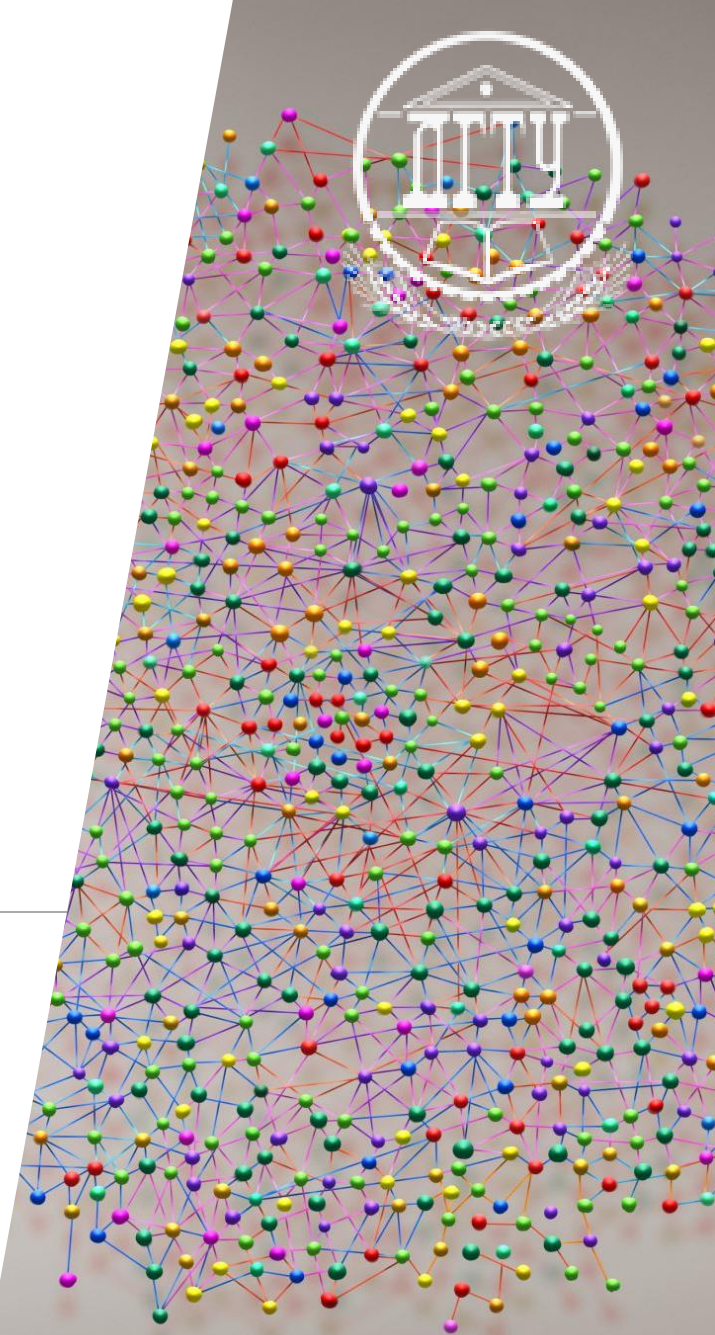


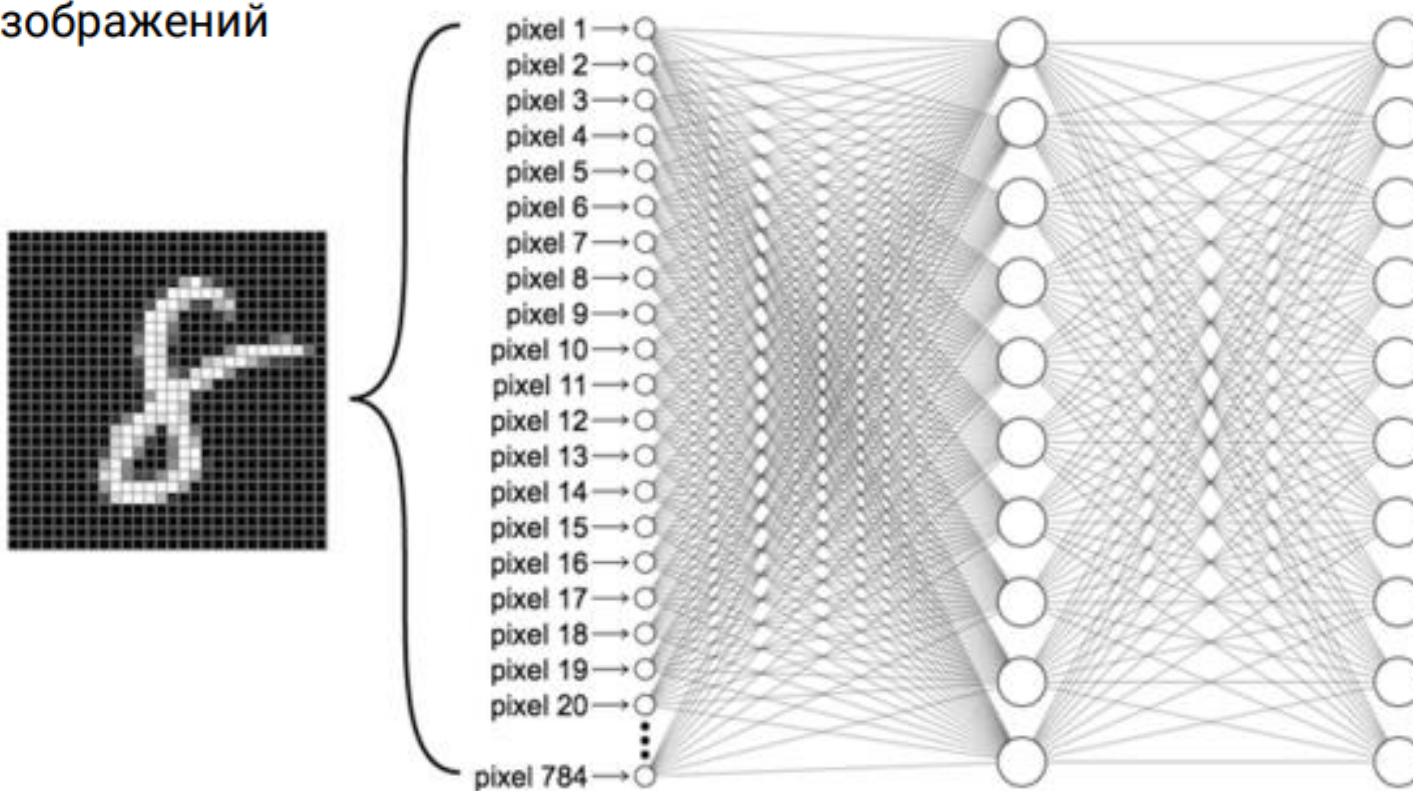
# Сверточные нейронные сети

---

СКЛЯРЕНКО АННА АНАТОЛЬЕВНА



- MLP: взвесим каждый пиксель
- Не учитывается “пространственная” информация
- Не учитывается специфика изображений



# Зрительная кора головного мозга

---

Экспериментируя на животных, David Hubel и Torsten Wiesel выяснили, что одинаковые фрагменты изображения, простейшие формы, активируют одинаковые участки мозга.

