Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
федерального государственного бюджетного образовательного учреждения высшего образования
**«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»**
(МИ ВлГУ)

Факультет информационных технологий и радиоэлектроники
Кафедра информационных систем

КУРСОВАЯ РАБОТА

по курсу Моделирование информационных систем

на тему: Моделирование аффинных преобразований над признаковым пространством

Руководитель
к. т. н., доц. каф. ИС
(уч. степень, звание)
Еремеев С. В.
(фамилия, инициалы)

(оценка)

(подпись) (дата)

Члены комиссии

(подпись) (Ф.И.О.)

(подпись) (Ф.И.О.)

Студент ИС - 122
(группа)
Клинцов С. М.
(фамилия, инициалы)

(подпись) (дата)

Муром 2025

В ходе курсовой работе было разработано приложение для применения различных аффинных преобразований к нейронным сетям:

– обучены сети различной сложности;

– реализован алгоритм применения аффинных преобразований к весам обученной модели;

– разработан интерфейс взаимодействия с пользователем;

– проведено тестирование и отладка проекта.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современное машинное обучение, особенно в области глубоких нейронных сетей, достигло невероятных успехов в решении сложных задач, от компьютерного зрения до обработки естественного языка. Однако эта мощь сопряжена с существенными вызовами, среди которых — вычислительная дороговизна дообучения, а также растущая потребность в управлении и интерпретации уже развёрнутых моделей. Обученная нейросетевая модель представляет собой сложную иерархическую структуру, где веса каждого слоя кодируют определённые паттерны в данных. Прямое манипулирование этими весами без потери качества предсказаний задача не из простых, так как даже незначительные изменения могут привести к искажению выходов.

В этом контексте концепция аффинных преобразований над признаковым пространством приобретает ключевое значение. Признаковое пространство — это абстрактное многомерное пространство, в котором данные представлены векторами признаков, извлечённых слоями нейронной сети. Аффинные преобразования (линейные трансформации, такие как поворот, масштабирование, сдвиг и их комбинации) являются фундаментальным инструментом геометрии и линейной алгебры. Их применение к признаковым пространствам обученных моделей открывает принципиально новые возможности: целенаправленное изменение внутренних представлений, коррекцию смещений (bias), сжатие или адаптацию модели к смещённым распределениям данных — и всё это без риска разрушения уже усвоенных знаний.

Таким образом, актуальность данной работы обусловлена необходимостью разработки эффективных и безопасных методов постобработки обученных моделей, которые позволили бы вносить в них управляемые изменения, сохраняя при этом исходную точность предсказаний.

Исследования в области манипуляций с весами и представлениями нейронных сетей ведутся, но носят зачастую фрагментарный характер. Классический подход к модификации модели – её дообучение на новых данных или с новыми целями. Однако он требует вычислительных ресурсов, исходных данных и изменяет архитектуру процесса обучения, что не всегда допустимо.

Однако комплексный подход, который бы ставил своей целью целенаправленное, контролируемое и обратимое аффинное преобразование весового пространства модели с гарантией сохранения точности на исходном задании, в литературе представлен недостаточно.

Объект исследования: обученные нейросетевые модели.

Целью данной работы является исследование возможности применения аффинных преобразований к признаковым пространствам обученной нейронной сети и разработка на этой основе алгоритма, позволяющего адаптировать веса модели к изменённым данным без изменения её выходной точности.

Для достижения поставленной цели необходимо решить следующие задачи:

– провести теоретический анализ структуры признаковых пространств глубоких нейронных сетей и свойств аффинных преобразований в контексте линейной алгебры;

– сформулировать строгие математические условия, при которых аффинное преобразование, применённое к весам одного или нескольких слоёв сети, не изменяет итоговое предсказание для измененных данных;

– разработать и формализовать алгоритм поиска и применения аффинных преобразований к весам полносвязных слоёв предобученной модели;

– реализовать спроектированный алгоритм на языке программирования Java с использованием библиотеки для глубокого обучения deeplearning4j;

– экспериментально проверить корректность работы алгоритма.

1. Анализ технического задания

Исходным заданием работы является реализация алгоритма, который позволит менять веса уже обученной нейросетевой модели без изменения точности её предсказаний, не прибегая к дообучению. Основными сложностями в реализации стали:

– обработка многослойных и многомерных современных сетевых моделей;

– тонкости применения аффинных преобразований к весам этих моделей.

Для решения этих задач потребовался тщательное изучение устройства нейронных сетей и выбор технологий, обеспечивающих высокую производительность, удобство разработки и корректную работу алгоритмов машинного обучения.

1.1 Языки программирования: Java, Python

Основным языком разработки выбран Java, что обусловлено его надёжностью, производительностью и доминирующим положением в промышленной разработке корпоративных систем. В контексте реализации алгоритмов манипуляции с весами обученных моделей использование Java обеспечивает строгую типизацию и высокую предсказуемость выполнения, что критически важно для обеспечения корректности и воспроизводимости преобразований. Хотя Python традиционно доминирует в сфере прототипирования и исследований машинного обучения благодаря лаконичному синтаксису и обширной экосистеме специализированных библиотек, для данной задачи приоритетными стали производительность во время выполнения, многопоточность и возможности низкоуровневой оптимизации, которые в полной мере предоставляет Java.

Для графического отображения данных был использован Python в паре с Matplotlib - библиотекой для визуализации данных двумерной и трёхмерной графикой.

1.2 Нейронные сети: Eclipse Deeplearning4J

В качестве основы для обучения модели, её анализа, проведения аффинных преобразований и реализации дополнительных вычислительных компонентов был выбран DL4J в связке с библиотекой ND4J. Этот стек представляет собой наиболее эффективное решение для глубокого обучения на платформе Java/JVM.

1.3 Среда разработки: VScode

Средой разработки стал Visual Studio Code - легковесный, но мощный редактор, с широкими возможностями кастомизации и поддержкой огромного множества языков. Его универсальность и производительность делают его оптимальным выбором для подобных проектов по сравнению с более специализированными IDE.

2. Разработка модели

Для всех экспериментов использовались исключительно полносвязные нейронные сети, что позволило сфокусироваться на свойствах преобразований без усложнений, присущих свёрточным или рекуррентным архитектурам.

2.1 Создание однослойной бинарной модели “Треугольник”

Первоначальной задачей стало построение максимально простой и интерпретируемой модели для дальнейшей визуализации работы нейронной сети и её внутреннего представления.

Была разработана однослойная нейронная сеть с архитектурой 2 – 2 – 1 (точка x, y на вход, два нейрона в скрытом слое, один выходной нейрон). Для датасета программно генерировался сбалансированный набор точек так, чтобы данные формировали визуальный треугольник на графике, который потом должны были ограничивать линии нейронов скрытого слоя.

Модель решает бинарную задачу классификации: определение принадлежности точки с координатами (x, y) заданной геометрической области - треугольнику.

Обучение модели продолжалось до сходимости, что визуально выражалось в стабилизации значений функции потерь. Финальное значение accuracy было примерно 96%.

2.2 Визуализация модели

В общем случае получение значения активации нейрона на каждом слое полносвязной сети можно описать следующим уравнением:

$$\sigma(w_1 a_1 + w_2 a_2 + \cdots + w_n a_n + b) = a \text{ , где} \qquad (1)$$

σ – функция активации;

w_n – вес n-го нейрона прошлого слоя;

a_n – активация нейрона прошлого слоя;

b – смещение;

a – новое значение активации нейрона.

Вспомним как выглядит уравнение линии на двумерной плоскости:

$$ax + by + c = 0$$

Так становиться видно, что уравнение высчитывания новой активации нейрона имеет некоторую схожесть с геометрическим уравнением линии. Сходство становиться очевиднее для слоя с двумя нейронами. Отбросим применение функции активации, оставив только сумму произведений весов на активации прошлых нейронов:

$$w_1 a_1 + w_2 a_2 + b$$

Этот факт даёт нам возможность манипулировать этими данными, применяя к ним различные аффинные преобразования.

По этому уравнению мы и сможем построить линии сети, ограничивающие область решений, так как веса и активации – это константы, которые получаются после обучения.

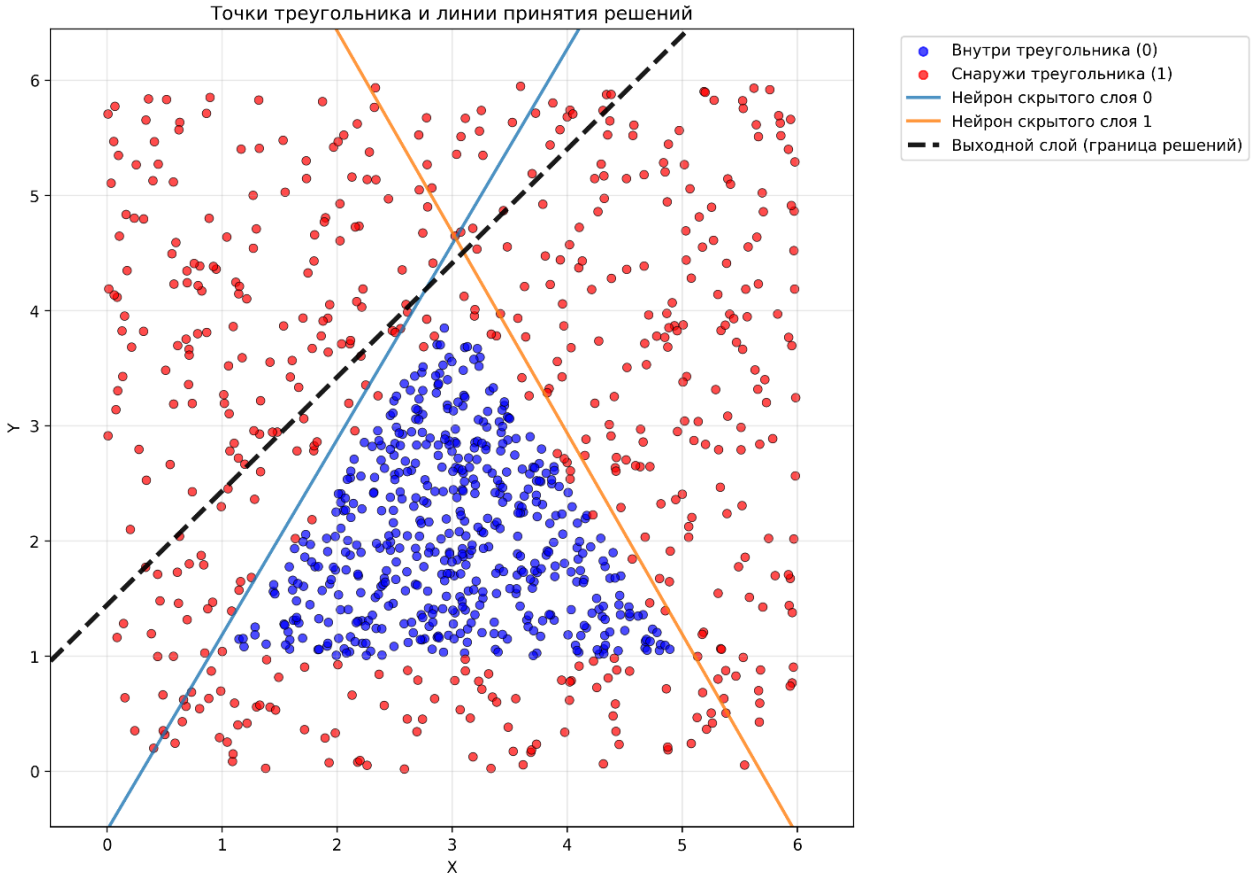

Рисунок 1 - Точки датасета "Треугольник" и ограничивающие линии нейронов

На рисунке 1 видно, как две линии скрытого слоя выделяют треугольник. Третья линия – это выходной слой, который оказывает решающее влияние в реальной работе модели. Можно заметь, что эта линия никак не ограничивает область треугольника. Это происходит, потому что её активация высчитывается по активации нейронов предыдущего слоя, что даёт неожиданный результат при визуализации. Зная это, в дальнейшем будем работать, только с весами и смещениями первого скрытого слоя, который работает непосредственно со входными данными.

2.2.1 Отображение нормалей для линий нейронов

Вспомним, что для каждого нейрона скрытого слоя с весами w_1, w_2 и смещением b уравнение разделяющей линии в пространстве признаков имеет вид:

$$w_1 x + w_2 y + b = 0 \qquad (2)$$

Вектор n = $[w_1, w_2]$ является нормалью (перпендикуляром) к данной линии, что следует из свойств скалярного произведения: если скалярное произведение двух ненулевых векторов равно нулю, то эти векторы перпендикулярны. Действительно, для любых двух точек (x_1, y_1) и (x_2, y_2), лежащих на линии $w_1(x_2 - x_1) + w_2(y_2 - y_1) = 0$.

Таким образом, вектор весов нейрона определяет не только положение разделяющей линии, но и её ориентацию в пространстве.

Зная направление нормали линии нейрона, мы сможем узнать, где находиться точка, передавая значения её координат.

Если результат уравнения $w_1x + w_2y + b > 0$, то нормаль указывает в сторону искомой области. Если $w_1x + w_2y + b < 0$, то область находиться в противоположном направлении от неё.

