**Домашнее задание № 5**

**Сверточные нейронные сети (CNN).**

**LeNet**

Схема сети представлена на рисунке 1. Сеть содержит 83 126 обучаемых параметров.

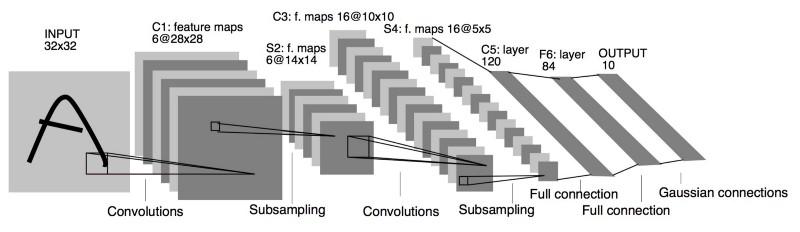


Рис.1. Архитектура нейронной сети LeNet.

Таблица 1. Архитектура сети LeNet-5.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип слоя | Размер ядра | Входной размер | Выходной размер |
| Convolutional | 5х5 | 3x32x32 | 6x32x32 |
| Average pooling | 2х2 | 6x32x32 | 6x16x16 |
| Convolutional | 5х5 | 6x16x16 | 16x12x12 |
| Average pooling | 2х2 | 16x12x12 | 16x6x6 |
| Dense | 1х1 | 1x576 | 1x120 |
| Dense | 1х1 | 1x120 | 1x84 |
| Output dense | 1х1 | 1x84 | 1x10 |

В качестве базы данных использовать MNIST.