**Другими словами, когда котик видит кружочек, то у него активируется зона “А”, когда квадратик, то “Б”.**

В мозгу животных существует область нейронов, которая реагирует на наличие определенной особенности у изображения. Т.е. перед тем как изображение попадает в глубины мозга, оно проходит так называемый **фича-экстрактор**.

# Структура изображения



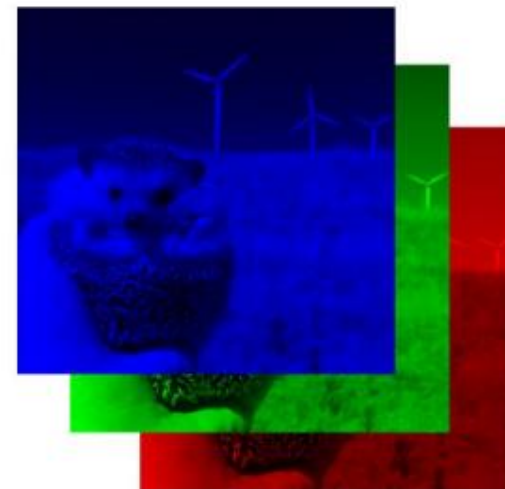
62 62 63 64 65 66 67 67 69 70 71 72 72 73 73 73 72 72 71 70 69 67 66 66 66 65 63 62 61 60 60  
61 62 63 64 66 66 67 68 68 69 70 71 71 72 72 73 72 72 71 71 70 69 68 66 66 65 65 63 62 61 60 60  
61 62 63 64 66 66 68 68 69 70 71 72 73 73 73 72 72 71 71 69 68 67 66 66 65 65 64 63 62 61 61  
61 63 64 64 66 67 68 68 68 69 70 71 71 73 74 73 73 73 71 70 69 68 66 66 65 64 63 62 61 61 60  
61 63 64 65 67 68 69 69 70 71 71 72 55 53 69 72 72 71 71 70 69 68 67 66 65 64 63 62 60 60 60  
63 64 65 66 67 68 69 69 70 71 72 45 4 5 11 48 72 71 71 69 69 68 67 66 65 64 62 62 60 59 59  
63 65 66 66 68 68 69 70 71 71 71 72 18 4 4 7 8 66 71 70 69 68 68 67 66 65 64 63 61 59 59 58  
63 65 67 67 68 69 69 70 71 71 72 64 4 27 24 54 33 20 52 64 68 68 67 66 65 64 63 62 61 59 58 58  
64 65 66 66 68 69 70 71 11 24 24 12 17 24 15 60 37 43 36 52 66 68 67 66 65 64 63 61 60 59 58 57  
65 66 67 67 68 69 71 10 0 0 0 5 34 36 12 47 34 17 29 54 43 63 67 66 65 64 63 62 60 59 58 57  
64 65 66 66 68 69 36 0 0 0 5 5 7 16 19 4 47 44 27 24 40 67 66 66 65 65 64 63 61 60 59 58 57  
63 64 65 65 67 30 0 0 0 5 5 5 0 8 9 20 27 51 78 41 44 66 65 65 65 64 63 62 60 59 58 57  
63 64 65 65 34 5 5 5 5 5 5 5 4 19 6 7 54 64 70 59 65 65 64 64 64 63 62 61 60 59 57 56  
63 64 64 65 14 5 6 5 5 4 5 4 18 7 5 4 19 10 11 65 64 64 64 63 61 66 62 61 60 59 58 56  
63 64 64 65 53 7 4 5 6 6 7 10 6 5 5 4 21 24 18 64 64 64 63 62 64 65 62 62 60 59 58 57  
64 64 64 64 65 50 4 4 4 5 11 18 0 0 4 0 35 16 20 66 64 64 63 61 72 67 63 62 61 59 58 57  
64 64 64 64 65 46 4 4 4 5 6 9 8 5 39 10 43 56 29 57 64 64 63 61 70 67 62 64 65 59 59 57  
64 64 64 65 66 22 5 4 4 5 6 6 6 18 66 20 57 60 46 36 75 70 62 61 70 67 62 61 60 59 58 58  
49 50 62 65 57 5 5 6 5 6 6 6 6 41 59 29 60 58 44 22 63 71 72 60 69 68 61 60 58 59 59 58  
42 52 57 52 28 5 5 5 5 5 5 5 5 70 50 43 61 62 64 39 42 64 60 62 56 63 65 65 67 61 53 53  
52 32 32 33 6 5 5 5 5 5 6 6 11 39 21 30 51 50 45 46 18 30 36 33 23 44 70 71 51 42 27 31  
50 50 51 39 5 5 5 5 6 5 6 6 42 60 28 34 42 30 43 37 26 29 40 26 29 26 35 42 35 33 18 19  
52 53 51 22 5 5 5 5 6 5 6 5 44 56 17 51 54 53 54 56 51 22 54 54 55 55 54 53 53 53 52 52  
54 54 53 8 5 5 5 5 5 6 13 52 42 21 51 54 51 49 49 50 22 41 45 42 42 41 40 41 44 43 42  
52 52 54 36 8 5 5 6 6 6 6 28 55 32 30 54 53 51 51 51 44 25 51 51 49 49 50 49 48 46 46  
54 54 52 53 30 7 5 6 6 5 6 40 54 29 52 51 53 56 55 52 52 51 38 52 52 50 49 46 46 45 46 47  
51 52 51 53 27 14 5 4 5 4 7 47 51 21 39 49 47 49 52 52 52 49 33 31 48 46 47 47 47 46 46 43  
48 50 51 53 25 14 17 8 4 4 17 46 40 18 43 47 46 49 52 54 53 53 54 11 50 49 46 47 47 47 47 45  
49 49 49 49 22 12 20 24 6 14 35 51 30 48 48 50 51 51 49 51 51 52 50 41 58 48 47 47 47 45 45  
51 49 50 50 22 13 19 36 13 12 42 50 40 73 50 50 50 49 48 49 49 48 49 45 51 46 44 44 44 42 45 47  
47 49 49 47 20 16 26 39 21 15 30 48 42 61 47 48 51 47 50 51 51 51 49 47 47 52 47 47 44 43 45 46  
48 50 48 52 19 13 33 36 18 18 36 49 51 54 47 47 49 46 46 49 49 49 49 47 44 53 44 44 46 46 45



