\boldsymbol{c} = \{c_i = |\mathcal{N}(0,1)| \mid i \in \{1, \ldots, n\}\}. \tag{3}$$

Обозначим $\boldsymbol{g} \in \mathbb{R}^n$ ортогональную проекцию целевого вектора $\boldsymbol{c}$ на гиперплоскость $H_{\boldsymbol{a}}$

$$\boldsymbol{c} = \boldsymbol{c} - \frac{\langle \boldsymbol{a}, \boldsymbol{c} \rangle}{||\boldsymbol{a}||^2} \boldsymbol{a}. \tag{4}$$

Построим *целевую гиперплоскость* H_c, ортогональную вектору $\boldsymbol{c}$:

$$H_{\boldsymbol{c}} = \{\boldsymbol{x} \in \mathbb{R}^n \mid \langle \boldsymbol{c}, \boldsymbol{x} \rangle = 0\}. \tag{5}$$

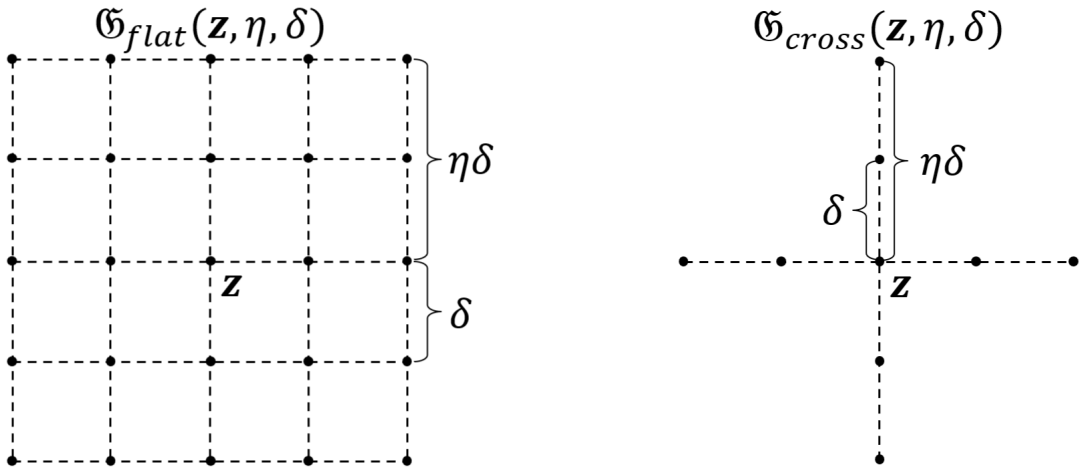
Целевая проекция $\gamma : H_{\boldsymbol{c}} \to H_{\boldsymbol{a}}$ вычисляется по следующему правилу

$$\gamma(\boldsymbol{x}) = \boldsymbol{x} - \frac{\langle \boldsymbol{a}, \boldsymbol{x} \rangle}{\langle \boldsymbol{a}, \boldsymbol{c} \rangle} \boldsymbol{c}. \tag{6}$$

Смещение $\beta : H_{\boldsymbol{c}} \to \mathbb{R}$ вычисляется по следующему правилу

$$\beta(\boldsymbol{x}) = -\frac{\langle \boldsymbol{a}, \boldsymbol{x} \rangle}{\langle \boldsymbol{a}, \boldsymbol{c} \rangle} \|\boldsymbol{c}\|. \tag{7}$$

Построим в гиперплоскости $H_{\boldsymbol{c}}$ набор базисных векторов следующим образом:



(a) равномерное (b) крестообразное

Рис. 2: Рецептивное поле в пространстве $\mathbb{R}^3$.