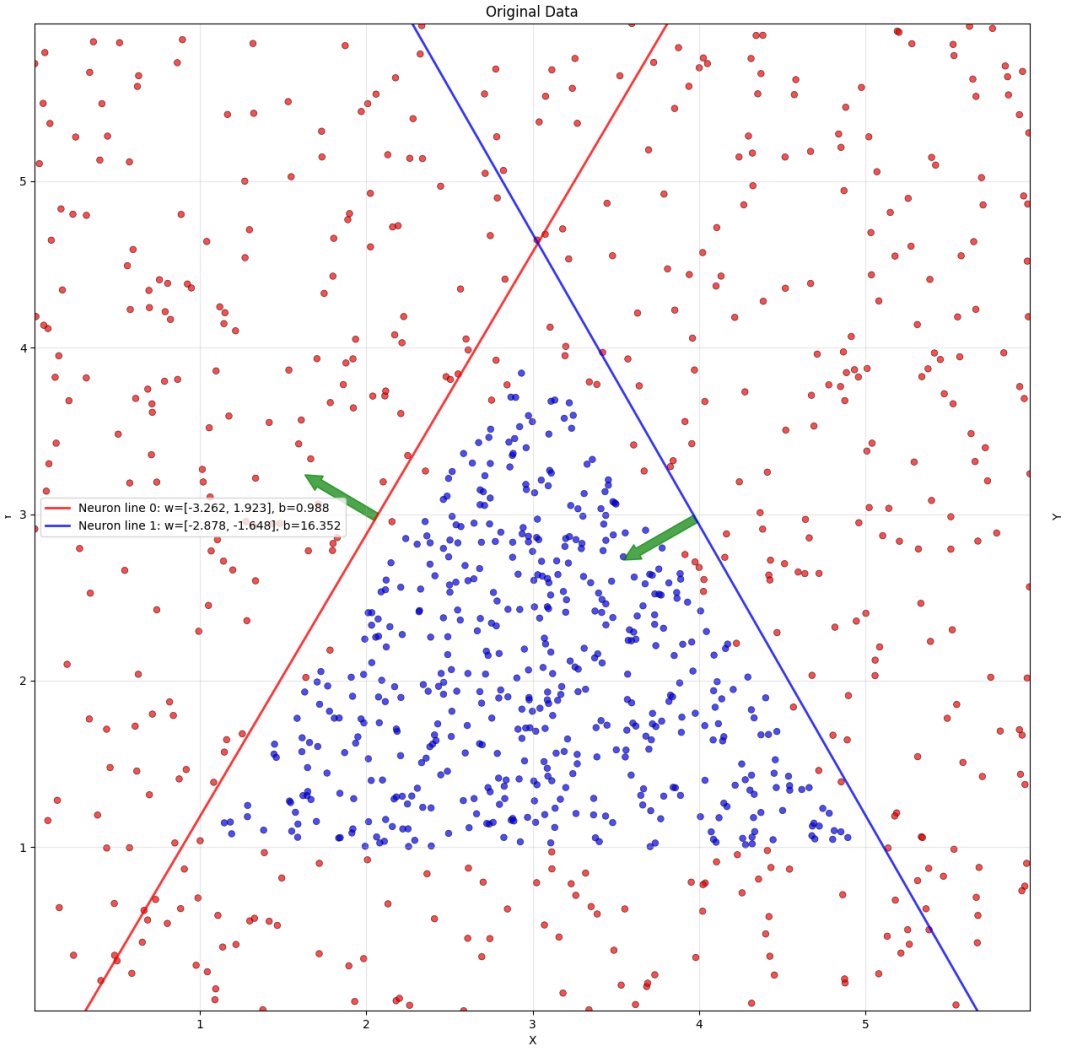

Рисунок 2 - Обновлённое отображение модели “Треугольник” с нормалями

Чтобы убедиться, что всё работает, согласно ожиданиям, проведём некоторые вычисления для конкретной точки.

Возьмём точку с $x = 3, y = 3$. Визуально можно оценить, что точка с данными координатами входит в область решений. Докажем это расчётами.

Подставляем в формулу 2 известные значения.

Для линии 0 (красная) это будет:

```
-3.26*3.0 + 1.92*3.0 + 0.98 = -3.04
```

Результат получился отрицательным. Точка находится противоположно направлению нормали.

Для линии 1 (синяя) это будет:

```
-2.87*3.0 - 1.64*3.0 + 16.35 = 2.82
```

Результат получился положительным. Точка находится по направлению нормали.

Комбинируя результаты, можно сказать, что точка действительно входит в область решений, так как находиться между линиями нейронов.

2.3 Знакомство с аффинными преобразованиями

Аффинное преобразование (от лат. affinis «соприкасающийся, близкий, смежный») — отображение плоскости или пространства в себя, при котором параллельные прямые переходят в параллельные прямые, пересекающиеся — в пересекающиеся, скрещивающиеся — в скрещивающиеся.

«Отображение в себя» означает, что если мы находились в пространстве Rn, то после образования мы должны остаться в нем же. Например: если мы применили какое-то преобразование к прямоугольнику и получили параллелепипед, то мы вышли из R2 в R3. А вот если из прямоугольника у нас получился другой прямоугольник, то все хорошо, мы отобразили исходное пространство в себя.

Теперь запишем в общем виде, как выглядит преобразование координат в формульном виде.

Пусть у нас есть исходная система координат. Точка в этой системе характеризуется двумя числами - x и y. Совершить переход к новым координатам x' и y' мы можем с помощью следующей системы:

$$\begin{cases} x' = ax + \beta y + \vartheta \\ y' = \gamma x + \sigma y + \mu \end{cases}$$

При этом, числа a, β, γ, σ должны образовывать невырожденную матрицу:

$$\begin{pmatrix} a & \beta \\ \gamma & \sigma \end{pmatrix}$$

Можно записать и в более общем виде, где аффинное преобразование – это преобразование вида:

$$M * v, \text{ где} \qquad (3)$$

M - аффинная матрица, а v - n-мерный вектор.

Для дальнейшей реализации было выбрано 3 аффинных преобразования: поворот, растяжение/сжатие и сдвиг.

2.4 Аффинное преобразование поворота

Для первой экспериментальной проверки концепции модификации весов без дообучения было выбрано аффинное преобразование поворота в пространстве.

Матрица M для поворота против часовой стрелки примет вид:

$$\begin{pmatrix} \cos(a) & \sin(a) \\ -\sin(a) & \cos(a) \end{pmatrix} \qquad (4)$$

И новые координаты будут выглядеть так:

$$\begin{cases} x' = xcos(a) + ysin(a) \\ y' = -xsin(a) + \mathrm{ycos}(a) \end{cases}$$

Таким образом получаем повёрнутую систему координат на угол a.

Для поворота точки остаётся вычислить:

$$\begin{pmatrix} \cos(a) & \sin(a) \\ -\sin(a) & \cos(a) \end{pmatrix} * \begin{pmatrix} x \\ y \end{pmatrix}$$

2.4.1 Обобщение алгоритма для n-мерной точки

Для заключительного эксперимента будет обучена модель на базе датасета чисел MNIST, каждое из которых имеет размер 28*28 пикселей, что в результате даёт 784. Нейроны на первом скрытом слое будут размерностью 784, так как должны будут обрабатывать входные 784 пикселя. Таким образом сложные многослойные сети могут состоять из нейронов, которые формируют гиперплоскости n-мерной размерности. Для обобщения матрицы поворота для n-мерного случая, нужно понять, как она формируется. Рассмотрим на примере 2-мерного случая.

Сначала создаётся пустая единичная квадратная матрица n*n. Для двумерного случая мы имеем точку с координатами (осями) x, y. Для заполнения значениями матрицы поворота, мы будем брать комбинации осей x, y для определения плоскости вращения. xx = cos, xy = -sin, yx = sin, yy = cos. Получаем формулу 4.

Зная алгоритм для двумерной системы, можно применить его для того, чтобы повернуть точку в n-мерном пространстве вокруг гиперплоскости, заданной двумя осями axis1 и axis2. Выбираем 2 оси из n и вращаем в плоскости, которую они образуют.

Пример получения обобщенной матрицы преобразования на Java:

```
    private double[][] createAffineMatrix(int dimensions, int axis1, int axis2)
{
        double[][] matrix = createIdentityMatrix(dimensions);

        double cos = Math.cos(angleDegrees);
        double sin = Math.sin(angleDegrees);

        matrix[axis1][axis1] = cos;
        matrix[axis1][axis2] = -sin;
        matrix[axis2][axis1] = sin;
        matrix[axis2][axis2] = cos;
```

```
        return matrix;
    }
```

2.4.2 Визуализация модели сети с применением поворота

Преобразования будем применять к скрытым слоям сети. В модели “Треугольник” один скрытый слой, который состоит из 2 нейронов. Вспомним, что каждый нейрон – это прямая, если говорить про двумерные случаи или гиперплоскости, если говорить о n-мерных случаях.

Так как смещение не зависит от входных координат, а веса — зависят, то при применении преобразования к линии нейрона, будем изменять только веса. Корректность данного вывода подтверждается эмпирически. При изменении не только весов, но и смещений, полностью нарушается архитектура модели, и итоговые результаты различны.

Воспользуемся формулой 3: для весов формулы 2 применим преобразование поворота, формула 4.

Для моделирования повёрнутых данных те же преобразования применим и к ним.

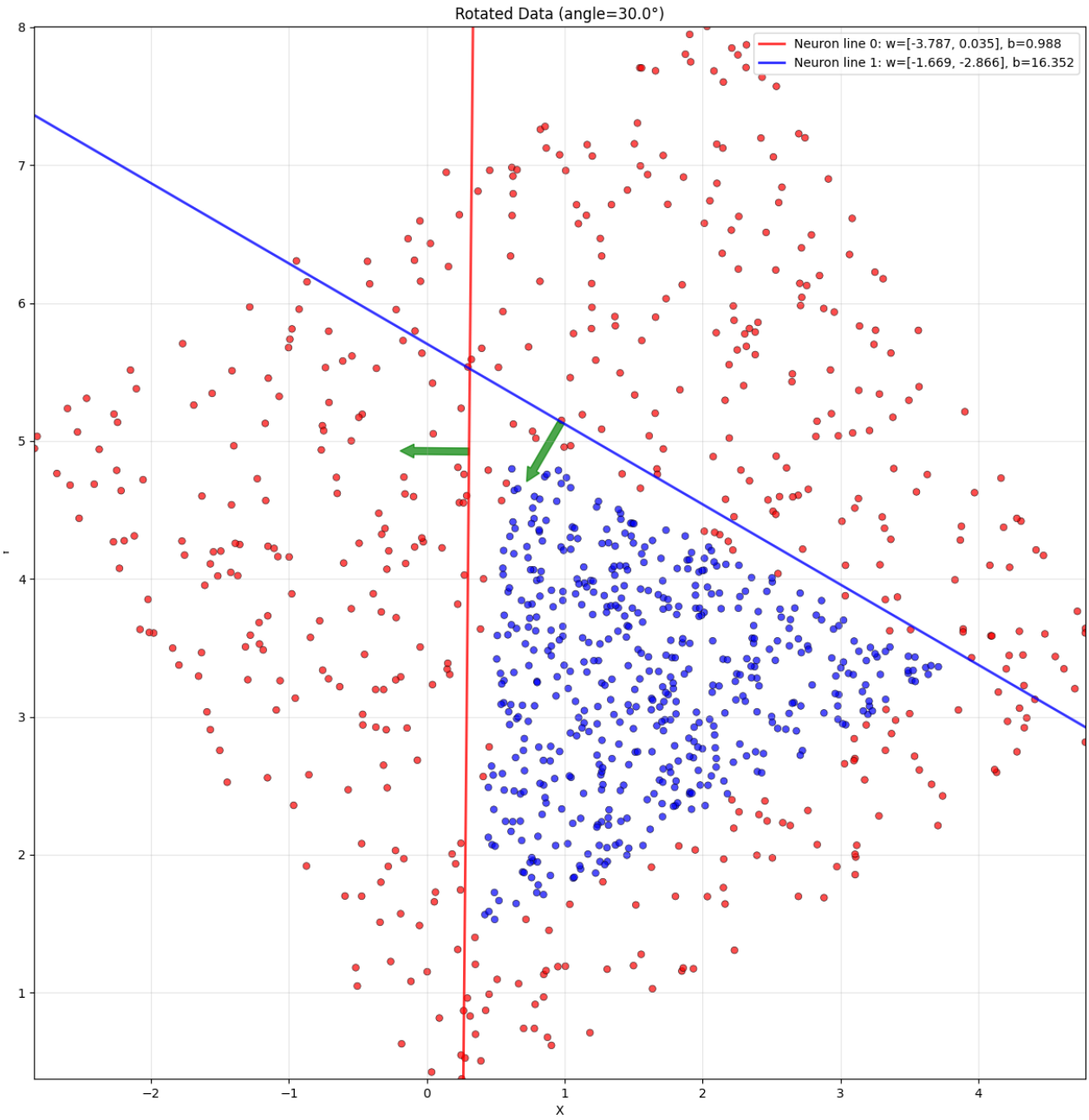

Рисунок 3 - Модель “Треугольник”, после применения преобразования поворота на 30 градусов

Веса сети поменялись, по сравнению с моделью на рисунке 2, в соответствии с изменёнными данными. Это позволяет модели предсказывать отношение точки к области решений с неизменной точностью.

Например, confidence для точки 3, 3 до манипуляций с весами и данными: 0.87179654, и confidence после поворота весов и данных также 0.87179654

2.5 Аффинное преобразование растяжение/сжатие

Для моделирования растяжения/сжатия сконструируем матрицу M из формулы 3 несколько иначе:

$$\begin{pmatrix} k_x & 0 \\ 0 & k_y \end{pmatrix}$$

Новые координаты тогда принимают вид:

$$\begin{cases} x' = x * k_x \\ y' = \mathrm{y} * k_y \end{cases}$$

Из вида системы уравнений понятно, что мы просто растягиваем наши оси, если коэффициент меньше 1 и сжимаем, если больше 1.