2],  
3],  
3],  
2],  
8],  
2],  
0, 166, 161, 157],



||



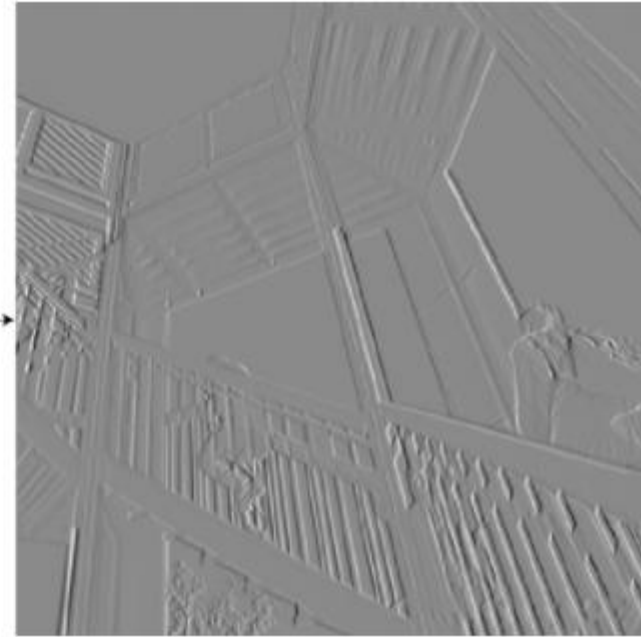
# Фильтр Собеля

---



$$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

Horizontal Sobel kernel





# Примеры фильтров



Ядро

-1	-1	-1
-1	8	-1
-1	-1	-1

\*

=

Детектирование краев



Исходная  
фотография

Суммируется в 0 (черный цвет),  
если в блоке однородный цвет

# Примеры фильтров

---

$*$

0	-1	0
-1	5	-1
0	-1	0

$=$



Увеличение резкости

$* \frac{1}{9}$

1	1	1
1	1	1
1	1	1

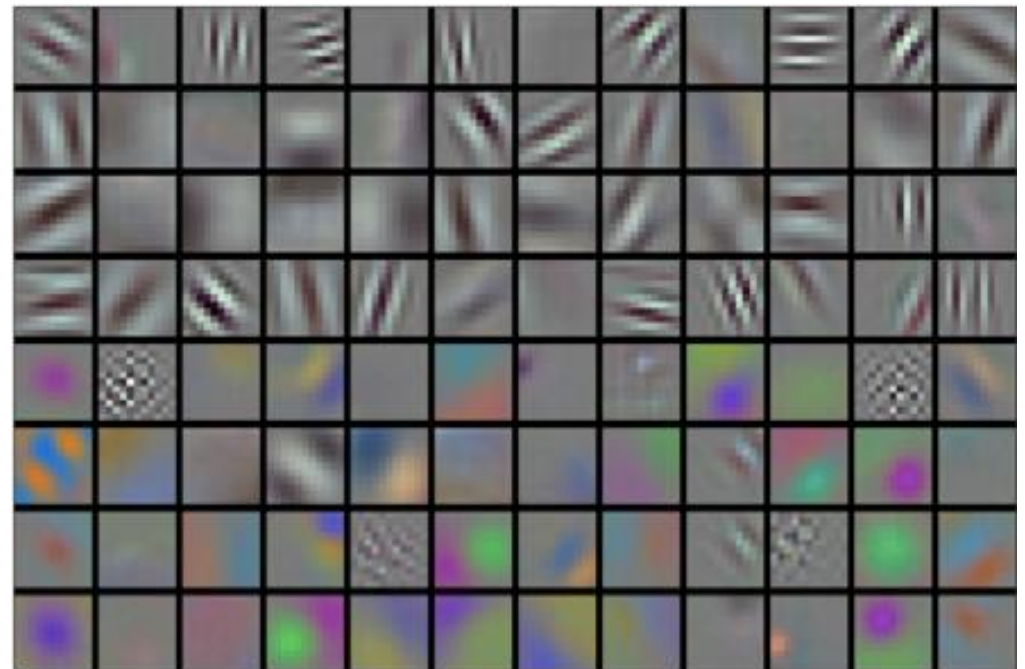
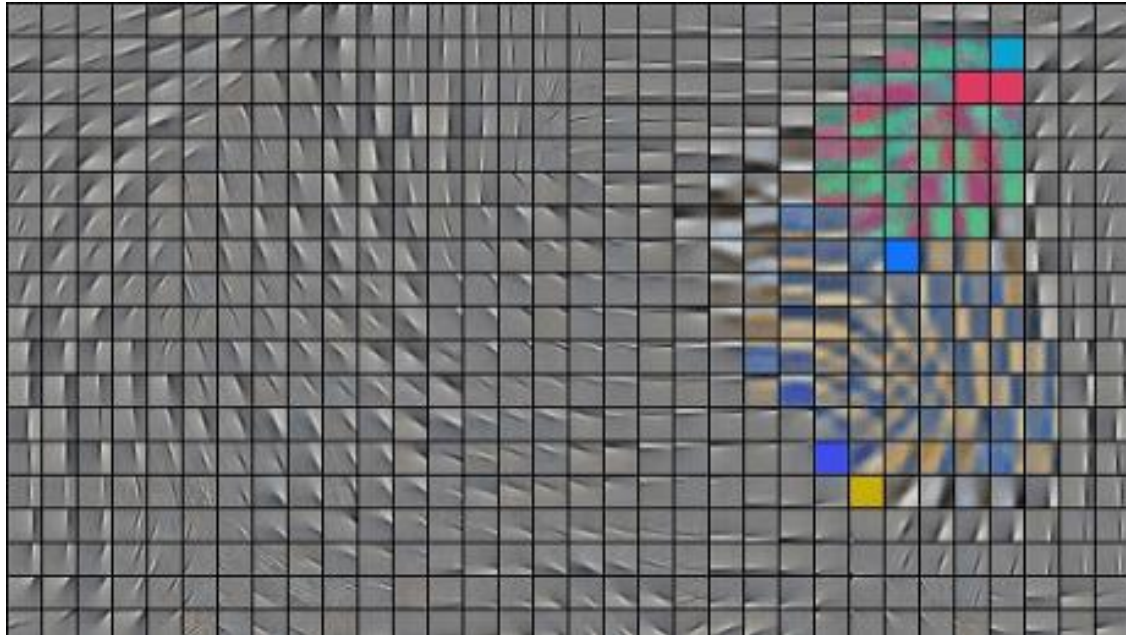
$=$