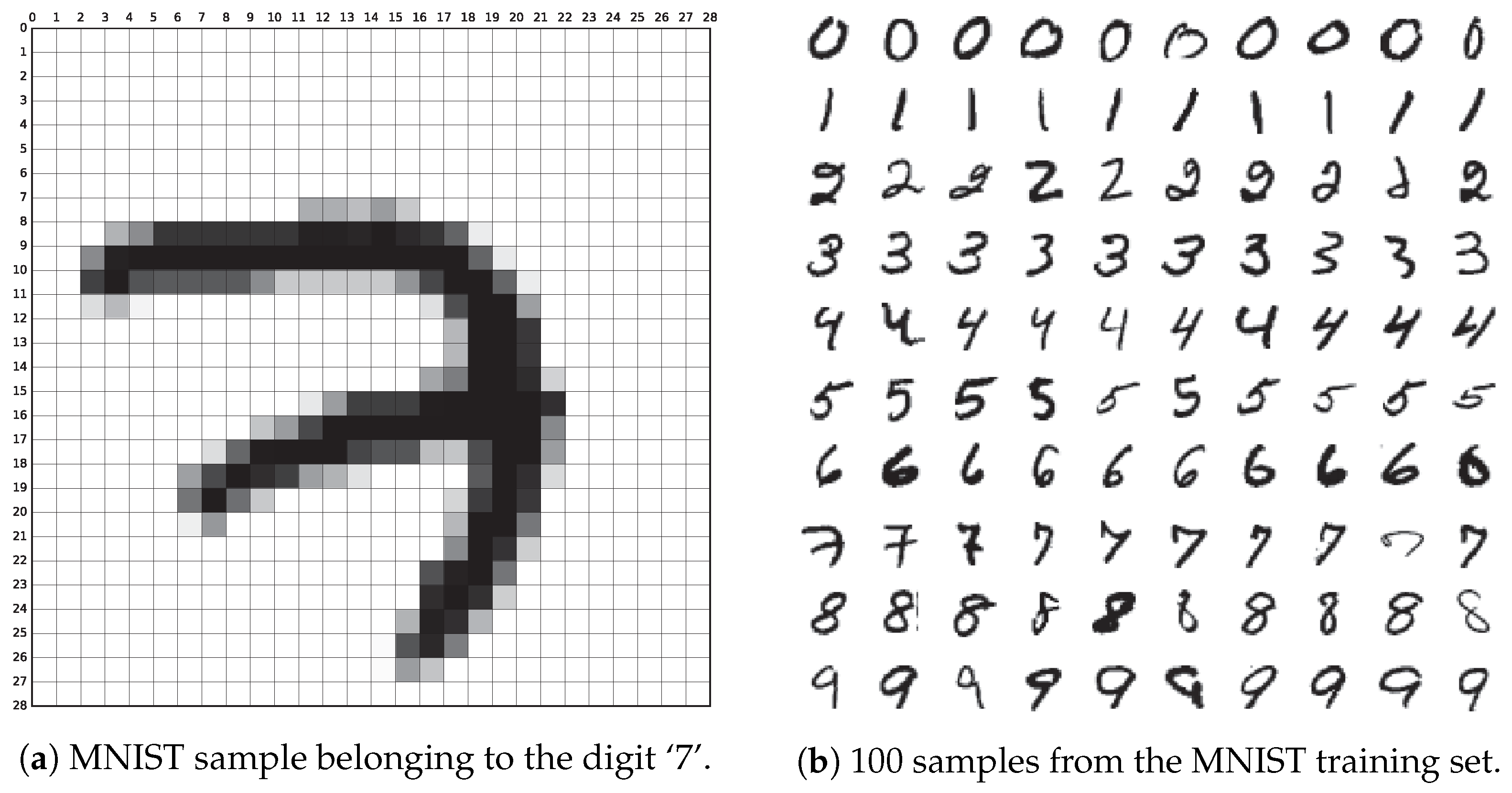


Таблица 2. Вариация гиперпараметров LeNet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Оптимизатор | Кол-во эпох | Скорость обучения | Верность |
| **1** | **SGD** |  |  |  |
| **2** | **AdaDelta** |  |  |  |
| **3** | **NAG** |  |  |  |
| **4** | **Adam** |  |  |  |

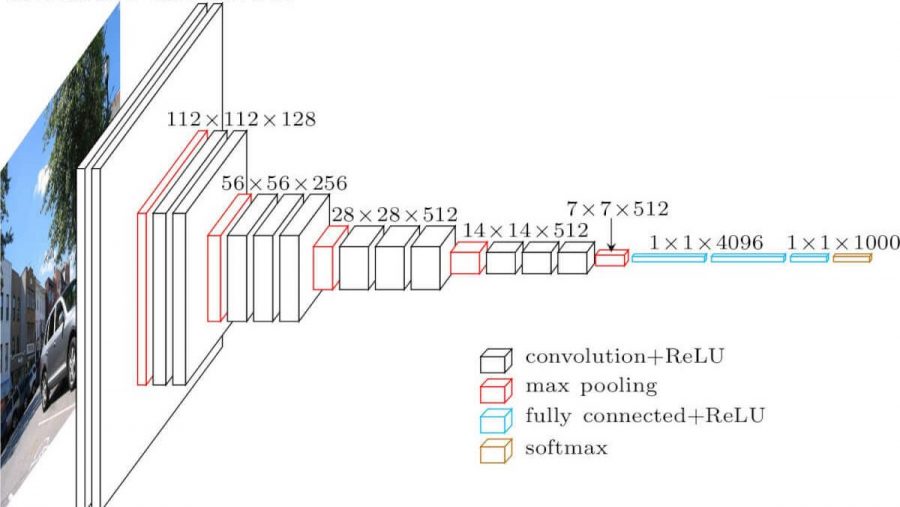
**VGG16**

Рис. 2. Свёрточная нейронная сеть VGG-16.

Таблица 3. Архитектура сети VGG16.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип слоя | Размер ядра | Входной размер | Выходной размер |
| Convolutional | 3х3 | 3x32x32 | 128x32x32 |
| Convolutional | 3х3 | 128x32x32 | 128x32x32 |
| Max-pooling | 2х2 | 128x32x32 | 128x16x16 |
| Convolutional | 3х3 | 128x16x16 | 256x16x16 |
| Convolutional | 3х3 | 256x16x16 | 256x16x16 |
| Max-pooling | 3х3 | 256x16x16 | 256x8x8 |
| Convolutional | 3х3 | 256x8x8 | 512x8x8 |
| Convolutional | 3x3 | 512x8x8 | 512x8x8 |
| Max-pooling | 2x2 | 512x8x8 | 512x4x4 |
| Dens | 1x1 | 512x8x8 | 1x1024 |
| Dropout | - | - | - |
| Dens | 1x1 | 1x1024 | 1x1024 |
| Dropout | - | - | - |
| Output dens | 1x1 | 1x1024 | 1x10 |

В качестве базы данных использовать CIFAR-10.

Here are the classes in the dataset, as well as 10 random images from each:

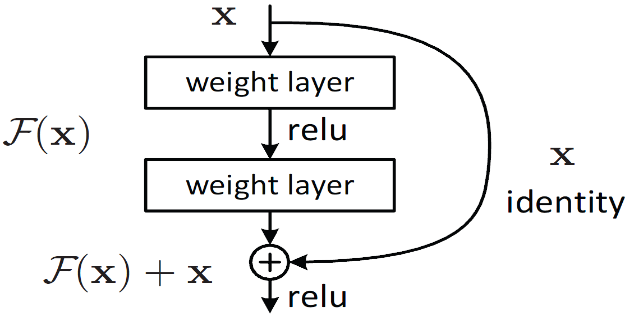
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **airplane** |  |  |  |  |  |  |  |  |  |  |
| **automobile** |  |  |  |  |  |  |  |  |  |  |
| **bird** |  |  |  |  |  |  |  |  |  |  |
| **cat** |  |  |  |  |  |  |  |  |  |  |
| **deer** |  |  |  |  |  |  |  |  |  |  |
| **dog** |  |  |  |  |  |  |  |  |  |  |
| **frog** |  |  |  |  |  |  |  |  |  |  |
| **horse** |  |  |  |  |  |  |  |  |  |  |
| **ship** |  |  |  |  |  |  |  |  |  |  |
| **truck** |  |  |  |  |  |  |  |  |  |  |

The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobile" includes sedans. "Truck" includes only big trucks.

