Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего профессионального образования

«Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского»

Институт Информационных Технологий, Математики и Механики

Направление: Прикладная математика и информатика

Магистерская программа: Компьютерные науки и приложения

**ОТЧЕТ**

по лабораторной работе №3

Тема:

**«Разработка свёрточных нейронных сетей»**

Выполнили: студенты группы 381803-4м

Котова О.А.

\_\_\_\_\_\_\_\_\_\_\_

Подпись

Лицов А.

\_\_\_\_\_\_\_\_\_\_\_

Подпись

Синицкая О.

\_\_\_\_\_\_\_\_\_\_\_

Подпись

Преподаватель: доцент, к.т.н. Кустикова В.Д.

\_\_\_\_\_\_\_\_\_\_\_

Подпись

Нижний Новгород

2019

Оглавление

[1. Постановка задачи 3](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187372)

[2. Тренировочные и тестовые наборы данных 3](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187373)

[3. Метрика качества решения 4](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187374)

[4. Разработанные программы 4](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187374)

[5. Тестовые конфигурации сетей 5](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187375)

[6. Результаты эксперимента 7](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187375)

[7. Заключение 7](file:///C:\Users\user\Downloads\Kursovaya.docx#_Toc420187375)

**Постановка задачи**

**Цели**

Цельнастоящей работы состоит в том, чтобы построить архитектуру сверточной нейронной сети, которая позволяет решать практическую задачу с высокими показателями качества.

**Задачи**

Выполнение практической работы предполагает решение следующих задач:

1. Разработка нескольких архитектур сверточных нейронных сетей (варьируются количество слоев и виды функций активации на каждом слое) в формате, который принимается выбранной библиотекой глубокого обучения.

2. Обучение разработанных глубоких моделей.

3. Тестирование обученных глубоких моделей.

4. Публикация разработанных программ/скриптов в репозитории на GitHub.

5. Подготовка отчета, содержащего минимальный объем информации по каждому этапу выполнения работы.

**Тренировочные и тестовые наборы данных**

Выбранная задача - Intel Image Classification: <https://www.kaggle.com/puneet6060/intel-image-classification>.

Эти данные содержат около 25 тыс. цветных изображений размером 150x150, распределенных по 6 категориям: здания, лес, ледник, гора, море, улица. Изображения хранятся в фомате jpg.

Тренировочная выборка содержит 14034 изображений.

Тестовая выборка содержит 3000 изображений.

Размер каждого изображения 150x150.

|  |  |  |  |
| --- | --- | --- | --- |
| *№* | *Категории* | *Размер тренировочной выборки* | *Размер тестовой выборки* |
| 1 | mountain | 2512 | 525 |
| 2 | street | 2382 | 501 |
| 3 | glasier | 2404 | 553 |
| 4 | buildings | 2191 | 437 |
| 5 | sea | 2274 | 510 |
| 6 | forest | 2271 | 474 |

Процентное соотношение категорий. Тренировочная выборка:



Процентное соотношение категорий. Тестовая выборка:



**Метрика качества решения**

Для оценки качества решения задачи выбрана метрика "Точность" ("Accuracy"). Она вычисляет, как часто прогнозы соответствуют меткам. Иными словами, частота с которой y\_pred совпадает с y\_true.

[](https://github.com/a-litsov/deep-learning/blob/master/lab2/img/accuracy.png)

**Разработанные программы**

Lab3.ipynb – скрипт для обучения свёрточных нейронных сетей.