$$
\begin{aligned}
&\boldsymbol{c}^{(0)} = \boldsymbol{c} = (c_1, c_2, c_3, c_4, \ldots, c_{n-1}, c_n);\\
&\boldsymbol{c}^{(1)} = \begin{cases} \left(-\frac{1}{c_1}\sum_{i=2}^{n} c_i^2, c_2, c_3, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_1 \neq 0;\\ (1,0,\ldots,0), \text{ если } c_1 = 0; \end{cases}\\
&\boldsymbol{c}^{(2)} = \begin{cases} \left(0, -\frac{1}{c_2}\sum_{i=3}^{n} c_i^2, c_3, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_2 \neq 0;\\ (0,1,0,\ldots,0), \text{ если } c_2 = 0; \end{cases}\\
&\boldsymbol{c}^{(3)} = \begin{cases} \left(0,0, -\frac{1}{c_3}\sum_{i=4}^{n} c_i^2, c_4, \ldots, c_{n-1}, c_n\right), \text{ если } c_3 \neq 0;\\ (0,0,1,0,\ldots,0), \text{ если } c_3 = 0; \end{cases}\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&\boldsymbol{c}^{(n-2)} = \begin{cases} \left(0,\ldots,0, -\frac{1}{c_{n-2}}\sum_{i=n-1}^{n} c_i^2, c_{n-1}, c_n\right), \text{ если } c_{n-2} \neq 0;\\ (0,\ldots,0,1,0,0), \text{ если } c_{n-2} = 0; \end{cases}\\
&\boldsymbol{c}^{(n-1)} = \begin{cases} \left(0,\ldots,0, -\frac{c_n^2}{c_{n-1}}, c_n\right), \text{ если } c_{n-1} \neq 0;\\ (0,\ldots,0,0,1,0), \text{ если } c_{n-1} = 0. \end{cases}
\end{aligned}
$$

Легко видеть, что

$$\forall i, j \in \{0,1,\ldots,n-1\}, i \neq j : \left\langle \boldsymbol{c}^{(i)}, \boldsymbol{c}^{(j)} \right\rangle = 0.$$

В том числе

$$\forall i = 1,\ldots,n-1 : \left\langle \boldsymbol{c}, \boldsymbol{c}^{(i)} \right\rangle = 0. \tag{8}$$

Обозначим E_c – ортонормированный базис

$$E_c = \left\{ \boldsymbol{e}^{(i)} = \frac{\boldsymbol{c}^{(i)}}{\|\boldsymbol{c}\|} \middle| i \in \{1,\ldots,n-1\} \right\}. \tag{9}$$

Кубическим рецептивным полем $\mathfrak{G}_{\text{cube}}(\boldsymbol{z},\eta,\delta) \subset H_c$ плотности $\delta \in \mathbb{R}_{>0}$ с центром в точке $\boldsymbol{z} = \boldsymbol{0}$ и рангом $\eta \in \mathbb{N}$ будем называть конечное упорядоченное множество точек, порождаемое алгоритмом 1 из [13].

Крестообразным рецептивным полем $\mathfrak{G}_{\text{cross}}(\boldsymbol{z},\eta,\delta) \subset H_c$ плотности $\delta \in \mathbb{R}_{>0}$ с центром в точке $\boldsymbol{z} = \boldsymbol{0}$ и рангом $\eta \in \mathbb{N}$ будем называть конечное упорядоченное множество точек,

Algorithm 1 Построение крестообразного рецептивного поля $\mathfrak{G}_{\text{cross}}(z, \eta, \delta)$

Require: $\boldsymbol{z} := \mathbf{0}$, $\eta \in \mathbb{N}$, $\delta \in \mathbb{R}_{>0}$

$\mathfrak{G} := \emptyset$
for $t = 1 \ldots n-1$ **do**
  for $i = 1 \ldots 2\eta$ **do**
    $\boldsymbol{s} := \mathbf{0}$
    for $j = 1 \ldots n-1$ **do**
      if $j \neq t$ **then**
        $s_j = 0$
      else
        if $j \leqslant \eta$ **then**
          $s_j := (i - \eta - 1)\delta$
        else
          $s_j := (i - \eta)\delta$
        end if
      end if
      $\boldsymbol{s} := \boldsymbol{s} + s_j \boldsymbol{e}^{(j)}$
    end for
    $\mathfrak{G} := \mathfrak{G} \cup \boldsymbol{s} + \boldsymbol{z}$
  end for
end for
$\mathfrak{G} := \mathfrak{G} \cup \mathbf{0}$
return $\mathfrak{G}$

порождаемое следующим алгоритмом 1. Поскольку хранить постоянно весь массив точек нецелесообразно, на практике координаты точки вычисляют по ее номеру в любой момент времени при помощи специального алгоритма 2.