Для n-мерного случая адаптировать матрицу также не составляет труда, просто ставим нужный коэффициент на тот элемент диагонали, ось которой хотим растянуть/сжать.

Пример получения обобщенной матрицы преобразования на Java:

```
public double[][] createAffineMatrix(int dimensions) {
    double[][] matrix = createIdentityMatrix(dimensions);

    for (int i = 0; i < dimensions; i++)
        matrix[i][i] = scaleFactor;

    return matrix;
}
```

2.5.1 Ортогональные и не ортогональные матрицы аффинных преобразований

Для корректного применения преобразований к весам модели, нужно учитывать один нюанс, а именно ортогональность полученной аффинной матрицы.

Ортогональность простыми словами означает перпендикулярность или независимость между объектами. Это значит, что они находятся под прямым углом друг к другу, как оси координат (X и Y) на графике.

Чтобы узнать, что перед нами ортогональная матрица нужно выяснить:

– является ли она квадратной;

– выполняется одно из условий:

либо $A * A^T = E$, где A^T — транспонированная, а E — единичная матрица, либо обратная матрица A^{-1} совпадает с транспонированной A^T.

Знание того факта, что аффинная матрица ортогональна, подскажет нам, нужно ли компенсировать веса перед применением преобразования или нет.

Компенсировать – значит применять к матрице весов обратную матрицу аффинного преобразования, с целью корректного наложения изменённых весов на данные.

Простая аналогия, для лучшего понимания вышеописанного.

При повороте картинки на 30 градусов против часовой стрелки, требует от наблюдателя “повернуть” свои глаза также на 30 градусов против часовой, чтобы увидеть такую же картинку. Но, например, при растяжении изображения в 5 раз, если наблюдатель также “увеличит” глаза в 5 раз, то он просто ничего не увидит. В данном преобразовании, глаза наблюдателя нужно пропорционально “уменьшить” в 5 раз, чтобы увидеть ту же картинку.

На этой аналогии можно понять, что аффинная матрица поворота – ортогональна, а матрица расширения – нет.

Докажем это кодом на Python:

```
import numpy as np

angle = 30.3
angle_rad = np.radians(angle)
rotate = np.array([
    [np.cos(angle_rad), -np.sin(angle_rad)],
    [np.sin(angle_rad), np.cos(angle_rad)]
])

print(rotate.T)
print()
print(np.linalg.inv(rotate))
print()
```

```
print(rotate @ rotate.T)

print()
print()

scale_x = 5
scale_y = 5
scale = np.array([
    [scale_x, 0],
    [0, scale_y]
])

print(scale.T)
print()
print(np.linalg.inv(scale))
print()
print(scale @ scale.T)
```

Результат:

```
[[ 0.86339555  0.50452762]
 [-0.50452762  0.86339555]]


[[ 0.86339555  0.50452762]
 [-0.50452762  0.86339555]]
# матрицы равны
[[ 1.00000000e+00 -2.23418993e-17]
 [-2.23418993e-17  1.00000000e+00]]
# получили единичную матрицу
# значит матрица - ортогональна


[[5 0]
 [0 5]]


[[0.2 0. ]
```

```
 [0.  0.2]]
# матрицы не равны
[[25  0]
 [ 0 25]]
# получили не единичную матрицу
# значит матрица – не ортогональна
```

Таким образом, в преобразовании растяжения/сжатия для изменения весов будем применять обратную аффинную матрицу.

2.5.2 Визуализация модели сети с применением растяжения/сжатия

Условия аналогичны случаю с поворотом, только к весам применяем обратную аффинную матрицу.

Для примера сделаем сжатие в 2 раза:

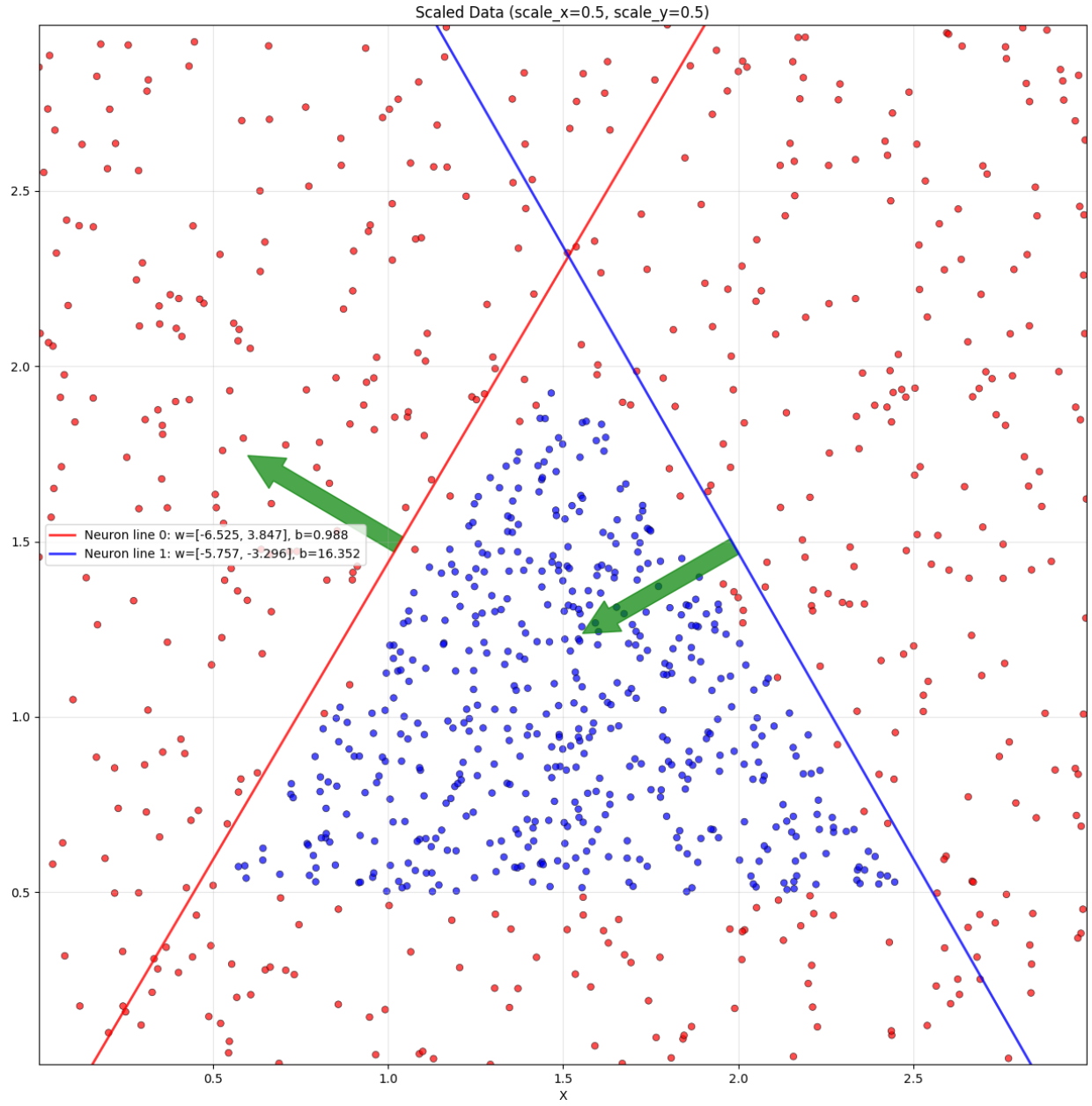

Рисунок 4 - Модель “Треугольник”, после применения преобразования сжатия в 2 раза

Веса сети также поменялись, в соответствии с изменёнными данными. Результат предсказаний не изменился.

2.6 Аффинное преобразование расширения

Для моделирования расширения сконструируем матрицу M из формулы 3 следующим образом:

$$\begin{pmatrix} 0 & k_{xy} \\ k_{yx} & 0 \end{pmatrix}$$

Новые координаты принимают вид:

$$\begin{cases} x' = x + k_{xy} * y \\ y' = k_{yx} * x + y \end{cases}$$

Из вида системы уравнений понятно, что мы растягиваем наши координаты вправо и вверх для x' и в противоположную сторону – вниз и влево для y'.

Для n-мерного случая это обобщается следующим образом: мы применяем расширение по координате относительно другой координаты. А значит мы делаем тоже самое что и для поворота: вставляем коэффициент расширения относительно требуемых координат, но оставляя диагональ нетронутой.

Пример получения обобщенной матрицы преобразования на Java:

```
    private double[][] createAffineMatrix(int dimensions, int axis1, int axis2)
{
        double[][] matrix = createIdentityMatrix(dimensions);

        matrix[axis1][axis2] = shear;
        matrix[axis2][axis1] = shear;

        return matrix;
    }
```

Если сравнить все три преобразование, можно сделать интересный вывод: поворот осуществляется одновременным применением растяжения и сжатия, используя для этого тригонометрию. Это особенно хорошо видно, при сравнении кода для получения матриц каждого из преобразований.

2.6.1 Визуализация модели сети с применением расширения

Для корректной компенсации расширения, будем делать по аналогии с преобразованием растяжения/сжатия – к весам применяем обратную аффинную матрицу.

Для примера сделаем расширение на коэффициент 0.3:

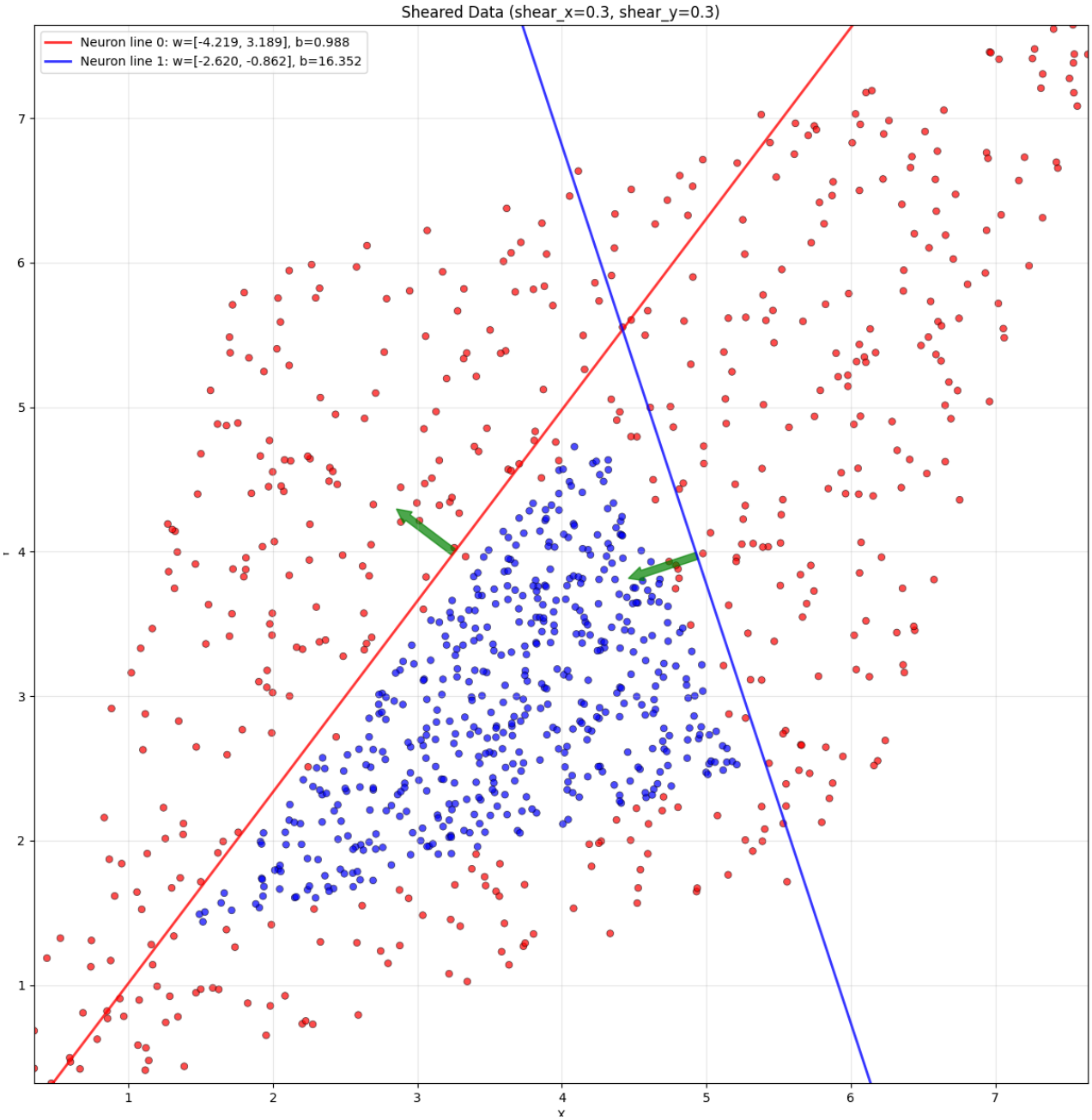

Рисунок 5 - Модель “Треугольник”, после применения преобразования расширение на коэффициент 0.3

2.7 Создание многослойной бинарной модели “Два треугольника”

После проведения всех экспериментов с преобразованиями на однослойной модели “Треугольник”, нужно выяснить, будет ли всё работать также хорошо и для многослойных моделей.