Размытие

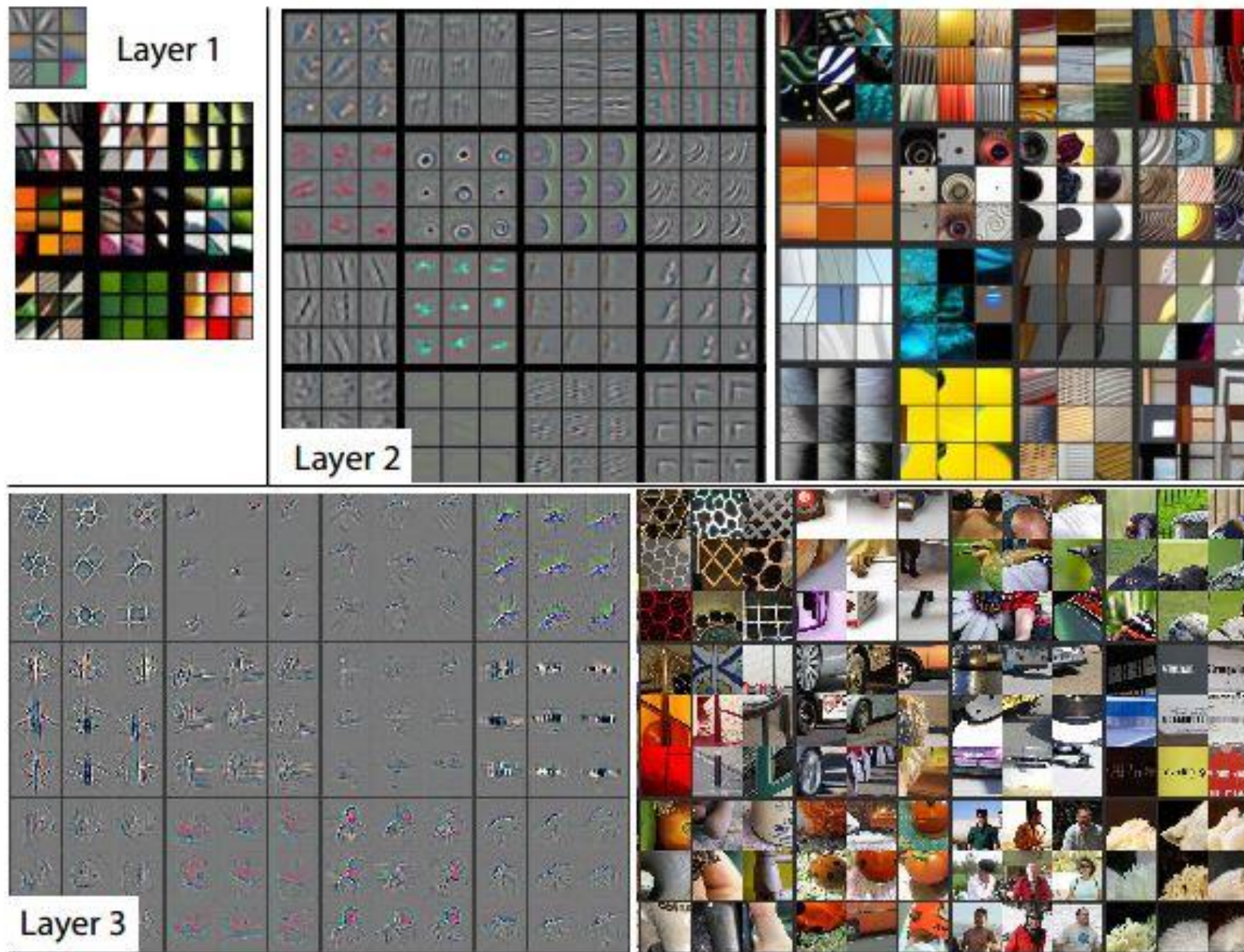
# Матрица Kernel

---

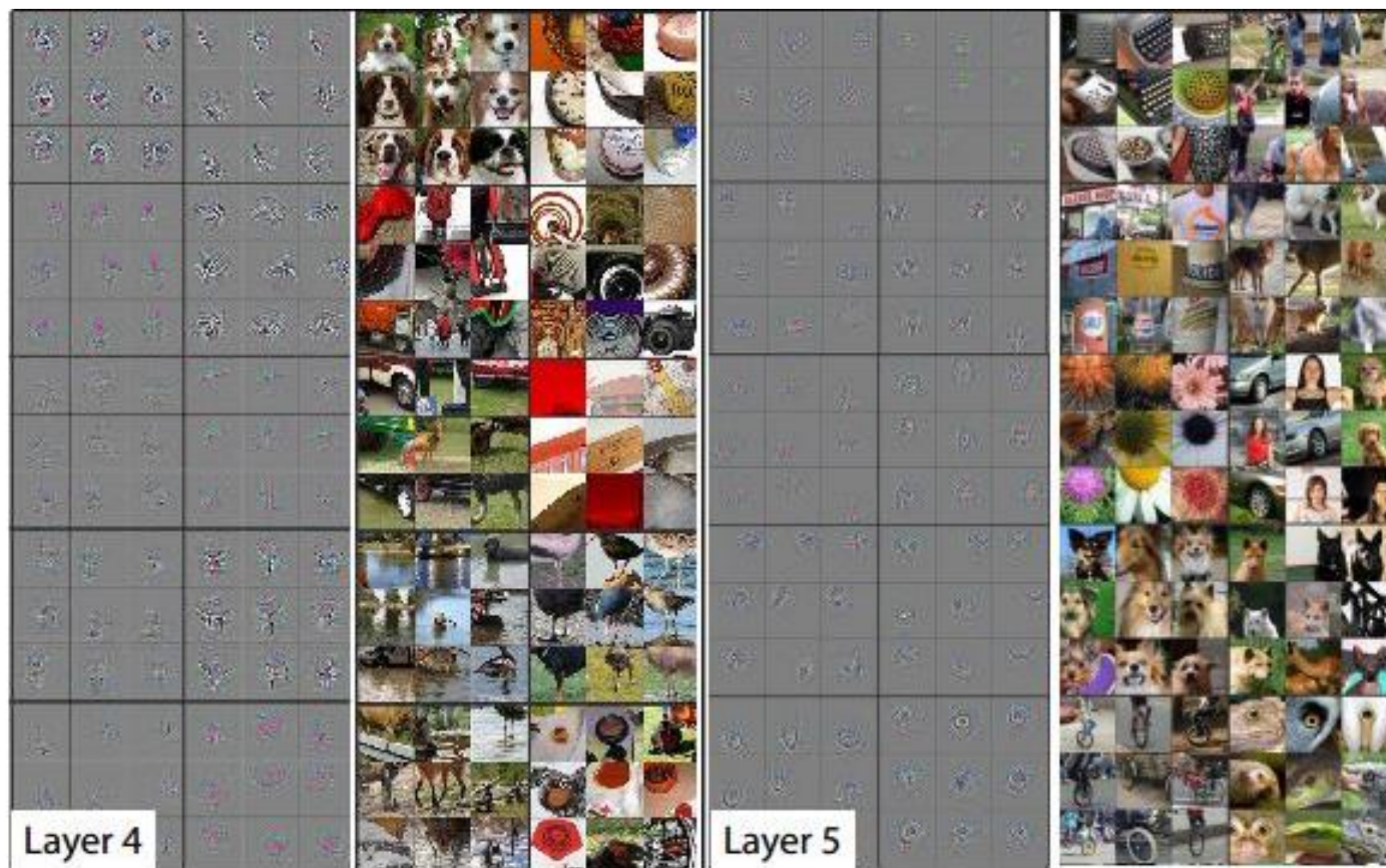




# Фича- экстрактор







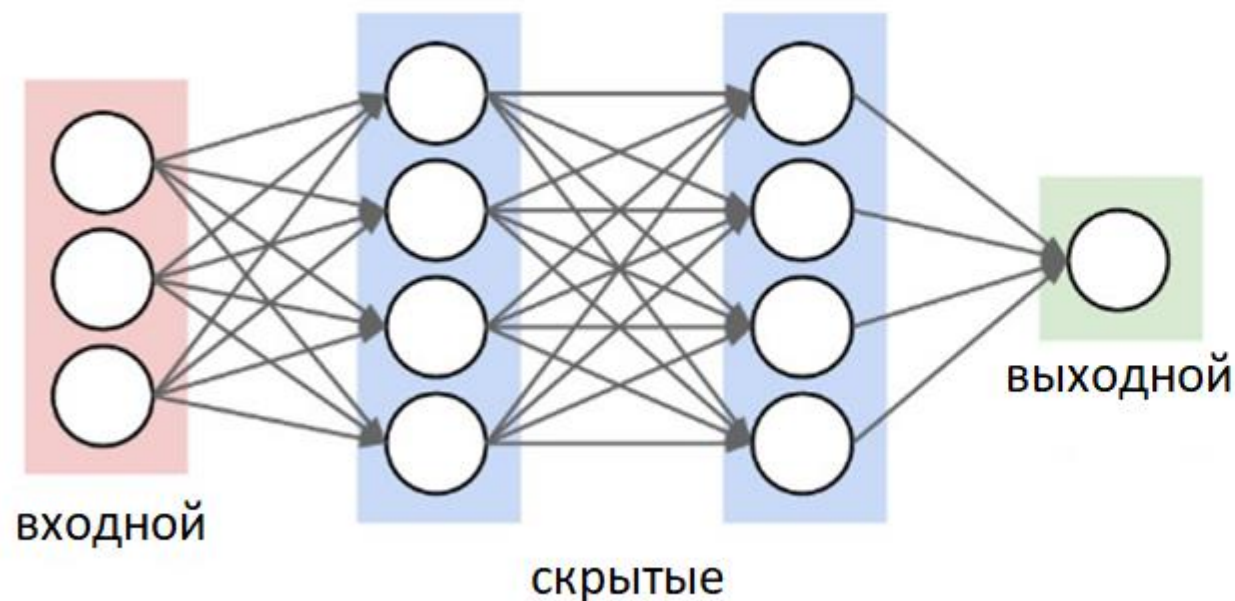
Подбор Kernel в процессе обучения

# Сверточные нейронные сети

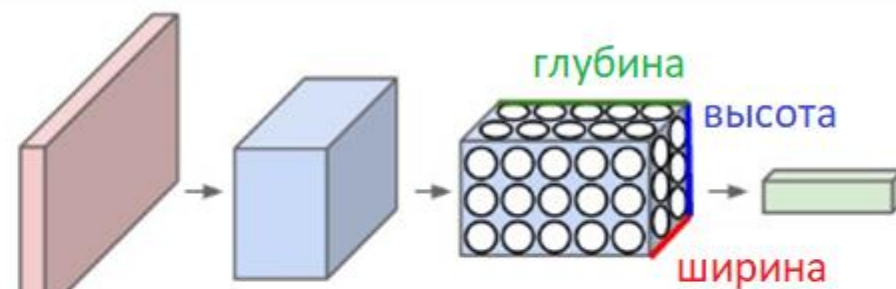
Сверточные нейронные сети (convolutional neural networks, CNN) — переиспользование одних и тех же частей нейронной сети для работы с разными маленькими, локальными участками входов (1988). Прототип коры головного мозга.