Образом рецептивного поля $\mathcal{I}(\mathfrak{G})$ назовем набор смещений, вычисленных для каждой точки рецептивного поля $\mathfrak{G}$:

$$\mathcal{I} = \{\beta(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathfrak{G}\}. \tag{10}$$

Таким образом набор данных, характеризующий одну случайно сгенерированную гиперплоскость и соответствующий ей образ, содержит следующие элементы:

- образ рецептивного поля $\mathcal{I}(\mathfrak{G})$, вычисленный для гиперплоскости A;
- вектор правильного ответа $\boldsymbol{g}$, соответствующий целевому вектору $\boldsymbol{c}$.

Коэффициентом угла наклона вектора правильного ответа $\boldsymbol{g}$ к базису рецептивного поля назовем косинус угла наклона, вычисляемый следующим образом.

$$\cos \alpha_i = \frac{\langle \boldsymbol{e}^{(i)}, \boldsymbol{g} \rangle}{\|\boldsymbol{g}\|}. \tag{11}$$

Взятые вместе, коэффициентов углов наклона образуют *вектор правильных ответов* $\mathcal{A}(\boldsymbol{g})$.

$$\mathcal{A}(\boldsymbol{g}) = \left\{ \frac{\langle \boldsymbol{e}^{(i)}, \boldsymbol{g} \rangle}{\|\boldsymbol{g}\|} \middle| i \in 1, \ldots, n-1 \right\}. \tag{12}$$

В этом случае один *прецедент обучающего множества* будет состоять из:

- образа рецептивного поля $\mathcal{I}(\mathfrak{G})$;
- вектора правильных ответов $\mathcal{A}(\boldsymbol{g})$.

Algorithm 2 Функция G вычисляет точку рецептивного поля по ее номеру k

Require: $\boldsymbol{z} := \mathbf{0}$, $\eta \in \mathbb{N}$, $\delta \in \mathbb{R}_{>0}$

function $\mathrm{G}(k, n, \boldsymbol{z}, \eta, \delta)$
  if $k = 2\eta + 1$ **then**
    $\boldsymbol{g} := \mathbf{0}$
    return $\boldsymbol{g}$
  end if
  $l := \lfloor (k-1)/2\eta \rfloor + 1$
  $k := (k-1) \mod 2\eta + 1$
  $\boldsymbol{g} := \boldsymbol{z}$
  if $k \leqslant \eta$ **then**
    $\boldsymbol{g} := \boldsymbol{g} + (i - \eta - 1)\delta \boldsymbol{e}^{(l)}$
  else
    $\boldsymbol{g} := \boldsymbol{g} + (i - \eta)\delta \boldsymbol{e}^{(l)}$
  end if
  return $\boldsymbol{g}$
end function

3. Обучение нейронных сетей

Создание обучающего множества и обучение искусственной нейронной сети производилось по схеме, представленной на Рис. 4. Сначала программа генерации случайных гиперплоскостей, приняв в качестве входного параметра число N, генерирует N пар $\{\boldsymbol{a}, \boldsymbol{c}\}$, где каждая пара состоит из случайного вектора (1), определяющего гиперплоскость (5) и случайного вектора целевой функции (3). Массив сгенерированных данных помещается в файл, передаваемый программе визуализации. Визуализатор принимает входные параметры, определяющие положение точек рецептивного поля:

- ранг рецептивного поля η;
- плотность рецептивного поля δ;
- форму рецептивного поля `cube` (кубическое) или `cross` (крестообразное).