Для этого усложним текущую модель, добавив ещё один треугольник. Она всё ещё будет бинарной, то есть на выходе мы будем узнавать принадлежит ли

точка одному из треугольников или нет. Для чёткого разделения этих треугольников нужно усложнить архитектуру модели: во-первых, увеличить количество нейронов на первом слое, по 3 на каждый треугольник и, во-вторых, добавить второй скрытый слой, который будет разделять области решений двух треугольников. Итоговая архитектура: 2 – 6 – 2 – 1.

2.7.1 Обоснование выбора слоя для применения преобразований

Аффинные преобразования будут применяться только к первому скрытому слою. Или можно сказать иначе, только к слою, который непосредственно работает с входными данными.

Чтобы понять, почему нужно делать, так, а не иначе, нужно вспомнить как вообще работают сети, основанные на полносвязных (dense) слоях.

Каждый полносвязный слой в нейронной сети выполняет преобразование по формуле 1. Из неё видно, что активация нейрона вычисляется по значению активации предыдущего нейрона, кроме первого слоя. Он работает непосредственно с исходными данными. Отсюда можно сделать вывод, что если при применении преобразования, первый скрытый слой выдаст тот же результат, как если бы этих преобразований не было, то все остальные слои автоматически сработают правильно и результат будет корректным.

Что бы это доказать, нужно сравнить активации нейронов первых скрытых слоёв до всех преобразований и после изменения весов и входных данных. Применим преобразование поворота на 30 градусов к модели “Два треугольника”, со входной точкой 3, 3:

```
====== Before weight and data transformation ======
------ First hiden layer weights ------
-0.2565350830554962 0.9576932787895203
0.27009227871894836 0.1462019383907318
-0.019547274336218834 -0.04933701455593109
1.410359501838684 -1.0748653411865234
0.05246511474251747 -2.7907469272613525
-0.7326761484146118 -0.7785971164703369
```

```
------ Neuron activations ------
3.0 3.0

-0.3422616422176361         -0.9828152060508728         -0.9952481389045715
0.775757372379303 -0.9999905228614807 0.49449196457862854

-0.9999547600746155 -0.7523653507232666

0.985088050365448

------ Prediction ------
Confidence: 0.985088050365448


====== After weight and data transformation ======
------ First hiden layer weights ------
0.7476058476903883 0.6512082321163086
0.24999965462541984 -0.1783956796000739
-0.05290834426082423 -0.004116314716547248
-0.3459362576915701 -1.7391887567843332
-2.4839285710086605 -1.2732320504960983
-1.0213258509573662 0.3161033286724542

------ Neuron activations ------
4.012990772626067 -1.3769186827180608

-0.3422614336013794         -0.9828152060508728         -0.9952481389045715
0.7757572531700134 -0.9999905228614807 0.4944923520088196

-0.9999547600746155 -0.75236576795578

0.9850881695747375

------ Prediction ------
Confidence: 0.9850881695747375
```

На полученных данных, можно увидеть, как активации нейронов первого скрытого слоя до и после преобразований равны до 5-6 знаков после запятой. А это значит, что следующие слои, используя эти данные, не нарушат результат предсказания.

2.7 Визуализация применения аффинных преобразований к модели “Два треугольника”

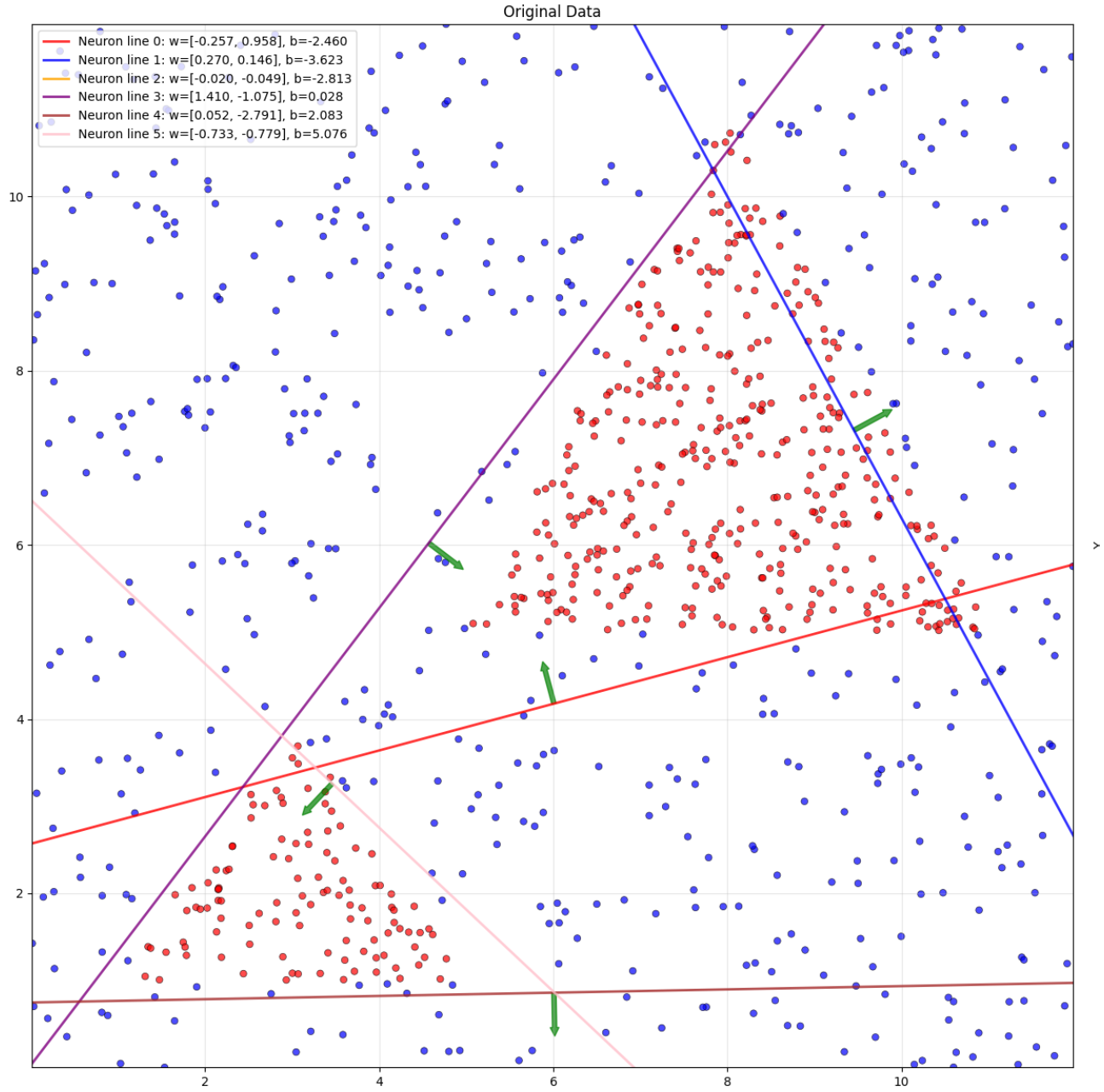

Рисунок 6 - Модель “Два треугольника”, до применения каких-либо преобразований

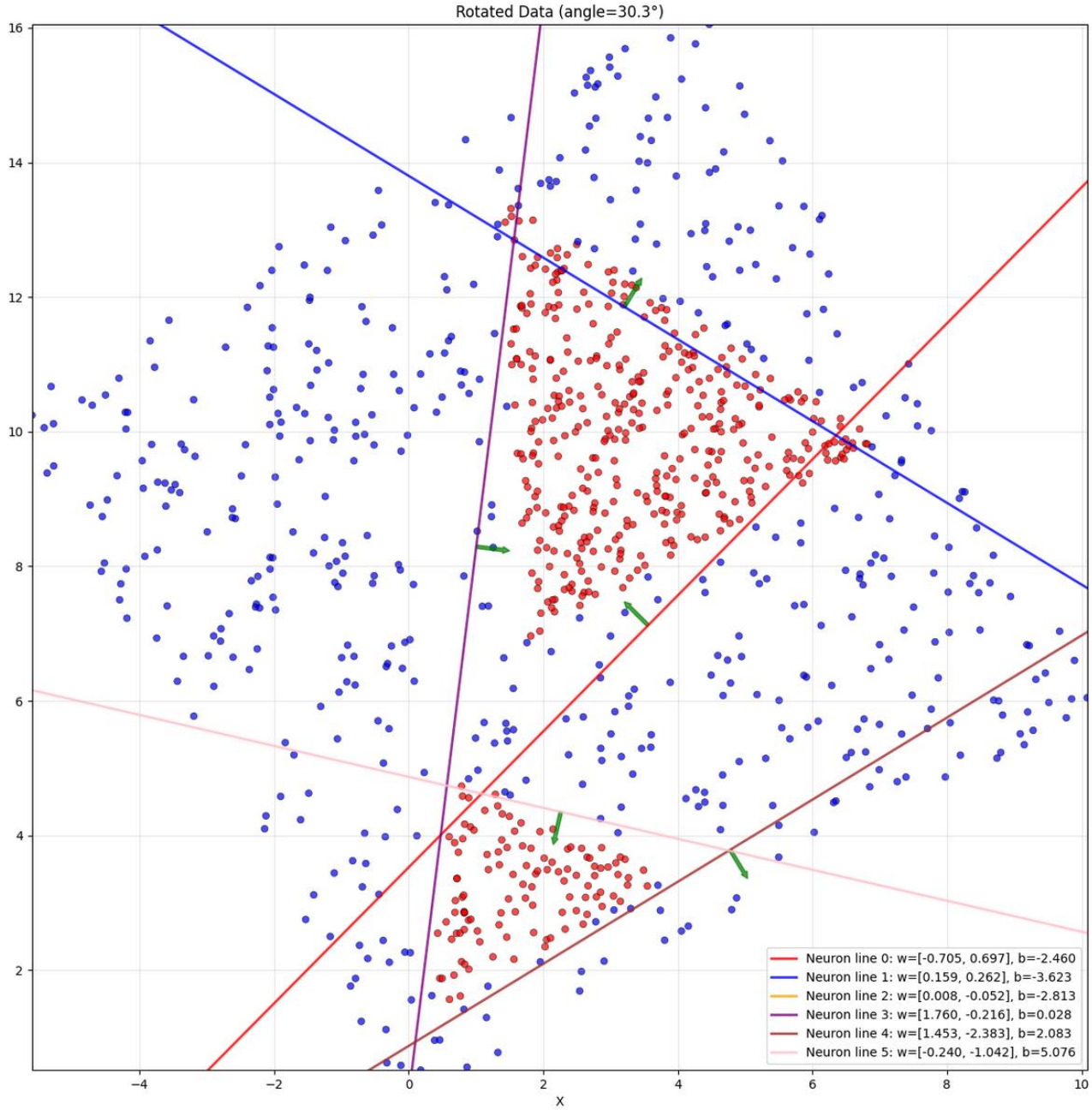

Рисунок 7 - Модель “Два треугольника”, после применения преобразования поворота на 30 градусов

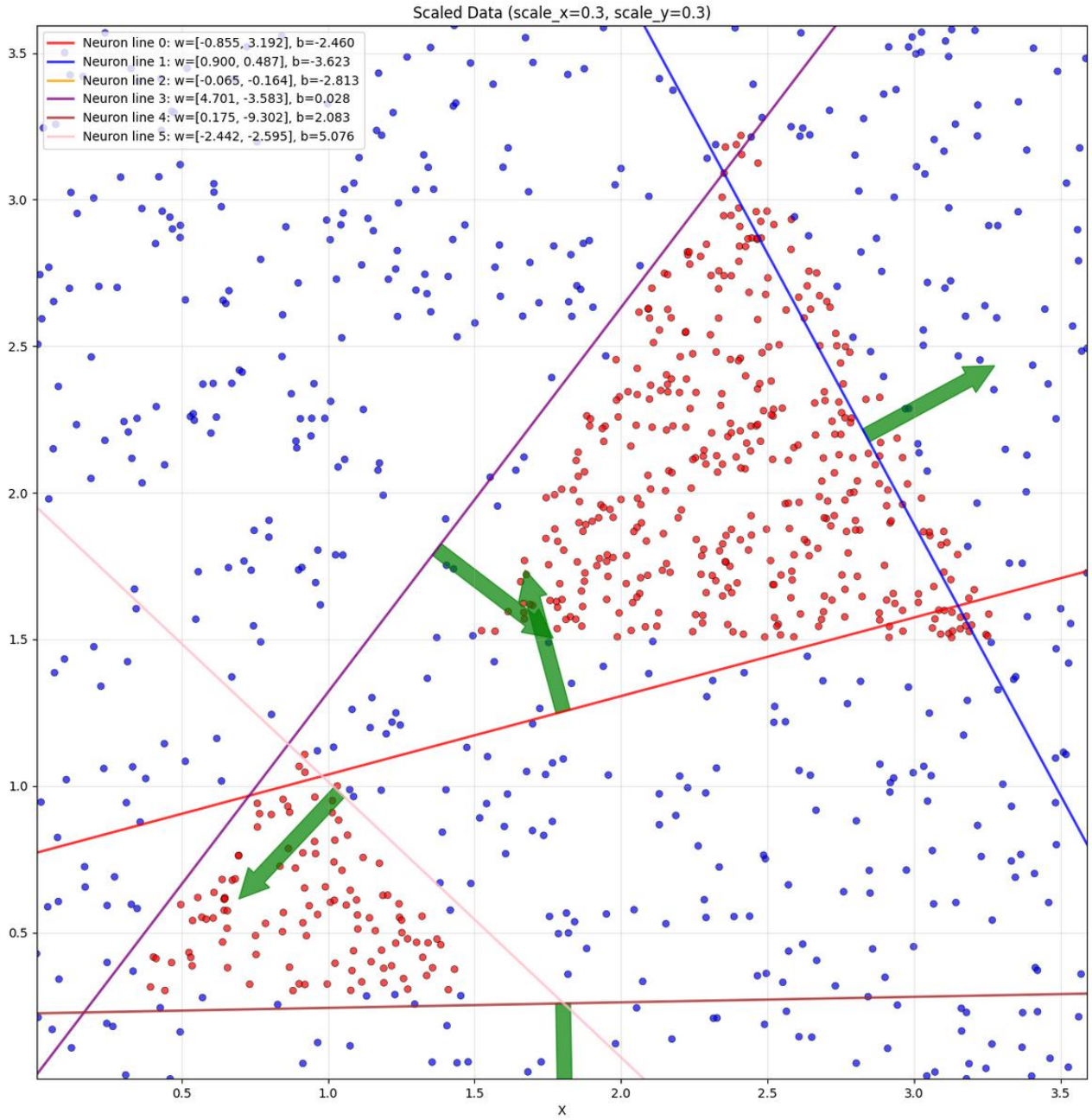

Рисунок 8 - Модель “Два треугольника”, после применения преобразования сжатия на коэффициент 0.3