Основная задача: обработка изображений, автоматизация извлечения признаков

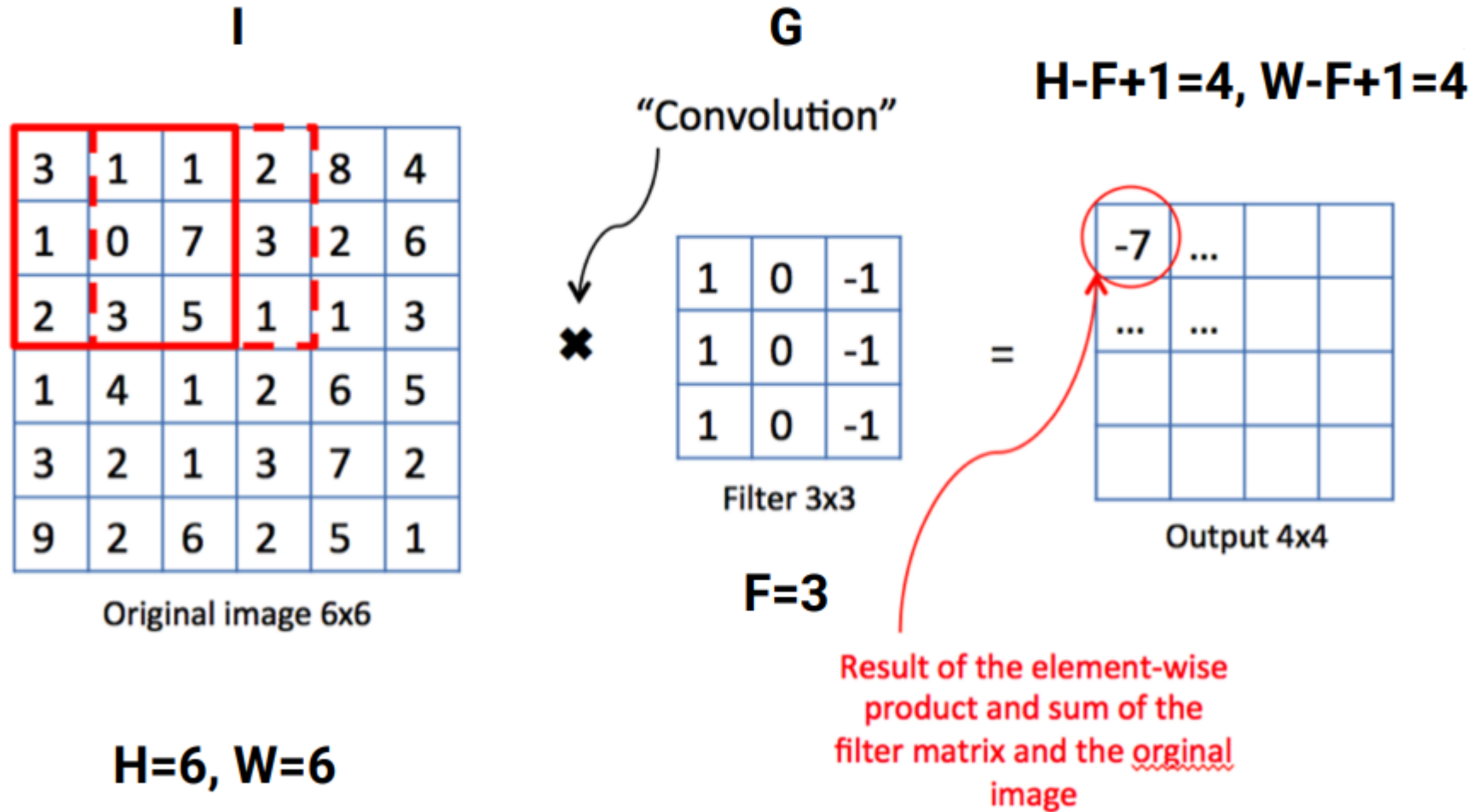
Стандартная нейронная сеть



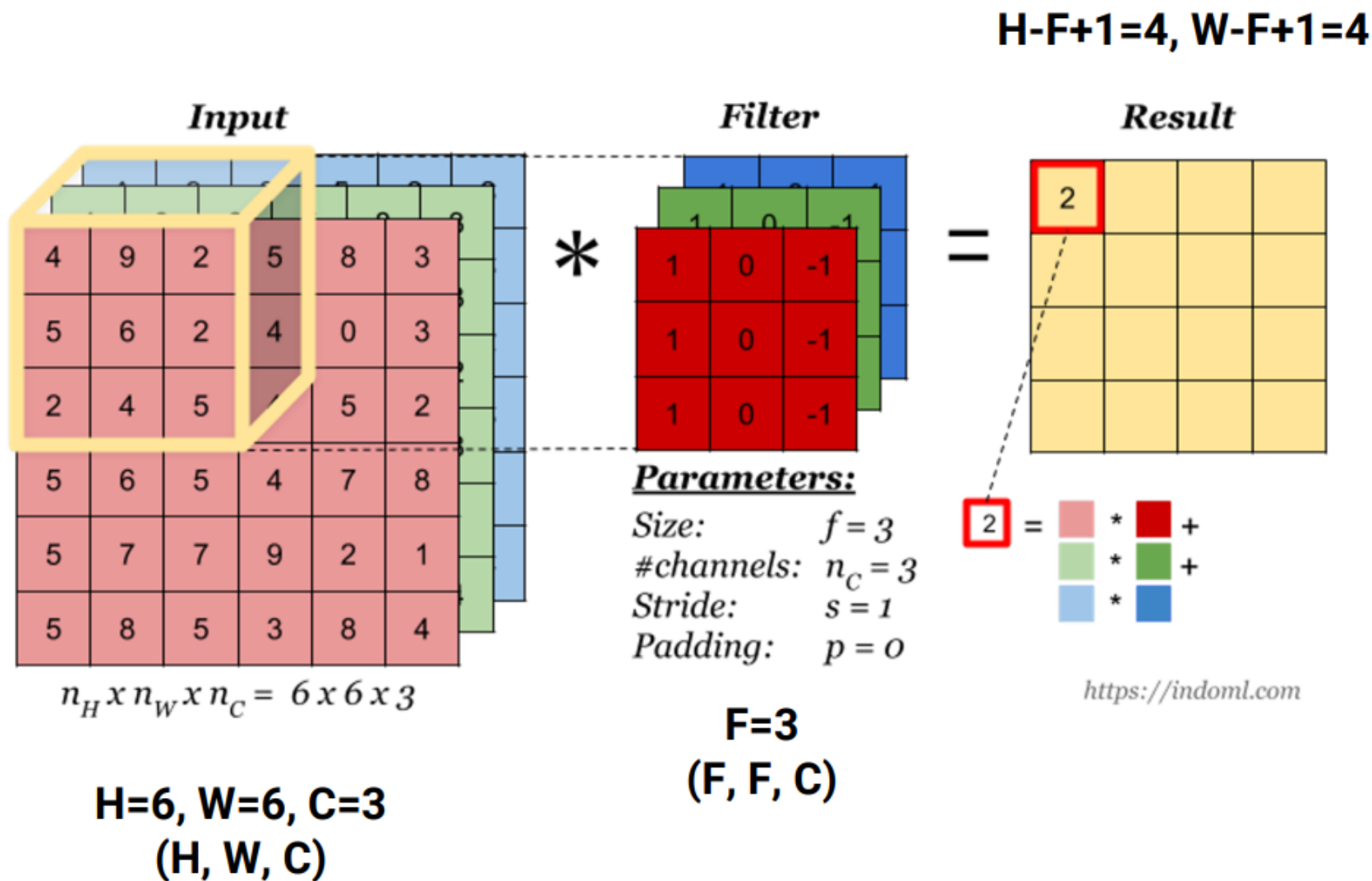
Свёрточная нейронная сеть



# Свёртка ч/б изображения с фильтром FxF

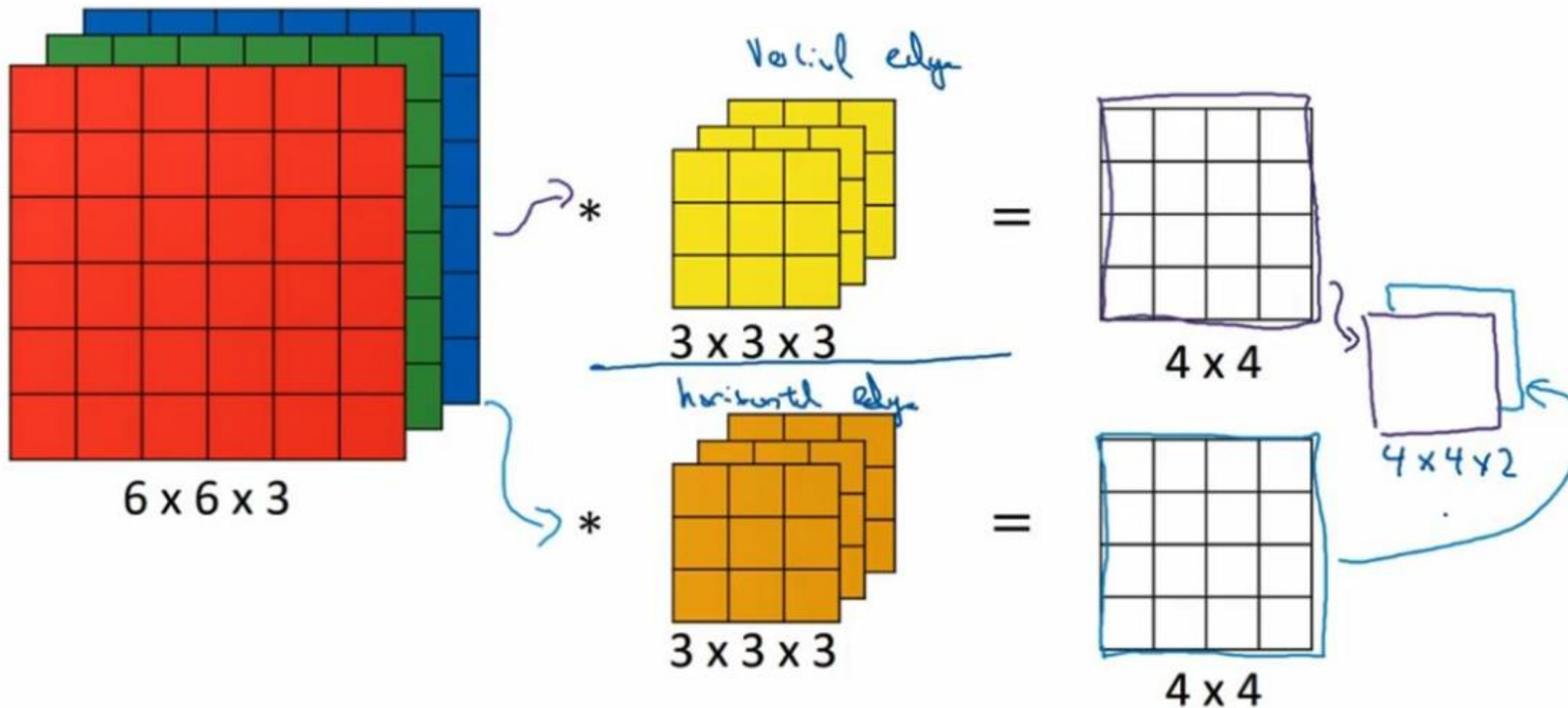


# Свёртка цветного изображения с фильтром $F \times F$





# Свёртка цветного изображения с фильтром



# Ядро свертки 1 на 1

1	2	3	6	5	8
3	5	5	1	3	4
2	1	3	4	9	3
4	7	8	5	7	9
1	5	3	7	4	8
5	4	9	8	3	5

$6 \times 6$

\*

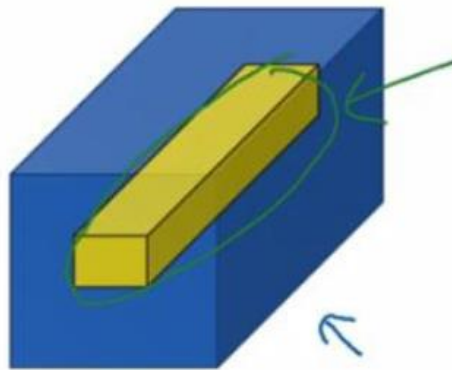
2

=

2	4	6	...		

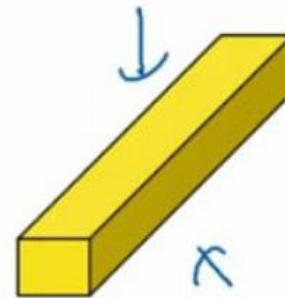