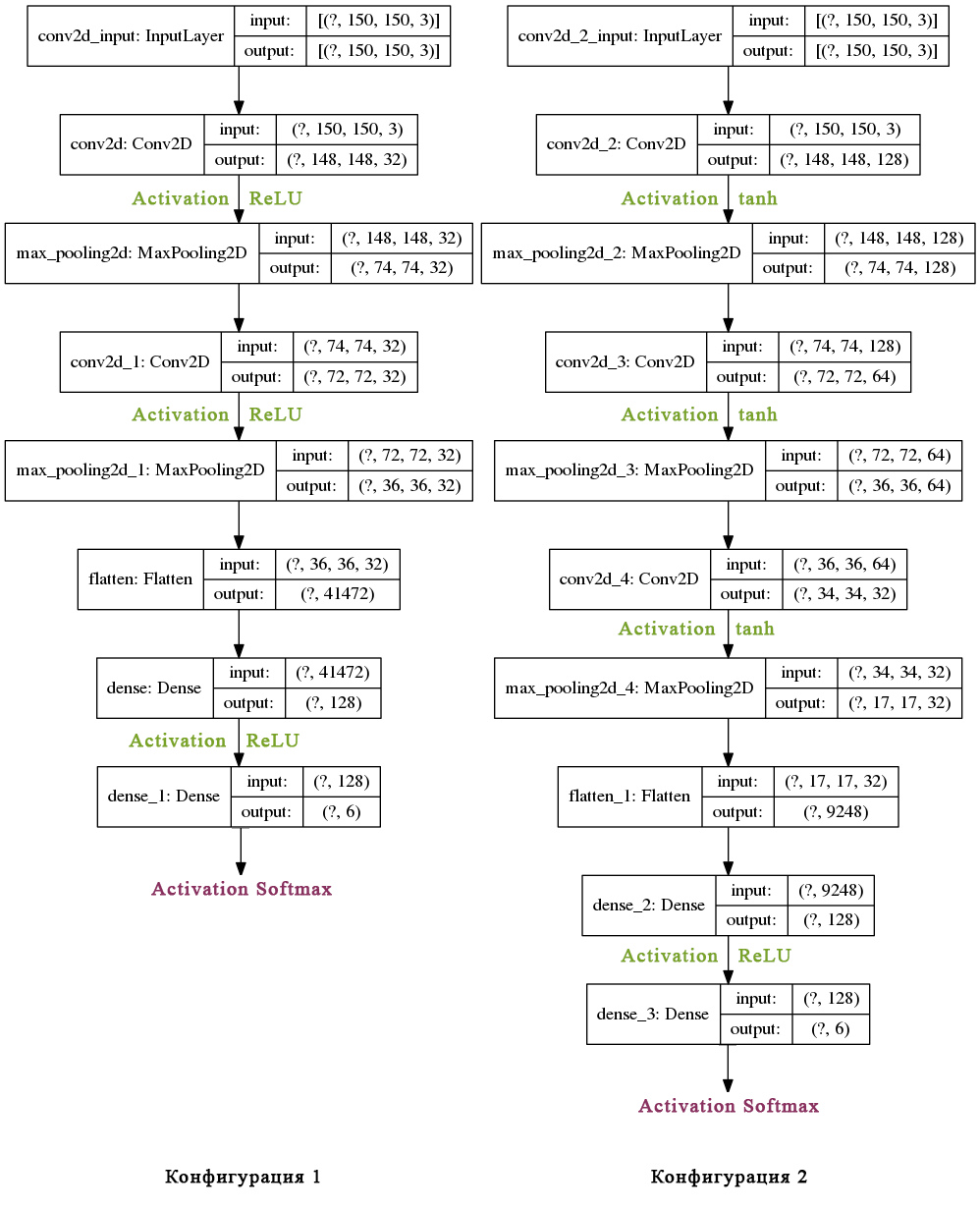
**Тестовые конфигурации сетей**

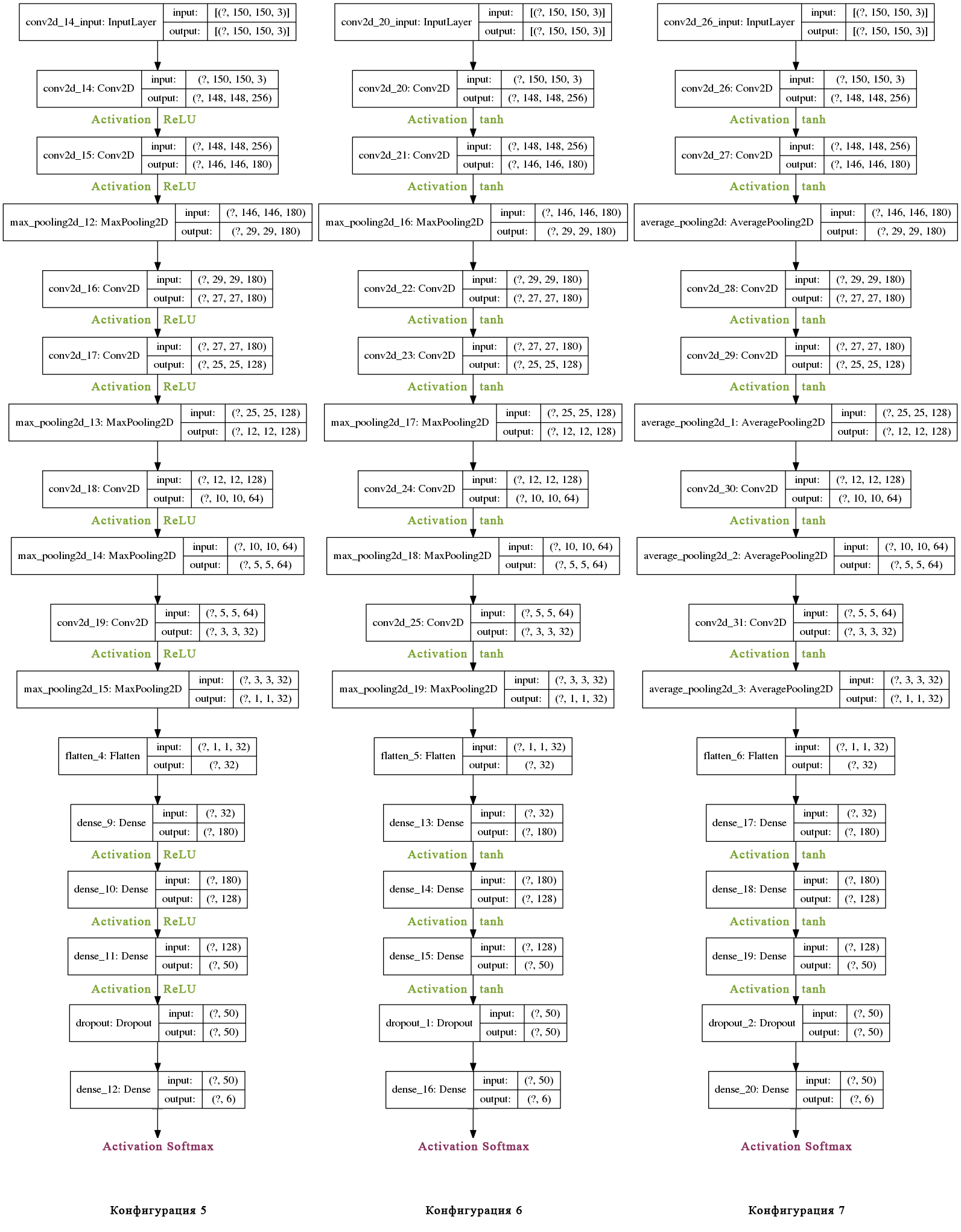
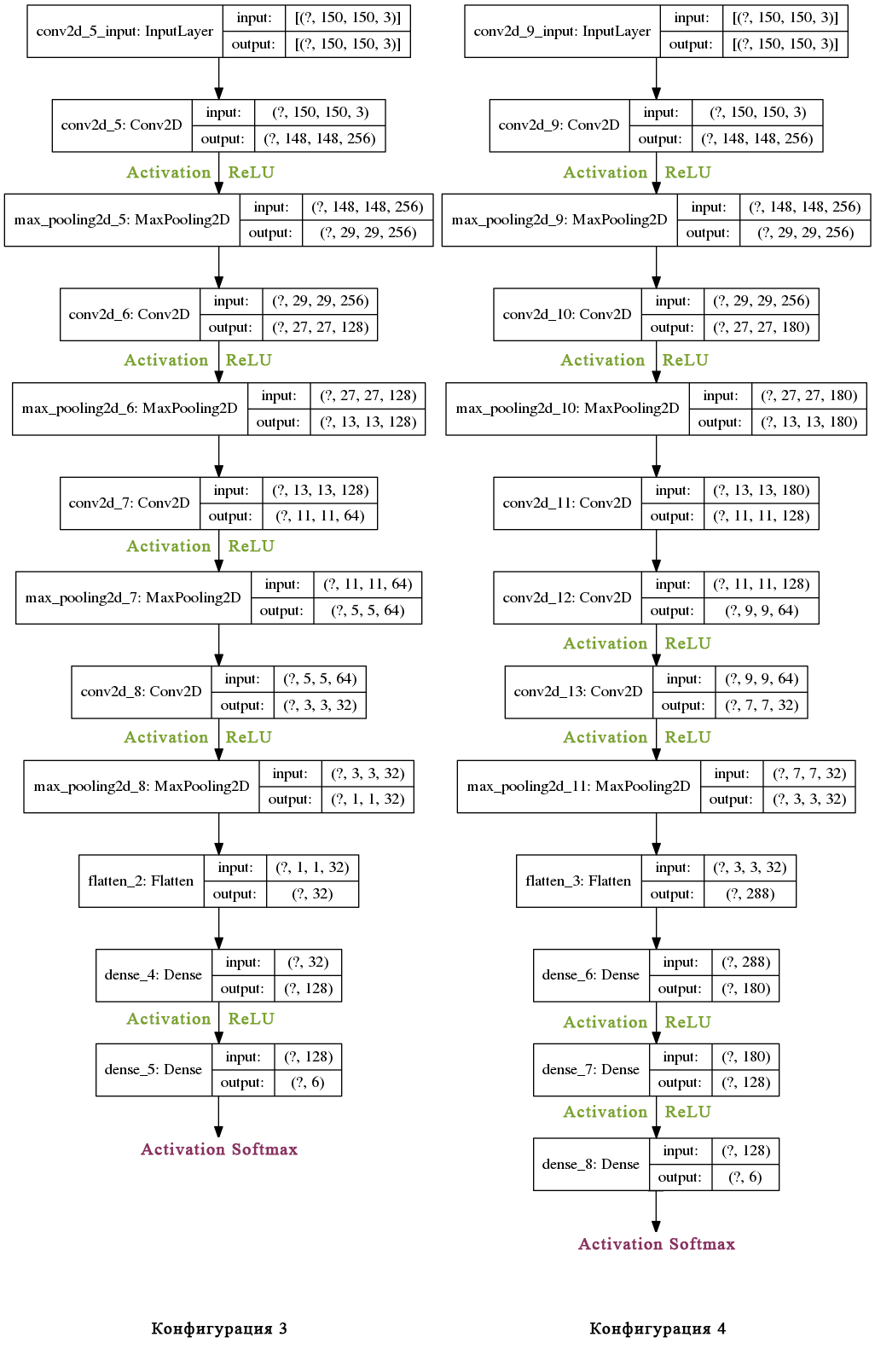
С помощью класса ImageDataGenerator и его метода flow\_from\_directory() генерируем пакеты. Данные возвращаются в формате (x, y), где x, y - numpy массивы.