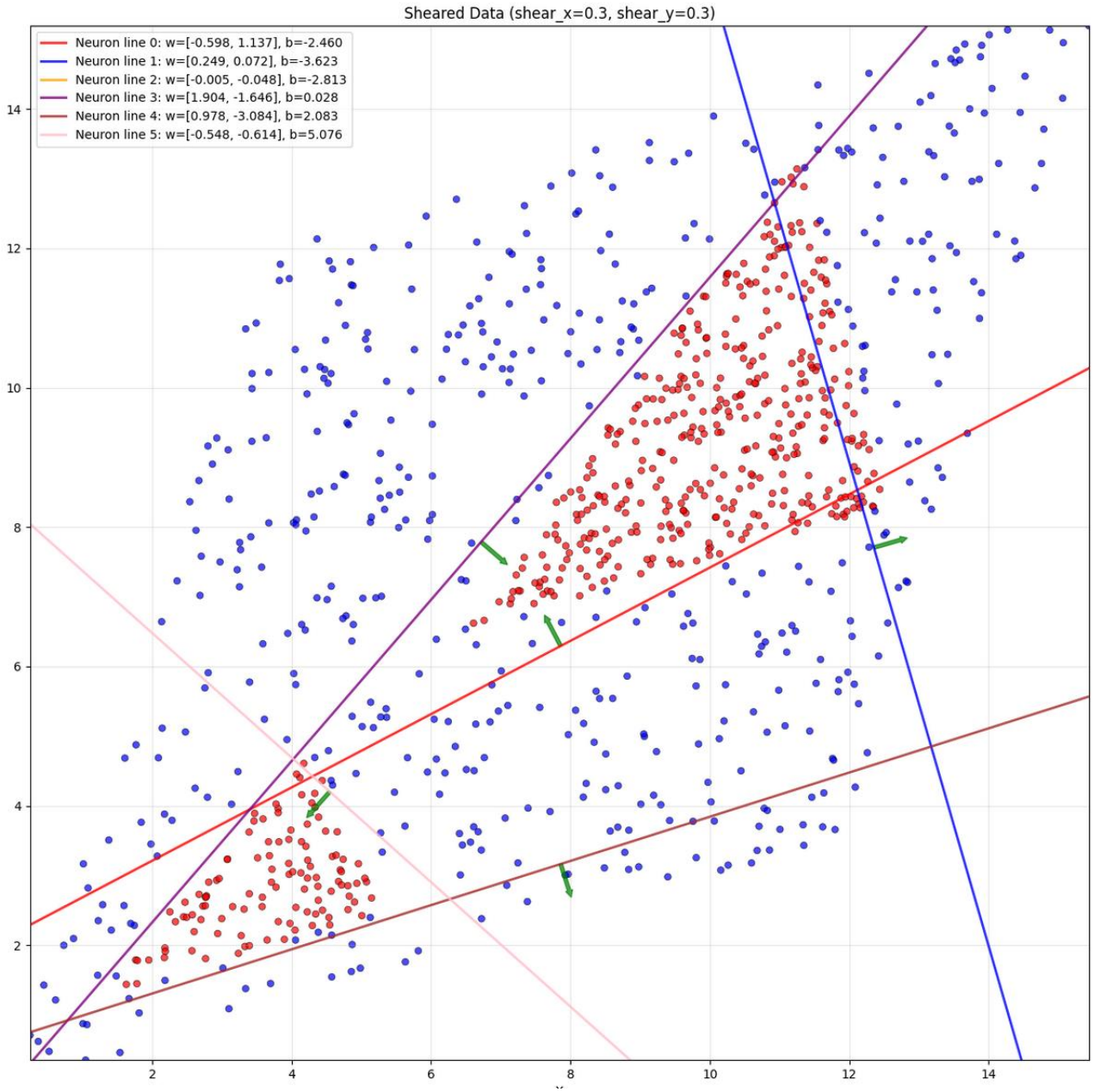

Рисунок 9 - Модель “Два треугольника”, после применения преобразования расширение на коэффициент 0.3

2.8 Создание многослойной модели “MNIST”

Прежде все модели получали на вход двумерную точку, с координатами x, y. Это позволило нам визуализировать результаты. Но теперь, имея достаточно информации и экспериментальных данных, можно расширить применение преобразований до n-мерных данных.

Для этого заключительной экспериментальной моделью послужит нейронная сеть на базе датасета чисел “MNIST”, где каждая цифра от 0-9 представлена изображением 28*28 пикселей. Её архитектура: 784 – 64 – 32 – 16 – 10.

Каждый нейрон на входном слое будет обрабатывать 784 пикселя входного изображения, или в геометрическом смысле он будет представлять собой гиперплоскость размерностью 784.

Имея обобщённые схемы построения матриц преобразований, не составит труда применить эти преобразования к модели и данным.

Для лучшей наглядности, будем применять преобразования ко всем осям последовательно. Так для матриц поворот между axis1 и axis2, результирующей будет все последовательно перемноженные матрицы поворотов. Похожим образом поступим и с другими преобразованиями.

Результаты после поворота изображения цифры 3 и весов первого скрытого слоя:

```
      ====== Before weight and data transformation ======
      ------ First hiden layer weights ------
      5.7705078397325956E-39   4.703835244793152E-39   -8.87386489726088E-38   -
6.339193992912607E-41        5.253545014169277E-39        1.5848125112127951E-40
3.264153814232014E-39 ...
      1.6218250075510065E-39   5.218488730487263E-39   5.335082367912945E-39   -
1.9093742550273876E-39     3.201489148205872E-39     6.238678854066588E-38     -
6.376859494335194E-39 ...
      -1.4172003992979751E-39  -4.3820831043995306E-39  -6.025940447445423E-38  -
6.51858822231547E-39      6.97999544922186E-38      2.940001249569004E-39      -
3.0647378323708777E-40 ...
      ...
      ------ Neuron activations ------
      ... 241.0 255.0 255.0 255.0 255.0 255.0 244.0 237.0 128.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 50.0 235.0 255.0 245.0 254.0 246.0
245.0 245.0 245.0 248.0 ...

      232.97018432617188    627.9869995117188    0.0    0.0    1112.656982421875
629.7344970703125  0.0  0.0  261.0110168457031  879.9109497070312  278.156005859375
209.40101623535156 ...
```

```
      1049.260009765625   951.4986572265625   836.6201171875   0.0   1646.404296875
20.05770492553711   0.0   1008.3165283203125   94.4949722290039   604.4305419921875
176.78518676757812 ...

      109.54348754882812      1440.6514892578125      915.6256713867188      0.0
1968.618896484375     3928.3076171875     152.3181610107422     375.6053771972656
1093.341552734375 ...

      0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0

      ------ Prediction ------
      Predicted digit: 3
      Confidence: 1.0

      ====== After weight and data transformation ======

      ------ First hiden layer weights ------
      1.839428899626901E-4        8.9961941361909E-5        1.0014484522300589E-4
1.1148036461767837E-4          1.240989655295359E-4          1.381458815488908E-4
1.537827854364992E-4 ...
      2.1830041127386877E-4       1.067653596302841E-4       1.1885026327233648E-4
1.3230307216514982E-4         1.4727862120277814E-4         1.6394927123321212E-4
1.825068928428718E-4 ...
      -4.259201622248689E-4   -2.0830707110614785E-4   -2.3188560716871792E-4   -
2.581330270089811E-4      -2.873514248960607E-4      -3.198770895245563E-4      -
3.5608437452402394E-4 ...
      ...

      ------ Neuron activations ------
      ...    6.764818237218361E-8    3.308506143967021E-8    3.6830000630381116E-8
4.099883413870403E-8          4.56395430888585E-8          5.0805539648099246E-8
5.655628177322188E-8 6.295795754098908E-8 ...

      232.97021484375     627.9871826171875     0.0     0.0     1112.6568603515625
629.734619140625  0.0  0.0  261.0109558105469  879.9109497070312  278.1557312011719
209.4010467529297 ...

      1049.260498046875   951.4990234375   836.6199951171875   0.0   1646.404296875
20.057880401611328   0.0   1008.316650390625   94.494873046875   604.4302978515625
176.78517150878906 ...
```

```
    109.54325103759766 1440.6514892578125 915.62548828125 0.0 1968.619384765625
3928.30810546875 152.3181610107422 375.60498046875 1093.342041015625 ...

    0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0

    ------ Prediction ------
    Predicted digit: 3
    Confidence: 1.0
```

Этот эксперимент показал подобные результаты с экспериментом с моделью “Два треугольника”. Активации нейронов до преобразований также примерно равны активациям нейронов после преобразований, с погрешностью в несколько знаков после запятой.

3. Исследование работы модели

Для исследования того, что моделирование аффинных преобразований над признаковым пространством выполняется корректно и соответствует поставленной задаче: модифицирование весов модели без изменения её выходной точности на валидационном множестве исходной задачи, выполним ряд экспериментов. За основу возьмём две последние многослойные модели “Два треугольника” и “MNIST”.

Для модели “Два треугольника” – 50 экспериментов для различных точек x, y. Каждый эксперимент будет состоять из:

– значений принадлежности точки к одному из треугольников (prediction) и уверенности (confidence) до преобразований;

– значений принадлежности точки к одному из треугольников (prediction) и уверенности (confidence) после преобразований только точки;

– значений принадлежности точки к одному из треугольников (prediction) и уверенности (confidence) после преобразований только весов;

– значений принадлежности точки к одному из треугольников (prediction) и уверенности (confidence) после преобразований точки и весов одновременно.

– для значений до и после вычисляется соответствие результатов и возможные изменения между ними.

Подробные результаты см. в приложениях A, C, E.

Для MNIST-а это будет по 5 штук на каждую цифру от 0 – 9. Каждый эксперимент будет состоять из:

– значений предсказанной цифры (prediction) и уверенности (confidence) до преобразований;

– значений предсказанной цифры (prediction) и уверенности (confidence) после преобразований только изображения;

– значений предсказанной цифры (prediction) и уверенности (confidence) после преобразований только весов;

– значений предсказанной цифры (prediction) и уверенности (confidence) после преобразований изображения и весов одновременно;

– для значений до и после вычисляется соответствие результатов и возможные изменения между ними.

Подробные результаты см. в приложениях B, D, F.

3.1 Анализ результатов экспериментов

Из результатов, представленных в таблицах Приложений A-F, следует, что все проведённые эксперименты подтверждают теоретические ожидания.

Полное совпадение точности при согласованных преобразованиях. Когда аффинное преобразование применяется одновременно и к входным данным, и к весам первого слоя модели, точность классификации остаётся неизменной. Предсказания модели до и после таких преобразований полностью идентичны.

Снижение точности при частичных преобразованиях. Если преобразование применяется только к данным или только к весам, точность модели существенно падает. Это особенно заметно на более простой архитектуре "Два треугольника", где малое количество нейронов делает сеть чувствительной даже к небольшим рассогласованиям между данными и весами.

Влияние сложности модели. На сложной модели "MNIST" нарушения точности при частичных преобразованиях также наблюдаются, но наиболее выражены они при преобразовании поворота. Это связано с тем, что поворот сильнее нарушает пространственные соотношения признаков, важных для распознавания рукописных цифр.

Таким образом, эксперименты доказывают, что сохранение точности нейронной сети при аффинных преобразованиях входных данных возможно при согласованном преобразовании её весовых параметров.

ЗАКЛЮЧЕНИЕ

В рамках данной работы было проведено исследование возможности применения аффинных преобразований к весам обученных нейронных сетей с сохранением их исходной точности предсказаний. Была достигнута поставленная цель – разработан и реализован алгоритм модификации весов модели без её дообучения, экспериментально подтверждена его корректность и эффективность.

Строго доказано, что аффинное преобразование, применённое к входным данным модели, может быть скомпенсировано соответствующим преобразованием весов первого полносвязного слоя. Для сохранения функционального отображения модели необходимо применять к весам обратное преобразование относительно преобразования данных, но только если матрица этого преобразования не ортогональна.

Был разработан универсальный алгоритм для трёх базовых аффинных преобразований: поворота, масштабирования и расширения. И было экспериментально доказано его работоспособность.