<https://blog.csdn.net/Tomxiaodai>

Сжатие модели



$6 \times 6 \times 32$

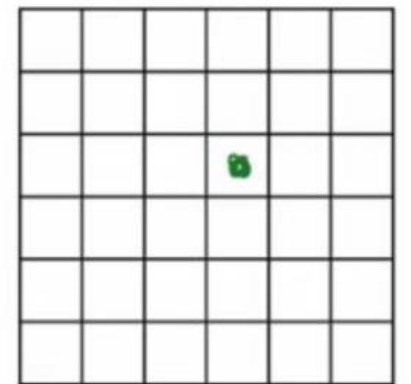
\*



$1 \times 1 \times 32$

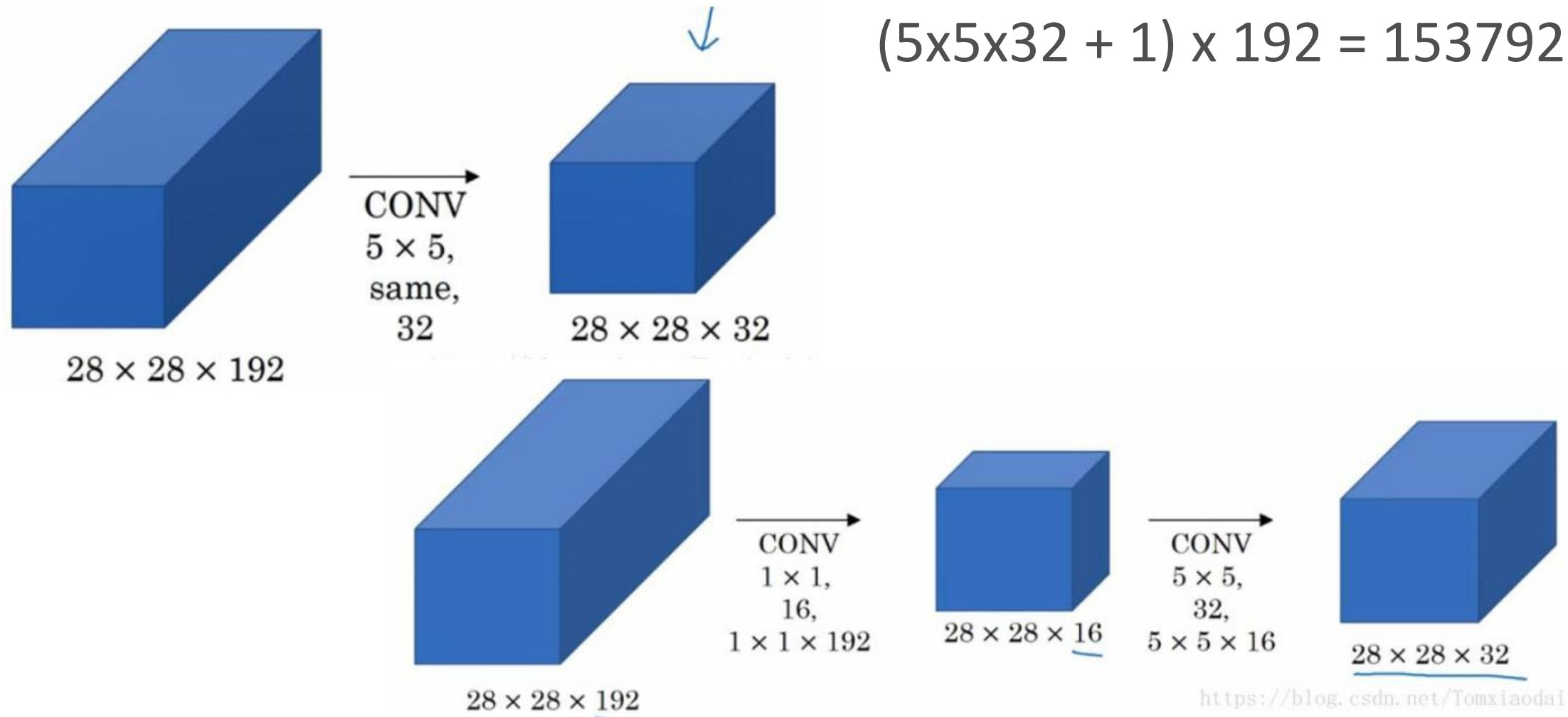
=

ReLU



<https://blog.csdn.net/Tomxiaodai>  $6 \times 6 \times \# \text{filters}$

# Ядро свертки 1 на 1



$$(1 \times 1 \times 16 + 1) \times 192 + (5 \times 5 \times 32 + 1) \times 16 = 16080$$

# Принципы работы CNN

## 1. Фильтр

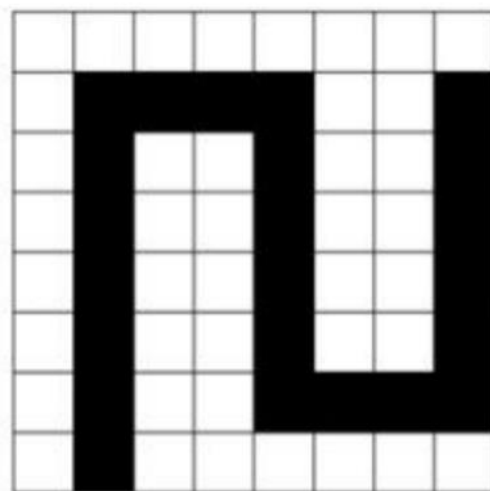


Исходное изображение

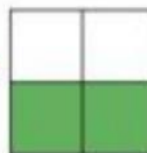
0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0