Число точек рецептивного поля $K = |\mathcal{I}(\mathfrak{G})|$ зависит от параметров и по-разному вычисляется для кубического рецептивного поля

$$K_{\mathrm{cube}} = (2\eta + 1)^{n-1}, \tag{13}$$

и крестообразного

$$K_{\mathrm{cross}} = 2\eta(n-1) + 1. \tag{14}$$

В соответствии с принятыми параметрами визуализатор вычисляет положение точек рецептивного поля и для каждой точки определяет смещение. Множество смещений нормализуется по формуле:

$$x = \frac{x - \min(\mathcal{I}(\mathfrak{G}))}{\max(\mathcal{I}(\mathfrak{G})) - \min(\mathcal{I}(\mathfrak{G}))} \cdot 511 - 256, \text{ где } x \in \mathcal{I}(\mathfrak{G}). \tag{15}$$

Также визуализатор вычисляет вектор правильных ответов. В результате работы визуализатора формируется файл прецедентов, содержащий N строк. В каждой строке записаны K нормализованных по формуле (15) коэффициентов, формирующих образ рецептивного

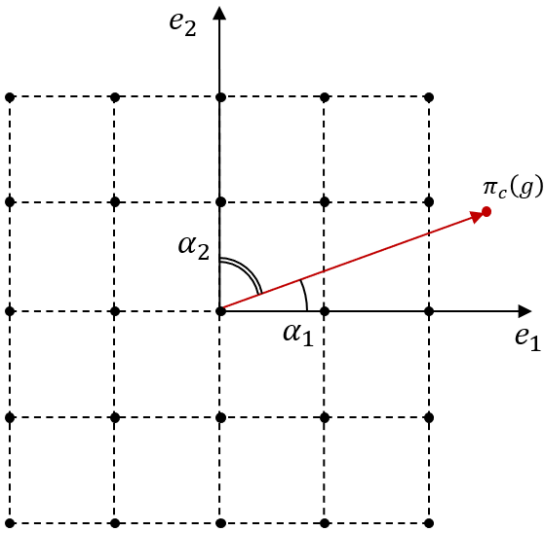


Рис. 3: Коэффициенты угла наклона в рецептивном поле в $\mathbb{R}^3$.

поля $\mathcal{I}(\mathfrak{G})$. И $n-1$ коэффициентов угла наклона, формирующих вектор правильных ответов $\mathcal{A}(\boldsymbol{g})$.

При обучении ИНС множество прецедентов случайным образом делилось по правилу 80/15/5. 80% прецедентов использовались для обучения, 15% – для валидации в конце каждой эпохи, 5% – для тестирования нейронной сети после окончания обучения.

Обучение производилось при помощи библиотеки `keras` пакета `tensorflow`. При обучении нейронных сетей использовались следующие метрики. В качестве функции потерь использовалась средняя абсолютная ошибка (MAE).

$$\mathrm{MAE} = \frac{1}{n}\sum |\boldsymbol{y}_{\mathrm{true}} - \boldsymbol{y}_{\mathrm{pred}}| \tag{16}$$

Для оценки достигаемой точности ответов нейронной сети была взята косинусовая мера (Cosine Similarity).

$$\mathrm{CS} = \frac{\sum(\boldsymbol{y}_{\mathrm{true}} \cdot \boldsymbol{y}_{\mathrm{pred}})}{\sqrt{\sum \boldsymbol{y}_{\mathrm{true}}^2} \cdot \sqrt{\sum \boldsymbol{y}_{\mathrm{pred}}^2}}. \tag{17}$$

Здесь $\boldsymbol{y}_{\mathrm{pred}}$ – это значения коэффициентов наклона, предсказанные нейронной сетью, а $\boldsymbol{y}_{\mathrm{true}}$ – коэффициенты наклона, рассчитанные визуализатором.

4. Вычислительные эксперименты

Были проведены две серии вычислительных экспериментов. Вычисления проводились на комплексе «Нейрокомпьютер» Южно-Уральского государственного университета на графическом ускорителе NVIDIA Tesla V100 SXM2 (5120 ядер CUDA, объем видеопамяти: 32 GB).