Таким образом, работа продемонстрировала принципиальную возможность и практическую реализуемость безопасного преобразования весов обученных нейронных сетей через аффинные преобразования. Полученные результаты подтверждают, что нейронные сети, несмотря на свою сложность, обладают определённой алгебраической структурой, позволяющей осуществлять контролируемые манипуляции с их параметрами при сохранении функциональной целостности.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Шилдт, Герберт Java. Полное руководство: 12-е изд. : Пер. с англ. – СПб. : ООО “Диалектика”, 2023 – 1638 с.

2. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение / пер. с анг. А. А. Слинкина. – 2-е изд., испр. – М.: ДМК Пресс, 2018. – 652 с.

3. Документация по Deeplearning4j [электронный ресурс]: Режим доступа: https://deeplearning4j.konduit.ai/.

4. Аффинные преобразования [электронный ресурс]: Режим доступа: https://en.wikipedia.org/wiki/Affine_transformation.

Приложение А «Таблица экспериментов с поворотом на 256 градусов для модели Два треугольника»

№	x	y	Ответ	До преобразований (prediction)	До преобразований (confidence)	После преобразования точки (prediction)	После преобразования точки (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После всех преобразований (prediction)	После всех преобразований (confidence)	Совпадение после преобразования точки	Изменение уверенности после преобразования точки	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований
1	0.748883	4.466553	out	out	0.013652	out	0.011325	out	0.012876	out	0.013652	ИСТИНА	-0.002327	ИСТИНА	-0.000776	ИСТИНА	0.000000
2	2.915012	7.792783	out	out	0.011009	out	0.029955	out	0.013105	out	0.011009	ИСТИНА	0.018946	ИСТИНА	0.002096	ИСТИНА	0.000000
3	10.806968	8.793428	out	out	0.100910	out	0.022732	out	0.012856	out	0.100910	ИСТИНА	-0.073178	ИСТИНА	-0.088054	ИСТИНА	0.000000
4	9.720528	5.146038	in	in	0.897826	out	0.012521	out	0.012846	in	0.897826	ЛОЖЬ	-0.885305	ЛОЖЬ	-0.884980	ИСТИНА	-0.000000
5	6.421433	5.919338	in	in	0.945953	out	0.029565	out	0.012831	in	0.945953	ЛОЖЬ	-0.916389	ЛОЖЬ	-0.933122	ИСТИНА	-0.000000
6	1.532912	1.323665	in	in	0.926742	out	0.014082	out	0.012797	in	0.926742	ЛОЖЬ	-0.912660	ЛОЖЬ	-0.913945	ИСТИНА	0.000000
7	10.725274	11.075804	out	out	0.074198	out	0.029556	out	0.012955	out	0.074198	ИСТИНА	-0.044641	ИСТИНА	-0.061243	ИСТИНА	0.000000
8	9.355639	5.716580	in	in	0.977043	out	0.019840	out	0.012838	in	0.977043	ЛОЖЬ	-0.957203	ЛОЖЬ	-0.964205	ИСТИНА	0.000000
9	3.381909	3.165688	in	in	0.953030	out	0.029044	out	0.012811	in	0.953030	ЛОЖЬ	-0.923986	ЛОЖЬ	-0.940219	ИСТИНА	0.000000
10	8.025612	8.196489	in	in	0.984033	out	0.029589	out	0.012871	in	0.984033	ЛОЖЬ	-0.954444	ЛОЖЬ	-0.971161	ИСТИНА	0.000000
11	4.403111	1.850734	in	in	0.820457	out	0.028851	out	0.012811	in	0.820457	ЛОЖЬ	-0.791607	ЛОЖЬ	-0.807647	ИСТИНА	0.000000
12	0.668112	4.915685	out	out	0.011815	out	0.011198	out	0.012876	out	0.011815	ИСТИНА	-0.000617	ИСТИНА	0.001061	ИСТИНА	0.000000
13	5.970264	5.633243	in	in	0.889519	out	0.029590	out	0.012830	in	0.889519	ЛОЖЬ	-0.859929	ЛОЖЬ	-0.876690	ИСТИНА	0.000000
14	10.339024	6.225935	in	in	0.742086	out	0.014304	out	0.012846	in	0.742086	ЛОЖЬ	-0.727781	ЛОЖЬ	-0.729239	ИСТИНА	-0.000000
15	3.966670	1.259314	in	in	0.946845	out	0.028355	out	0.012810	in	0.946845	ЛОЖЬ	-0.918490	ЛОЖЬ	-0.934035	ИСТИНА	0.000000
16	9.105072	1.059733	out	out	0.010959	out	0.011015	out	0.012899	out	0.010959	ИСТИНА	0.000057	ИСТИНА	0.001941	ИСТИНА	0.000000
17	8.692580	6.557457	in	in	0.985662	out	0.028379	out	0.012834	in	0.985662	ЛОЖЬ	-0.957284	ЛОЖЬ	-0.972828	ИСТИНА	0.000000
18	5.592625	3.496603	out	out	0.021800	out	0.029177	out	0.012815	out	0.021800	ИСТИНА	0.007376	ИСТИНА	-0.008986	ИСТИНА	0.000000
19	3.478852	9.713805	out	out	0.010988	out	0.030230	out	0.013500	out	0.010988	ИСТИНА	0.019242	ИСТИНА	0.002512	ИСТИНА	0.000000
20	8.929009	8.338689	in	in	0.900946	out	0.029464	out	0.012861	in	0.900946	ЛОЖЬ	-0.871482	ЛОЖЬ	-0.888085	ИСТИНА	0.000000
21	2.566664	9.321138	out	out	0.011006	out	0.030267	out	0.012974	out	0.011006	ИСТИНА	0.019261	ИСТИНА	0.001968	ИСТИНА	0.000000
22	2.659987	6.161905	out	out	0.011013	out	0.029528	out	0.012928	out	0.011013	ИСТИНА	0.018514	ИСТИНА	0.001915	ИСТИНА	0.000000
23	9.333763	6.351922	in	in	0.976309	out	0.024916	out	0.012836	in	0.976309	ЛОЖЬ	-0.951393	ЛОЖЬ	-0.963473	ИСТИНА	0.000000
24	11.134523	4.545376	out	out	0.093248	out	0.011019	out	0.012883	out	0.093248	ИСТИНА	-0.082228	ИСТИНА	-0.080365	ИСТИНА	0.000000
25	9.453466	7.729161	in	in	0.831754	out	0.028759	out	0.012845	in	0.831755	ЛОЖЬ	-0.802996	ЛОЖЬ	-0.818910	ИСТИНА	0.000001
26	4.522678	1.265478	in	in	0.856819	out	0.027642	out	0.012814	in	0.856819	ЛОЖЬ	-0.829177	ЛОЖЬ	-0.844005	ИСТИНА	0.000000
27	7.607137	6.024675	in	in	0.979792	out	0.029210	out	0.012829	in	0.979792	ЛОЖЬ	-0.950582	ЛОЖЬ	-0.966962	ИСТИНА	0.000000
28	4.607725	2.230707	out	out	0.143867	out	0.029048	out	0.012811	out	0.143868	ИСТИНА	-0.114820	ИСТИНА	-0.131056	ИСТИНА	0.000000
29	3.410720	2.715500	in	in	0.979761	out	0.028875	out	0.012808	in	0.979762	ЛОЖЬ	-0.950886	ЛОЖЬ	-0.966953	ИСТИНА	0.000000
30	5.808163	3.463310	out	out	0.018694	out	0.028930	out	0.012816	out	0.018694	ИСТИНА	0.010236	ИСТИНА	-0.005878	ИСТИНА	-0.000000
31	8.703228	10.721523	out	out	0.228629	out	0.029706	out	0.013050	out	0.228629	ИСТИНА	-0.198923	ИСТИНА	-0.215579	ИСТИНА	0.000000
32	0.819743	1.325585	out	out	0.020923	out	0.011082	out	0.012777	out	0.020923	ИСТИНА	-0.009841	ИСТИНА	-0.008146	ИСТИНА	0.000000
33	5.991563	6.706435	in	in	0.871804	out	0.029652	out	0.012855	in	0.871804	ЛОЖЬ	-0.842152	ЛОЖЬ	-0.858949	ИСТИНА	0.000000
34	4.700014	9.126263	out	out	0.010940	out	0.029960	out	0.013208	out	0.010940	ИСТИНА	0.019020	ИСТИНА	0.002268	ИСТИНА	0.000000
35	8.784705	6.103191	in	in	0.985049	out	0.026878	out	0.012833	in	0.985049	ЛОЖЬ	-0.958172	ЛОЖЬ	-0.972217	ИСТИНА	0.000000
36	1.787260	1.287769	in	in	0.966574	out	0.015987	out	0.012797	in	0.966574	ЛОЖЬ	-0.950587	ЛОЖЬ	-0.953777	ИСТИНА	0.000000
37	4.958366	2.233794	out	out	0.032880	out	0.028645	out	0.012813	out	0.032880	ИСТИНА	-0.004235	ИСТИНА	-0.020067	ИСТИНА	0.000000
38	1.115230	1.226435	out	out	0.159314	out	0.011568	out	0.012769	out	0.159314	ИСТИНА	-0.147745	ИСТИНА	-0.146545	ИСТИНА	0.000000
39	6.208071	8.980055	out	out	0.010905	out	0.029784	out	0.013013	out	0.010905	ИСТИНА	0.018878	ИСТИНА	0.002108	ИСТИНА	0.000000
40	10.181705	6.179089	in	in	0.843038	out	0.015307	out	0.012845	in	0.843038	ЛОЖЬ	-0.827731	ЛОЖЬ	-0.830193	ИСТИНА	0.000000
41	2.237359	7.911226	out	out	0.011011	out	0.029800	out	0.013068	out	0.011011	ИСТИНА	0.018788	ИСТИНА	0.002057	ИСТИНА	0.000000
42	0.478516	9.843496	out	out	0.011013	out	0.011109	out	0.010787	out	0.011013	ИСТИНА	0.000097	ИСТИНА	-0.000225	ИСТИНА	0.000000
43	2.566568	4.970932	out	out	0.011025	out	0.028944	out	0.012860	out	0.011025	ИСТИНА	0.017918	ИСТИНА	0.001835	ИСТИНА	0.000000
44	6.071526	8.838198	out	out	0.010938	out	0.029784	out	0.013005	out	0.010938	ИСТИНА	0.018846	ИСТИНА	0.002067	ИСТИНА	0.000000
45	4.615602	1.526862	in	in	0.822274	out	0.028054	out	0.012813	in	0.822274	ЛОЖЬ	-0.794220	ЛОЖЬ	-0.809461	ИСТИНА	-0.000000
46	1.536477	0.001555	out	out	0.013186	out	0.012408	out	0.011995	out	0.013186	ИСТИНА	-0.000777	ИСТИНА	-0.001191	ИСТИНА	0.000000
47	7.160419	9.162447	in	in	0.878086	out	0.029718	out	0.012968	in	0.878086	ЛОЖЬ	-0.848368	ЛОЖЬ	-0.865118	ИСТИНА	0.000000
48	6.955130	6.118203	in	in	0.971345	out	0.029503	out	0.012832	in	0.971345	ЛОЖЬ	-0.941842	ЛОЖЬ	-0.958513	ИСТИНА	0.000000
49	8.003185	8.169173	in	in	0.984478	out	0.029589	out	0.012870	in	0.984478	ЛОЖЬ	-0.954889	ЛОЖЬ	-0.971608	ИСТИНА	0.000000
50	4.154975	4.025421	out	out	0.226443	out	0.029550	out	0.012817	out	0.226443	ИСТИНА	-0.196893	ИСТИНА	-0.213626	ИСТИНА	0.000000

Приложение В «Таблица экспериментов с поворотом на 256 градусов для модели MNIST»

№	Цифра	До преобразований (prediction)	До преобразований (confidence)	После преобразования изображения (prediction)	После преобразования изображения (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После всех преобразований (prediction)	После всех преобразований (confidence)	Совпадение после преобразования изображени	Изменение уверенности после преобразования изображени	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований
1	0	0	1.000000	2	1.000000	5	1.000000	0	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
2	0	0	1.000000	2	1.000000	2	1.000000	0	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
3	0	0	1.000000	1	1.000000	1	1.000000	0	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
4	0	0	1.000000	1	1.000000	1	1.000000	0	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
5	0	0	1.000000	2	1.000000	8	1.000000	0	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
6	1	1	1.000000	0	1.000000	7	1.000000	1	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
7	1	1	1.000000	7	1.000000	7	1.000000	1	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
8	1	1	1.000000	7	1.000000	7	1.000000	1	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
9	1	1	1.000000	4	1.000000	7	1.000000	1	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
10	1	1	1.000000	7	1.000000	4	1.000000	1	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
11	2	2	1.000000	2	1.000000	1	1.000000	2	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
12	2	2	1.000000	2	1.000000	5	1.000000	2	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
13	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
14	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
15	2	6	1.000000	2	1.000000	2	1.000000	6	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
16	3	3	1.000000	2	1.000000	0	1.000000	3	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
17	3	3	1.000000	2	1.000000	5	1.000000	3	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
18	3	3	1.000000	3	1.000000	5	1.000000	3	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
19	3	3	1.000000	0	1.000000	0	1.000000	3	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
20	3	3	1.000000	5	1.000000	0	1.000000	3	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
21	4	4	1.000000	3	1.000000	2	1.000000	4	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
22	4	4	1.000000	0	1.000000	9	1.000000	4	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
23	4	4	1.000000	0	1.000000	4	1.000000	4	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
24	4	4	1.000000	8	1.000000	2	1.000000	4	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
25	4	4	1.000000	2	1.000000	3	1.000000	4	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
26	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
27	5	5	1.000000	6	1.000000	5	1.000000	5	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
28	5	5	1.000000	2	1.000000	5	1.000000	5	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
29	5	5	1.000000	0	1.000000	3	1.000000	5	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
30	5	5	1.000000	5	1.000000	0	1.000000	5	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
31	6	6	1.000000	2	1.000000	2	1.000000	6	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
32	6	6	1.000000	5	1.000000	5	1.000000	6	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
33	6	6	1.000000	6	1.000000	2	1.000000	6	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
34	6	6	1.000000	2	1.000000	3	1.000000	6	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
35	6	6	1.000000	2	1.000000	2	1.000000	6	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
36	7	7	1.000000	1	1.000000	7	1.000000	7	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
37	7	7	1.000000	2	1.000000	0	1.000000	7	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
38	7	7	1.000000	1	1.000000	0	1.000000	7	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
39	7	7	1.000000	3	1.000000	7	1.000000	7	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
40	7	3	1.000000	0	1.000000	0	1.000000	3	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
41	8	8	1.000000	0	1.000000	7	1.000000	8	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
42	8	8	1.000000	2	1.000000	0	1.000000	8	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
43	8	8	1.000000	0	1.000000	5	1.000000	8	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
44	8	8	1.000000	5	1.000000	7	1.000000	8	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
45	8	8	1.000000	2	1.000000	4	1.000000	8	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
46	9	9	1.000000	3	1.000000	2	1.000000	9	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
47	9	9	1.000000	0	1.000000	0	1.000000	9	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
48	9	9	1.000000	0	1.000000	0	1.000000	9	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
49	9	9	1.000000	0	1.000000	0	1.000000	9	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
50	9	9	1.000000	0	1.000000	0	1.000000	9	1.000000	ЛОЖЬ	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000