Форма x: (batch\_size, 150, 150, 3).

Форма y: (batch\_size, 6).

Методу fit\_generator подается на вход генератор данных в формате (x, y). Сети подается на вход массив numpy формата (150, 150, 3), который "сглаживается" сетью с помощью метода Flatten().





**Результаты эксперимента**

В таблице приведены конфигурация системы и программное обеспечение, с помощью которых проводилось обучение и тестирование построенных моделей.

|  |  |
| --- | --- |
| *Параметры* | *Версия* |
| GPU | Tesla P100, having 3584 CUDA cores, 16GB(16.28GB Usable) GDDR6 VRAM Tesla P100 Spec Sheet |
| Python | 3.7.5 |
| TensorFlow | 2.0.0 |

Параметры обучения:

|  |  |
| --- | --- |
| Количество эпох | 20 |
| Размер пачки | 128 |

Результаты экспериментов:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Номер сети | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Среднее время обучения за одну эпоху, с | 35.305 | 41.307 | 46.414 | 43.393 | 113 | 119 | 115 |
| Ошибка на тренировочном наборе | 0.0135 | 0.071 | 0.401 | 0.2511 | 0.2606 | 0.4344 | 0.6398 |
| Ошибка на тестовом наборе | 0.5915 | 0.6261 | 0.5070 | 0.4767 | 0.5015 | 0.5179 | 0.6688 |
| Номер эпохи с достигнутым максимальным качеством решения на тренировочном наборе | 14 | 19 | 20 | 20 | 20 | 19 | 19 |
| Точность (Accuracy) на тренировочном наборе, % | 99.80 | 99.91 | 85.55 | 90.97 | 91.09 | 85.24 | 77.61 |
| Номер эпохи с достигнутым максимальным качеством решения на тестовом наборе | 14 | 19 | 20 | 13 | 15 | 15 | 19 |
| Точность (Accuracy) на тестовом наборе, % | 79.47 | 80.33 | 82.37 | 84.07 | 84.03 | 82.13 | 75.57 |

**Анализ результатов**