Первой задачей было исследовать зависимость точности, достигаемой при обучении нейронных сетей, от конфигурации и плотности рецептивного поля. Для этого в пространстве $\mathbb{R}^3$ и $\mathbb{R}^4$ были взяты рецептивные поля с рангом от 1 до 5 и плотностью, подобранной так, чтобы длина рецептивного поля по каждому измерению примерно равнялась 10. Параметры рецептивных полей и соответствующее им количество точек для конфигураций `cube` и `cross` представлены в Таблице 1.

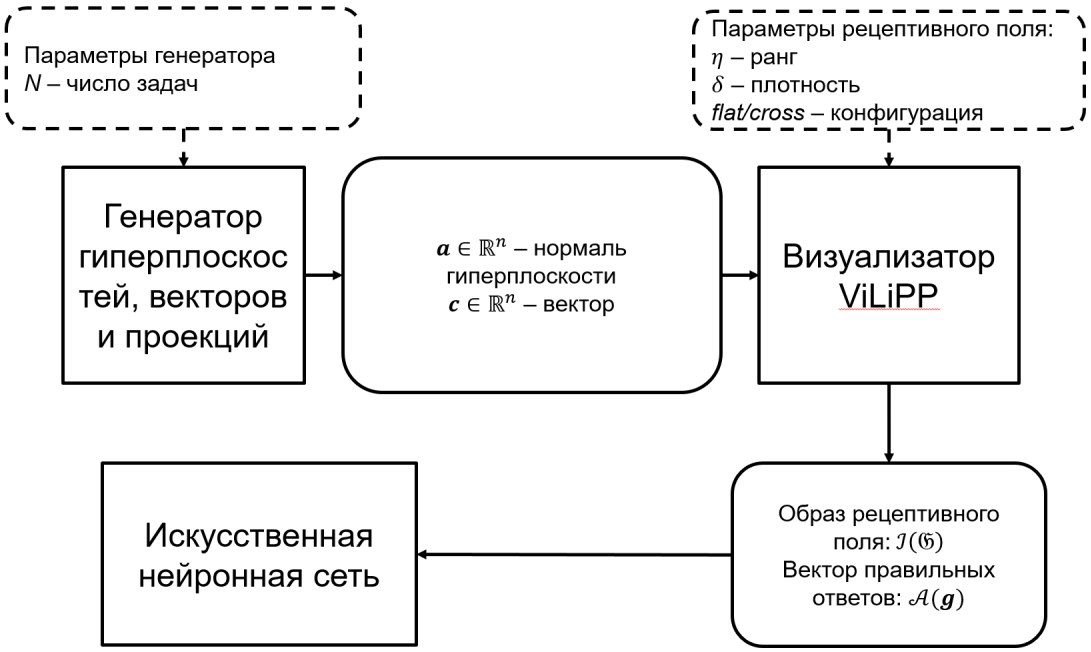


Рис. 4: Архитектура программного комплекса, генерирующего прецеденты для обучения ИНС.

Таблица 1: Параметры рецептивных полей в $\mathbb{R}^3$ и $\mathbb{R}^4$.

Параметры	K_{cube}	K_{cross}
$n = 3$, $\eta = 1$, $\delta = 5$	9	5
$n = 3$, $\eta = 2$, $\delta = 2.5$	25	9
$n = 3$, $\eta = 3$, $\delta = 1.66667$	49	13
$n = 3$, $\eta = 4$, $\delta = 1.25$	81	17
$n = 3$, $\eta = 5$, $\delta = 1$	121	21
$n = 4$, $\eta = 1$, $\delta = 5$	27	7
$n = 4$, $\eta = 2$, $\delta = 2.5$	125	13
$n = 4$, $\eta = 3$, $\delta = 1.66667$	343	19
$n = 4$, $\eta = 4$, $\delta = 1.25$	729	25
$n = 4$, $\eta = 5$, $\delta = 1$	1331	31