Приложение С «Таблица экспериментов с растяжением в 2 раза для модели Два треугольника»

№	x	y	Ответ	До преобразований (prediction)	До преобразований (confidence)	После преобразования точки (prediction)	После преобразования точки (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После всех преобразований (prediction)	После всех преобразований (confidence)	Совпадение после преобразования точек	Изменение уверенности после преобразования точек	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований
1	0.748883	4.466553	out	out	0.013652	out	0.000030	out	0.011239	out	0.013652	ИСТИНА	-0.013622	ИСТИНА	-0.002413	ИСТИНА	0.000000
2	2.915012	7.792783	out	out	0.011009	out	0.000141	out	0.013956	out	0.011009	ИСТИНА	-0.010868	ИСТИНА	0.002947	ИСТИНА	0.000000
3	10.806968	8.793428	out	out	0.100910	out	0.064464	in	0.984234	out	0.100910	ИСТИНА	-0.036446	ЛОЖЬ	0.883324	ИСТИНА	0.000000
4	9.720528	5.146038	in	in	0.897826	out	0.064463	in	0.923017	in	0.897826	ЛОЖЬ	-0.833363	ИСТИНА	0.025191	ИСТИНА	0.000000
5	6.421433	5.919338	in	in	0.945953	out	0.064464	in	0.559402	in	0.945953	ЛОЖЬ	-0.881489	ИСТИНА	-0.386551	ИСТИНА	0.000000
6	1.552912	1.323665	in	in	0.926742	in	0.985889	out	0.011076	in	0.926742	ИСТИНА	0.059147	ЛОЖЬ	-0.915666	ИСТИНА	0.000000
7	10.725274	11.075804	out	out	0.074198	out	0.064464	in	0.981775	out	0.074198	ИСТИНА	-0.009734	ЛОЖЬ	0.907577	ИСТИНА	0.000000
8	9.355639	5.716380	in	in	0.977043	out	0.064464	in	0.956938	in	0.977043	ЛОЖЬ	-0.912579	ИСТИНА	-0.020105	ИСТИНА	0.000000
9	3.381909	3.165688	in	in	0.953030	out	0.064498	out	0.012056	in	0.953030	ЛОЖЬ	-0.888531	ЛОЖЬ	-0.940974	ИСТИНА	0.000000
10	8.025612	8.196489	in	in	0.984033	out	0.064464	in	0.922502	in	0.984033	ЛОЖЬ	-0.919568	ИСТИНА	-0.061531	ИСТИНА	0.000000
11	4.403111	1.850734	in	in	0.820457	out	0.056412	out	0.012283	in	0.820457	ЛОЖЬ	-0.764046	ЛОЖЬ	-0.808175	ИСТИНА	0.000000
12	0.668112	4.915685	out	out	0.011815	out	0.000046	out	0.011339	out	0.011815	ИСТИНА	-0.011770	ИСТИНА	-0.000477	ИСТИНА	0.000000
13	5.970264	5.633243	in	in	0.889519	out	0.064464	out	0.308440	in	0.889519	ЛОЖЬ	-0.825055	ЛОЖЬ	-0.581079	ИСТИНА	0.000000
14	10.339024	6.225935	in	in	0.742086	out	0.064464	in	0.974203	in	0.742084	ЛОЖЬ	-0.677621	ИСТИНА	0.232117	ИСТИНА	-0.000001
15	3.966670	1.259314	in	in	0.946845	out	0.048943	out	0.011806	in	0.946845	ЛОЖЬ	-0.897902	ЛОЖЬ	-0.935039	ИСТИНА	0.000000
16	9.105072	1.059733	out	out	0.010959	out	0.048775	out	0.014249	out	0.010959	ИСТИНА	0.037816	ИСТИНА	0.003290	ИСТИНА	0.000000
17	8.692580	6.557457	in	in	0.985662	out	0.064464	in	0.963766	in	0.985662	ЛОЖЬ	-0.921198	ИСТИНА	-0.021897	ИСТИНА	0.000000
18	5.592625	3.496603	out	out	0.021800	out	0.064463	out	0.026027	out	0.021800	ИСТИНА	0.042663	ИСТИНА	0.004227	ИСТИНА	0.000000
19	3.478852	9.713805	out	out	0.010988	out	0.000141	out	0.015495	out	0.010988	ИСТИНА	-0.010848	ИСТИНА	0.004506	ИСТИНА	0.000000
20	8.929009	8.338689	in	in	0.900946	out	0.064464	in	0.968632	in	0.900946	ЛОЖЬ	-0.836482	ИСТИНА	0.067686	ИСТИНА	0.000000
21	2.566664	9.321138	out	out	0.011006	out	0.000141	out	0.014215	out	0.011006	ИСТИНА	-0.010865	ИСТИНА	0.003210	ИСТИНА	0.000000
22	2.659987	6.161905	out	out	0.011013	out	0.000141	out	0.013217	out	0.011013	ИСТИНА	-0.010873	ИСТИНА	0.002204	ИСТИНА	0.000000
23	9.333763	6.351922	in	in	0.976309	out	0.064464	in	0.970097	in	0.976309	ЛОЖЬ	-0.911845	ИСТИНА	-0.006212	ИСТИНА	0.000000
24	11.134523	4.545376	out	out	0.093248	out	0.055475	in	0.767023	out	0.093248	ИСТИНА	-0.037773	ЛОЖЬ	0.673775	ИСТИНА	0.000000
25	9.453466	7.729161	in	in	0.831754	out	0.064464	in	0.977211	in	0.831755	ЛОЖЬ	-0.767290	ИСТИНА	0.145457	ИСТИНА	0.000000
26	4.522678	1.265478	in	in	0.856819	out	0.048792	out	0.012174	in	0.856819	ЛОЖЬ	-0.808027	ЛОЖЬ	-0.844645	ИСТИНА	0.000000
27	7.607137	6.024675	in	in	0.979792	out	0.064464	in	0.901627	in	0.979792	ЛОЖЬ	-0.915328	ИСТИНА	-0.078164	ИСТИНА	0.000000
28	4.607725	2.230707	out	out	0.143867	out	0.063287	out	0.012813	out	0.143868	ИСТИНА	-0.080581	ИСТИНА	-0.131054	ИСТИНА	0.000000
29	3.410720	2.715500	in	in	0.979761	out	0.064528	out	0.011793	in	0.979762	ЛОЖЬ	-0.915233	ЛОЖЬ	-0.967968	ИСТИНА	0.000000
30	5.808163	3.463310	out	out	0.018694	out	0.064461	out	0.028260	out	0.018694	ИСТИНА	0.045768	ИСТИНА	0.009566	ИСТИНА	0.000000
31	8.703228	10.721523	out	out	0.228629	out	0.064464	in	0.860499	out	0.228629	ИСТИНА	-0.164165	ЛОЖЬ	0.631870	ИСТИНА	0.000000
32	0.819743	1.325585	out	out	0.020923	out	0.011018	out	0.011037	out	0.020923	ИСТИНА	-0.009905	ИСТИНА	-0.009886	ИСТИНА	0.000000
33	5.991563	6.706435	in	in	0.871804	out	0.064464	out	0.280121	in	0.871805	ЛОЖЬ	-0.807340	ЛОЖЬ	-0.391683	ИСТИНА	0.000000
34	4.700014	9.126263	out	out	0.010940	out	0.000141	out	0.020791	out	0.010940	ИСТИНА	-0.010799	ИСТИНА	0.009851	ИСТИНА	0.000000
35	8.784705	6.103191	in	in	0.985049	out	0.064464	in	0.959069	in	0.985049	ЛОЖЬ	-0.920585	ИСТИНА	-0.025981	ИСТИНА	0.000000
36	1.787260	1.287769	in	in	0.966574	in	0.983911	out	0.011099	in	0.966574	ИСТИНА	0.017337	ЛОЖЬ	-0.955475	ИСТИНА	0.000000
37	4.958566	2.223794	out	out	0.032880	out	0.061271	out	0.013323	out	0.032880	ИСТИНА	0.028392	ИСТИНА	-0.019557	ИСТИНА	0.000000
38	1.115230	1.226435	out	out	0.159314	in	0.720706	out	0.011049	out	0.159314	ЛОЖЬ	0.561392	ИСТИНА	-0.148264	ИСТИНА	0.000000
39	6.208071	8.980055	out	out	0.010905	out	0.000145	out	0.094602	out	0.010905	ИСТИНА	-0.010761	ИСТИНА	0.083697	ИСТИНА	0.000000
40	10.181705	6.179089	in	in	0.843038	out	0.064464	in	0.973049	in	0.843038	ЛОЖЬ	-0.778574	ИСТИНА	0.130011	ИСТИНА	0.000000
41	2.237359	7.911226	out	out	0.011011	out	0.000141	out	0.013195	out	0.011011	ИСТИНА	-0.010870	ИСТИНА	0.002184	ИСТИНА	0.000000
42	0.478516	9.843496	out	out	0.011013	out	0.000141	out	0.014307	out	0.011013	ИСТИНА	-0.010872	ИСТИНА	0.003294	ИСТИНА	0.000000
43	2.566568	4.970932	out	out	0.011025	out	0.000140	out	0.012461	out	0.011025	ИСТИНА	-0.010885	ИСТИНА	0.001436	ИСТИНА	0.000000
44	6.071526	8.838198	out	out	0.010938	out	0.000143	out	0.082990	out	0.010938	ИСТИНА	-0.010795	ИСТИНА	0.072052	ИСТИНА	0.000000
45	4.615602	1.526862	in	in	0.822274	out	0.049124	out	0.012326	in	0.822274	ЛОЖЬ	-0.773151	ЛОЖЬ	-0.809949	ИСТИНА	-0.000000
46	1.536477	0.001555	out	out	0.013186	out	0.010998	out	0.011120	out	0.013186	ИСТИНА	-0.002188	ИСТИНА	-0.002066	ИСТИНА	0.000000
47	7.160419	9.162447	in	in	0.878086	out	0.064400	out	0.436221	in	0.878086	ЛОЖЬ	-0.813686	ЛОЖЬ	-0.441866	ИСТИНА	-0.000000
48	6.955130	6.118203	in	in	0.971345	out	0.064464	in	0.787829	in	0.971345	ЛОЖЬ	-0.906881	ИСТИНА	-0.183516	ИСТИНА	0.000000
49	8.003185	8.169173	in	in	0.984478	out	0.064464	in	0.921024	in	0.984478	ЛОЖЬ	-0.920014	ИСТИНА	-0.063454	ИСТИНА	0.000000
50	4.154975	4.025421	out	out	0.226443	out	0.064465	out	0.016583	out	0.226443	ИСТИНА	-0.161978	ИСТИНА	-0.209860	ИСТИНА	0.000000

Приложение D «Таблица экспериментов с растяжением в 2 раза для модели MNIST»

№	Цифра	До преобразований (prediction)	До преобразований (confidence)	После преобразования изображения (prediction)	После преобразования изображения (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После всех преобразований (prediction)	После всех преобразований (confidence)	Совпадение после преобразования изображения	Изменение уверенности после преобразования изображение	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований
1	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
2	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
3	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
4	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
5	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
6	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
7	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
8	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
9	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
10	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
11	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
12	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
13	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
14	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
15	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
16	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
17	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
18	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
19	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
20	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
21	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
22	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
23	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
24	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
25	4	7	0.848400	7	1.000000	1	0.736971	7	0.843361	ИСТИНА	0.151600	ЛОЖЬ	-0.111429	ИСТИНА	-0.000039
26	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
27	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
28	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
29	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
30	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
31	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
32	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
33	6	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
34	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
35	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
36	7	7	1.000000	7	1.000000	7	1.000000	7	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
37	7	7	1.000000	7	1.000000	7	1.000000	7	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
38	7	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
39	7	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	-0.000000	ИСТИНА	0.000000
40	7	7	0.804943	7	0.999861	9	0.518935	7	0.804962	ИСТИНА	0.194918	ЛОЖЬ	-0.286608	ИСТИНА	0.000019
41	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
42	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
43	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
44	8	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
45	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
46	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
47	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
48	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
49	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
50	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000