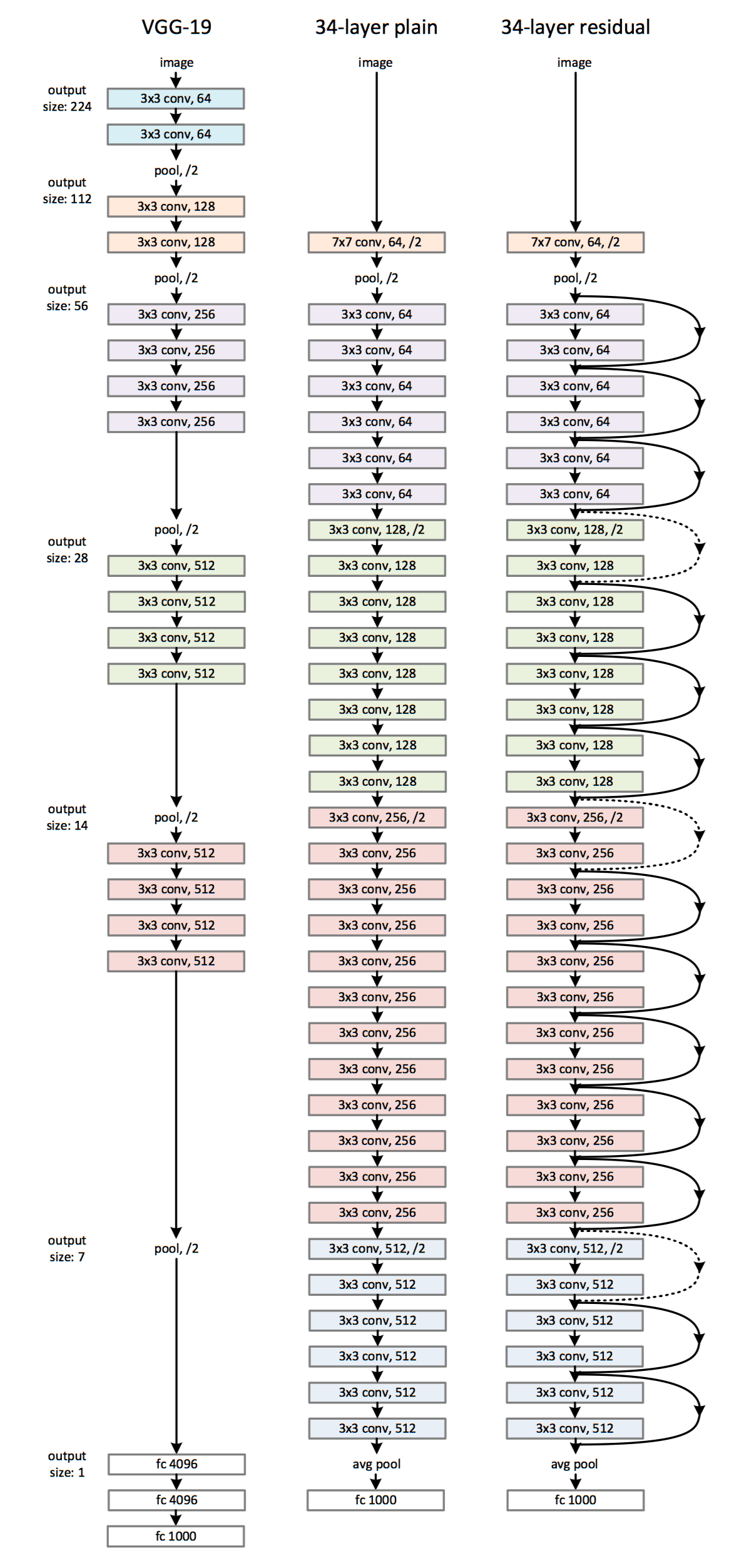
Таблица 4. Вариация гиперпараметров VGG

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Оптимизатор | Кол-во эпох | Скорость обучения | Dropout | Верность |
| **1** | **SGD** |  |  |  |  |
| **2** | **AdaDelta** |  |  |  |  |
| **3** | **NAG** |  |  |  |  |
| **4** | **Adam** |  |  |  |  |

**ResNet (34) «Остаточные» CNN для классификации изображений**



## **Архитектура ResNet**



В качестве базы данных использовать ImageNet.

Таблица 4. Вариация гиперпараметров ResNet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Оптимизатор | Кол-во эпох | Скорость обучения | Dropout | Верность |
| **1** | **SGD** |  |  |  |  |
| **2** | **AdaDelta** |  |  |  |  |
| **3** | **NAG** |  |  |  |  |
| **4** | **Adam** |  |  |  |  |

Для реализации использовать фреймворк PyTorch.

Нужную информацию можно получить:

<https://www.manning.com/books/deep-learning-with-pytorch>

<https://github.com/deep-learning-with-pytorch/dlwpt-code>