Для каждой конфигурации нейронная сеть имеет входной слой, состоящий из K нейронов, и выходной слой из одного нейрона. Соответственно для каждого коэффициента $\cos\alpha_i$ обучалась отдельная нейронная сеть. Архитектура использовавшихся ИНС представлена на Рис. 5. Средняя точность ответов ИНС по всем координатам в $\mathbb{R}^3$ представлена на Рис. 6. Видно, что кубическое рецептивное поле демонстрирует чуть лучшую точность при низком ранге. Однако начиная с ранга 3 различие несущественны. Результаты экспериментов в $\mathbb{R}^4$ показали, что, начиная с ранга 3, крестообразное рецептивное поле демонстрирует даже

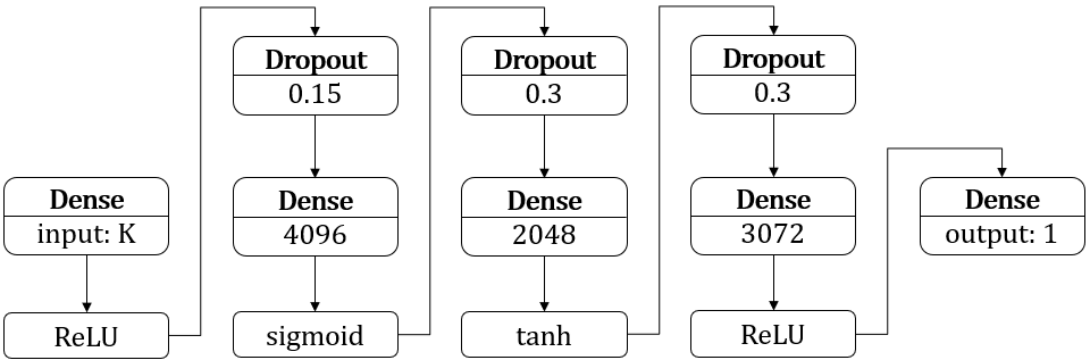
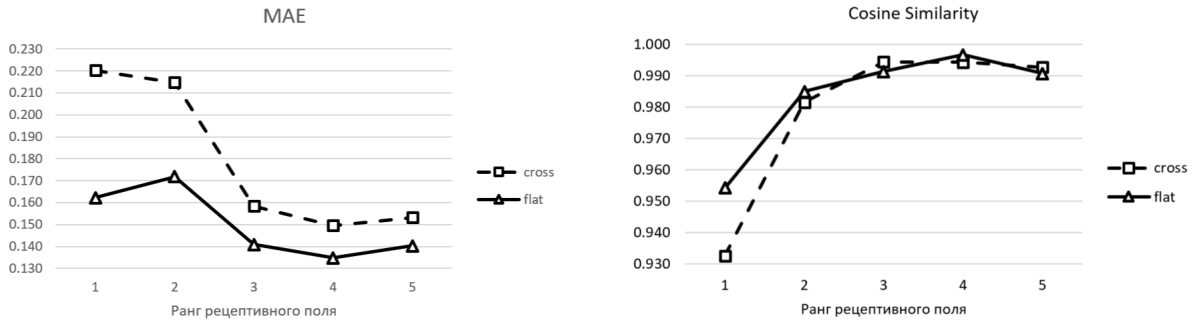


Рис. 5: Архитектура экспериментальной искусственной нейронной сети.



(a) средняя абсолютная ошибка

(b) косинусовая мера

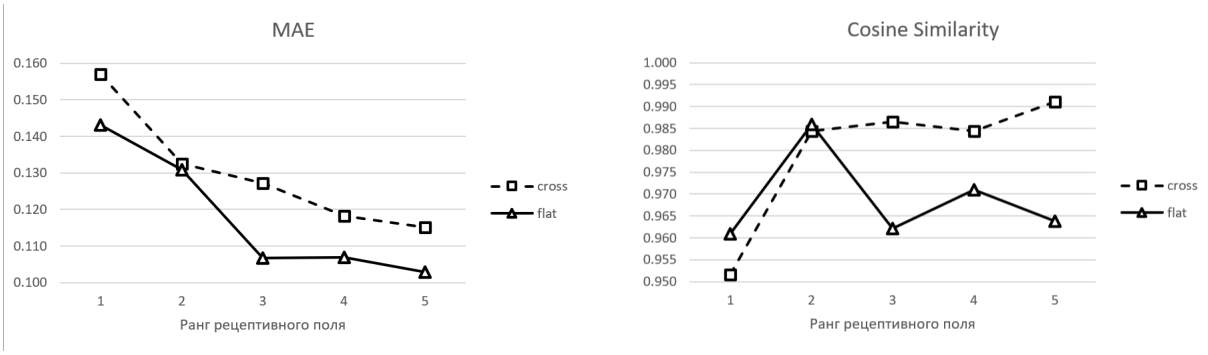
Рис. 6: Точность работы ИНС в $\mathbb{R}^3$.