Приложение Е «Таблица экспериментов с расширением на коэффициент 0.2 для модели Два треугольника»

№	x	y	Ответ	До преобразования (prediction)	До преобразования (confidence)	После преобразования точки (prediction)	После преобразования точки (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После преобразования (prediction)	После всех преобразований (confidence)	Совпадение после преобразования точки	Изменение уверенности после преобразования точки	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований
1	0.748883	4.466553	out	out	0.058330	out	0.008863	out	0.257380	out	0.058330	ИСТИНА	-0.049467	ИСТИНА	0.199050	ИСТИНА	0.000000
2	2.915012	7.792783	out	out	0.007039	out	0.007040	out	0.007040	out	0.007039	ИСТИНА	0.000002	ИСТИНА	0.000001	ИСТИНА	0.000000
3	10.806968	8.793428	out	in	0.112117	out	0.035415	in	0.992929	out	0.112117	ИСТИНА	-0.076702	ЛОЖЬ	0.880812	ИСТИНА	0.000000
4	9.720528	5.146038	in	in	0.871569	in	0.711415	out	0.008068	in	0.871570	ИСТИНА	-0.160154	ЛОЖЬ	-0.863501	ИСТИНА	0.000000
5	6.421433	5.919338	in	in	0.962739	in	0.993300	out	0.405668	in	0.962739	ИСТИНА	0.030561	ЛОЖЬ	-0.557071	ИСТИНА	0.000000
6	1.552912	1.323665	in	in	0.966482	in	0.983528	in	0.745517	in	0.966482	ИСТИНА	0.017046	ИСТИНА	-0.220965	ИСТИНА	0.000000
7	10.725274	11.075804	out	out	0.044843	out	0.034109	in	0.974599	out	0.044843	ИСТИНА	-0.010733	ЛОЖЬ	0.929756	ИСТИНА	0.000000
8	9.355639	5.716580	in	in	0.990762	in	0.704562	out	0.009522	in	0.990762	ИСТИНА	-0.286201	ЛОЖЬ	-0.981241	ИСТИНА	0.000000
9	3.381909	3.165688	in	in	0.932532	out	0.083337	in	0.991611	in	0.932532	ЛОЖЬ	-0.849195	ИСТИНА	0.059079	ИСТИНА	-0.000000
10	8.025612	8.196489	in	in	0.994599	out	0.370167	in	0.983850	in	0.994599	ЛОЖЬ	-0.624432	ИСТИНА	-0.010749	ИСТИНА	0.000000
11	4.403111	1.850734	in	in	0.816637	in	0.010728	in	0.833571	in	0.816638	ЛОЖЬ	-0.805909	ИСТИНА	0.015934	ИСТИНА	0.000001
12	0.668112	4.915685	out	out	0.027775	out	0.007420	out	0.216514	out	0.027775	ИСТИНА	-0.020355	ИСТИНА	0.188739	ИСТИНА	0.000000
13	5.970264	5.633243	in	in	0.916516	in	0.989602	out	0.234686	in	0.916516	ИСТИНА	0.073086	ЛОЖЬ	-0.681831	ИСТИНА	0.000000
14	10.339024	6.225935	in	in	0.981418	out	0.062777	out	0.018083	in	0.981418	ЛОЖЬ	-0.918641	ЛОЖЬ	-0.963335	ИСТИНА	-0.000000
15	3.966670	1.259314	in	in	0.984042	in	0.825397	out	0.009477	in	0.984042	ИСТИНА	-0.158646	ЛОЖЬ	-0.974565	ИСТИНА	0.000000
16	9.105072	1.059733	out	out	0.007016	out	0.007887	out	0.006925	out	0.007016	ИСТИНА	0.000871	ИСТИНА	-0.000091	ИСТИНА	0.000000
17	8.692580	6.557457	in	in	0.994279	in	0.796698	in	0.500095	in	0.994279	ИСТИНА	-0.197581	ИСТИНА	-0.494184	ИСТИНА	0.000000
18	5.592625	3.496603	out	out	0.008345	out	0.092097	out	0.009244	out	0.008345	ИСТИНА	0.083752	ИСТИНА	0.000899	ИСТИНА	0.000000
19	3.478852	9.713805	out	in	0.007040	out	0.007033	out	0.007038	out	0.007040	ИСТИНА	-0.000007	ИСТИНА	-0.000001	ИСТИНА	0.000000
20	8.929009	8.338689	in	in	0.990527	out	0.062797	in	0.992306	in	0.990527	ЛОЖЬ	-0.927730	ИСТИНА	0.001779	ИСТИНА	0.000000
21	2.566664	9.321138	out	out	0.007039	out	0.007040	out	0.007038	out	0.007039	ИСТИНА	0.000002	ИСТИНА	-0.000000	ИСТИНА	0.000000
22	2.659987	6.161905	out	out	0.007040	out	0.007041	out	0.007080	out	0.007040	ИСТИНА	0.000001	ИСТИНА	0.000039	ИСТИНА	0.000000
23	9.333763	6.351922	in	in	0.993916	out	0.306754	out	0.051092	in	0.993916	ЛОЖЬ	-0.687162	ЛОЖЬ	-0.942824	ИСТИНА	0.000000
24	11.134523	4.545376	out	out	0.083259	out	0.084042	out	0.009189	out	0.083259	ИСТИНА	0.000783	ИСТИНА	-0.074070	ИСТИНА	0.000000
25	9.453466	7.729161	in	in	0.986636	out	0.057165	in	0.991535	in	0.986636	ЛОЖЬ	-0.929470	ИСТИНА	0.004899	ИСТИНА	0.000000
26	4.522678	1.265478	in	in	0.939251	out	0.030788	out	0.008225	in	0.939251	ЛОЖЬ	-0.908462	ЛОЖЬ	-0.931026	ИСТИНА	0.000000
27	7.607137	6.024675	in	in	0.986923	in	0.994110	out	0.100233	in	0.986923	ИСТИНА	0.007187	ЛОЖЬ	-0.877691	ИСТИНА	0.000000
28	4.607725	2.230707	out	out	0.055153	out	0.008541	in	0.971009	out	0.055153	ИСТИНА	-0.046612	ЛОЖЬ	0.915857	ИСТИНА	0.000000
29	3.410720	2.715500	in	in	0.976773	out	0.089838	in	0.992729	in	0.976773	ЛОЖЬ	-0.886935	ИСТИНА	0.015956	ИСТИНА	0.000000
30	5.808163	3.463310	out	out	0.008047	out	0.083205	out	0.008426	out	0.008047	ИСТИНА	0.075158	ИСТИНА	0.000379	ИСТИНА	0.000000
31	8.703228	10.721523	out	out	0.812419	out	0.037813	out	0.009421	in	0.812419	ЛОЖЬ	-0.774606	ЛОЖЬ	-0.802998	ИСТИНА	0.000000
32	0.819743	1.325585	out	out	0.011148	out	0.029074	out	0.008048	out	0.011148	ИСТИНА	0.017925	ИСТИНА	-0.003101	ИСТИНА	-0.000000
33	5.991563	6.706435	in	in	0.911216	in	0.993094	out	0.070521	in	0.911217	ИСТИНА	0.081877	ЛОЖЬ	-0.840695	ИСТИНА	0.000000
34	4.700014	9.126263	out	out	0.007041	out	0.006999	out	0.007039	out	0.007041	ИСТИНА	-0.000042	ИСТИНА	-0.000002	ИСТИНА	0.000000
35	8.784705	6.103191	in	in	0.993252	in	0.912899	out	0.026140	in	0.993252	ИСТИНА	-0.080353	ЛОЖЬ	-0.967111	ИСТИНА	0.000000
36	1.787260	1.287769	in	in	0.986596	in	0.990830	in	0.853128	in	0.986596	ИСТИНА	0.004234	ИСТИНА	-0.133468	ИСТИНА	0.000000
37	4.958566	2.223794	out	out	0.014606	out	0.007988	in	0.867712	out	0.014606	ИСТИНА	-0.006617	ЛОЖЬ	0.853107	ИСТИНА	0.000000
38	1.115230	1.226435	out	out	0.155192	in	0.661116	out	0.021891	out	0.155192	ЛОЖЬ	0.505924	ИСТИНА	-0.133301	ИСТИНА	0.000000
39	6.208071	8.980055	out	out	0.007172	in	0.974577	out	0.007041	out	0.007172	ЛОЖЬ	0.967406	ИСТИНА	-0.000131	ИСТИНА	0.000000
40	10.181705	6.179089	in	in	0.987254	out	0.076054	out	0.016420	in	0.987254	ЛОЖЬ	-0.911200	ЛОЖЬ	-0.970834	ИСТИНА	0.000000
41	2.237359	7.911226	out	out	0.007039	out	0.007039	out	0.007041	out	0.007039	ИСТИНА	0.000001	ИСТИНА	0.000002	ИСТИНА	0.000000
42	0.478516	9.843496	out	out	0.007038	out	0.007039	out	0.007040	out	0.007038	ИСТИНА	0.000001	ИСТИНА	0.000001	ИСТИНА	0.000000
43	2.566568	4.970932	out	out	0.007072	out	0.007062	out	0.008732	out	0.007072	ИСТИНА	-0.000010	ИСТИНА	0.001660	ИСТИНА	0.000000
44	6.071526	8.838198	out	out	0.007137	in	0.964094	out	0.007041	out	0.007137	ЛОЖЬ	0.956957	ИСТИНА	-0.000096	ИСТИНА	0.000000
45	4.615602	1.526862	in	in	0.868888	out	0.011230	out	0.010987	in	0.868888	ЛОЖЬ	-0.857658	ЛОЖЬ	-0.857902	ИСТИНА	0.000000
46	1.536477	0.001555	out	out	0.007991	out	0.008389	out	0.007970	out	0.007991	ИСТИНА	0.000398	ИСТИНА	-0.000021	ИСТИНА	0.000000
47	7.160419	9.162447	in	in	0.624955	in	0.611439	out	0.007113	in	0.624953	ИСТИНА	-0.013516	ЛОЖЬ	-0.617843	ИСТИНА	-0.000003
48	6.955130	6.118203	in	in	0.981169	in	0.994574	out	0.476691	in	0.981169	ИСТИНА	0.013406	ЛОЖЬ	-0.504478	ИСТИНА	0.000000
49	8.003185	8.169173	in	in	0.994601	out	0.408043	in	0.983382	in	0.994601	ЛОЖЬ	-0.386558	ИСТИНА	-0.011219	ИСТИНА	0.000000
50	4.154975	4.025421	out	out	0.078693	out	0.409878	in	0.784280	out	0.078693	ИСТИНА	0.331185	ЛОЖЬ	0.705586	ИСТИНА	-0.000000

Приложение F «Таблица экспериментов с расширением на коэффициент 0.2 для модели MNIST»

№	Цифра	До преобразований (prediction)	До преобразований (confidence)	После преобразования изображения (prediction)	После преобразования изображения (confidence)	После преобразования весов (prediction)	После преобразования весов (confidence)	После всех преобразований (prediction)	После всех преобразований (confidence)	Совпадение после преобразования изображений	Изменение уверенности после преобразования изображений	Совпадение после преобразования весов	Изменение уверенности после преобразования весов	Совпадение после всех преобразований	Изменение уверенности после всех преобразований2
1	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
2	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
3	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
4	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
5	0	0	1.000000	0	1.000000	0	1.000000	0	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
6	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
7	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
8	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
9	1	1	1.000000	1	1.000000	1	1.000000	1	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
10	1	1	1.000000	1	0.959434	1	1.000000	1	1.000000	ИСТИНА	-0.040566	ИСТИНА	0.000000	ИСТИНА	0.000000
11	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
12	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
13	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
14	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
15	2	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
16	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
17	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
18	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
19	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
20	3	3	1.000000	3	1.000000	3	1.000000	3	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
21	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
22	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
23	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
24	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
25	4	4	1.000000	4	1.000000	4	1.000000	4	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
26	5	3	1.000000	3	1.000000	5	1.000000	3	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
27	5	5	1.000000	5	1.000000	5	1.000000	5	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
28	5	6	1.000000	6	1.000000	5	1.000000	6	1.000000	ИСТИНА	0.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000
29	5	5	1.000000	3	1.000000	5	1.000000	5	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
30	5	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
31	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
32	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
33	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
34	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
35	6	6	1.000000	6	1.000000	6	1.000000	6	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
36	7	7	1.000000	7	1.000000	7	1.000000	7	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
37	7	7	1.000000	9	1.000000	7	1.000000	7	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
38	7	7	1.000000	7	1.000000	7	1.000000	7	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
39	7	8	0.999977	2	1.000000	8	1.000000	8	0.999977	ЛОЖЬ	0.000023	ИСТИНА	0.000023	ИСТИНА	0.000000
40	7	8	0.999868	7	1.000000	8	0.998103	8	0.999868	ЛОЖЬ	0.000132	ИСТИНА	-0.001765	ИСТИНА	-0.000000
41	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
42	8	5	1.000000	3	1.000000	5	1.000000	5	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
43	8	8	1.000000	8	1.000000	8	1.000000	8	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
44	8	2	1.000000	2	1.000000	2	1.000000	2	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
45	8	8	1.000000	3	1.000000	8	1.000000	8	1.000000	ЛОЖЬ	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
46	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
47	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
48	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
49	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000
50	9	9	1.000000	9	1.000000	9	1.000000	9	1.000000	ИСТИНА	0.000000	ИСТИНА	0.000000	ИСТИНА	0.000000