чуть более высокие результаты, чем кубическое. Графики представлены на Рис. 7.

Второй задачей было установить удастся ли исключить падение точности ответов нейронной сети при увеличении размерности, если увеличивать число скрытых слоев и производить оптимизацию гиперпараметров для каждой размерности. Основываясь на результатах первой части для испытаний была выбрана крестообразная форма рецептивного поля. Для осуществления поиска в пространстве гиперпараметров использовался Байесовский подход. Варианты гиперпараметров ИНС, среди которых осуществлялся поиск:

- ранг рецептивного поля равен размерности $\eta = n$;
- плотность рецептивного поля равна единице $\delta = 1$;
- число скрытых слоев $\{n-1, n, n+1\}$;
- число нейронов в скрытом слое $\{1024, 2048, 4096\}$;
- коэффициент прореживания (dropout) $\{0., 0.15, 0.30, 0.45\}$;
- функция активации $\{\mathrm{ReLU}, \mathrm{sigmoid}, \tanh\}$;
- размер выборки (batch size) $\{128, 256, 512, 1024\}$;
- скорость обучения $\{0.01, 0.005, 0.001, 0.0005, 0.0001, 0.00005, 0.00001\}$.

Для каждого количества скрытых слоев выполнялось 30 попыток подбора гиперпараметров. В Таблице 2 приведены минимальные значения средней абсолютной ошибки и максимальные значения косинусовой меры, которые удалось достигнуть.



(a) средняя абсолютная ошибка

(b) косинусовая мера

Рис. 7: Точность работы ИНС в $\mathbb{R}^4$.

Таблица 2: Максимальные показатели точности крестообразного рецептивного поля в $\mathbb{R}^6$, $\mathbb{R}^8$ и $\mathbb{R}^{10}$.

Параметры	MAE	Cosine Similarity
$n = 6$, 5 скрытых слоев	0.0133856749162077	0.990947365760803
$n = 6$, 6 скрытых слоев	0.0174567010253667	0.987473666667938
$n = 6$, 7 скрытых слоев	0.0124648576602339	0.989684224128723
$n = 8$, 7 скрытых слоев	0.0064711607992649	0.99115788936615
$n = 8$, 8 скрытых слоев	0.0208722446113824	0.987157881259918
$n = 8$, 9 скрытых слоев	0.0237346738576889	0.980842113494873
$n = 10$, 9 скрытых слоев	0.0121533023193478	0.990631580352783
$n = 10$, 10 скрытых слоев	0.0073878364637494	0.986105263233184
$n = 10$, 11 скрытых слоев	0.0179088432341814	0.988421022891998

5. Заключение

В настоящей работе была исследована зависимость точности нейронной сети, от конфигурации рецептивного поля. Установлено, что разница в точности работы кубического рецептивного поля незначительно отличается от крестообразного рецептивного поля.

Повышение ранга от 1 до 3 значительно увеличивает точность работы крестообразного рецептивного поля. А для кубического – результаты противоречивые: средняя абсолютная ошибка значительно снижается, но косинусовая мера колеблется около значения 0.97.

При повышении размерности пространства подбор гиперпараметров ИНС позволяет сохранять высокую точность по обеим метрикам для крестообразного рецептивного поля! Причем важнейшей метрикой является скорость обучения. Наилучшие результаты получаются при `learning rate = 0.00005`. При уменьшении до `learning rate = 0.00001` обучение становится бесконечно долгим. При повышении до `learning rate = 0.0001` резко снижается итоговая точность.

Для применения полученных результатов к решения многомерной задачи линейного

программирования целесообразно провести исследование достигаемой точности в размерностях 1000, 10 000, 100 000 при ограниченном небольшом числе скрытых слоев.

Также необходимо исследовать возможности по определению проекций векторов в пространства, образуемые пересечением гиперплоскостей высоких размерностей.

Литература

1. *Соколинская И.М., Соколинский Л.Б.* О решении задачи линейного программирования в эпоху больших данных // Параллельные вычислительные технологии (ПаВТ'2017). Короткие статьи и описания плакатов. Челябинск: Издательский центр ЮУрГУ, 2017. С. 471–484. URL: http://omega.sp.susu.ru/pavt2017/short/014.pdf.

2. *Brogaard J., Hendershott T., Riordan R.* High-Frequency Trading and Price Discovery // Review of Financial Studies. 2014. Vol. 27, no. 8. P. 2267–2306. DOI: 10.1093/rfs/hhu032.

3. *Deng S., Huang X., Wang J., et al.* A Decision Support System for Trading in Apple Futures Market Using Predictions Fusion // IEEE Access. 2021. Vol. 9. P. 1271–1285. DOI: 10.1109/ACCESS.2020.3047138.

4. *Seregin G.* Lecture notes on regularity theory for the Navier-Stokes equations. Singapore: World Scientific Publishing Company, 2014. 268 p. DOI: 10.1142/9314.

5. *Demin D.A.* Synthesis of optimal control of technological processes based on a multialternative parametric description of the final state. Eastern-European Journal of Enterprise Technologies. 2017. Vol. 3, 4(87). P. 51–63. DOI: 10.15587/1729-4061.2017.105294.

6. *Kazarinov L.S., Shnayder D.A., Kolesnikova O.V.* Heat load control in steam boilers. 2017 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2017 - Proceedings. IEEE, 2017. DOI: 10.1109/ICIEAM.2017.8076177.

7. *Zagoskina E.V., Barbasova T.A., Shnaider D.A.* Intelligent Control System of Blast-furnace Melting Ffiency. SIBIRCON 2019 - International Multi-Conference on Engineering, Computer and Information Sciences, Proceedings. IEEE, 2019. P. 710–713. DOI: 10.1109/SIBIRCON48586.2019.8958221.

8. *Fleming J., Yan X., Allison C., et al.* Real-time predictive eco-driving assistance considering road geometry and long-range radar measurements. IET Intelligent Transport Systems. 2021. Vol. 15, no. 4. P. 573–583. DOI: 10.1049/ITR2.12047.

9. *Scholl M., Minnerup K., Reiter C., et al.* Optimization of a thermal management system for battery electric vehicles. 14th International Conference on Ecological Vehicles and Renewable Energies, EVER 2019. IEEE, 2019. DOI: 10.1109/EVER.2019.8813657.

10. *Meisel S.* Dynamic Vehicle Routing. Anticipatory Optimization for Dynamic Decision Making. Operations Research/Computer Science Interfaces Series, vol. 51. New York, NY: Springer, 2011. P. 77–96. DOI: 10.1007/978-1-4614-0505-4_6.

11. *Cheng A.M.K.* Real-Time Scheduling and Schedulability Analysis. Real-Time Systems: Scheduling, Analysis, and Verification. John Wiley, Sons, 2002. P. 41–85. DOI: 10.1002/0471224626.CH3.

12. *Kopetz H.* Real-Time Scheduling. Real-Time Systems. Real-Time Systems Series. Boston, MA: Springer, 2011. P. 239–258. DOI: 10.1007/978-1-4419-8237-7_10.

13. *Ольховский Н.А., Соколинский Л.Б.* Визуальное представление многомерных задач линейного программирования // ВестникЮУрГУ. Серия: Вычислительная математика и информатика. 2022. Т. 11, № 1. С. 31–56. DOI: 10.14529/cmse220103.