Фильтр



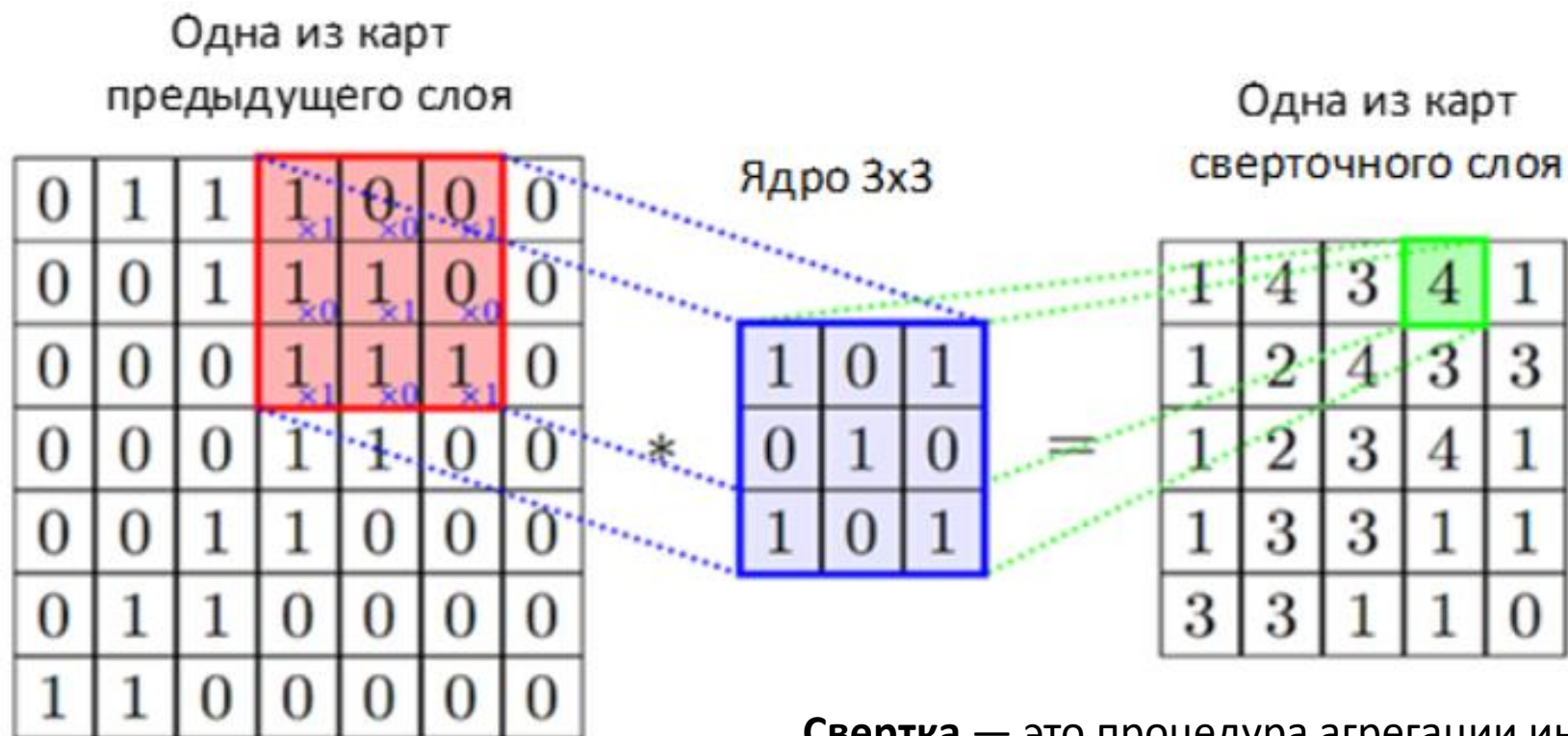
Исходное изображение



Фильтры  
определители  
признаков

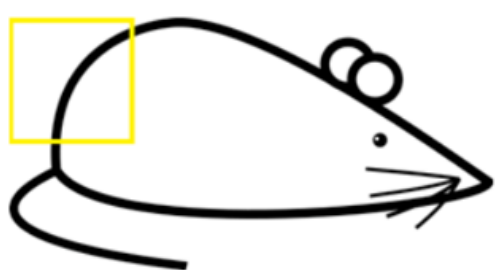
# Принципы работы CNN

## 2. Формирование карты признаков





# Принцип работы CNN



Фрагмент исходного изображения

0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Пиксельное представлени

\*



Изображение фильтра

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Пиксельное представление фильтра

$$\text{Результат} = (50 \cdot 30) + (50 \cdot 30) + (50 \cdot 30) + (20 \cdot 30) + (50 \cdot 30) = 6600$$



0	0	0	0	0	0	0
0	40	0	0	0	0	0
40	0	40	0	0	0	0
40	20	0	0	0	0	0
0	50	0	0	0	0	0
0	0	50	0	0	0	0
25	25	0	50	0	0	0

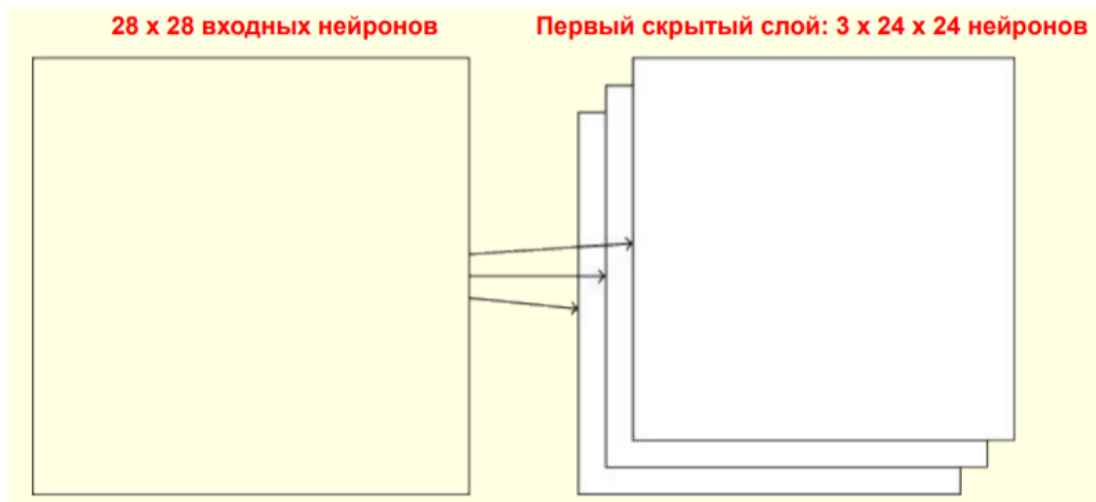
\*

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0







Результат = 0

# Принципы работы CNN

Несколько карт признаков

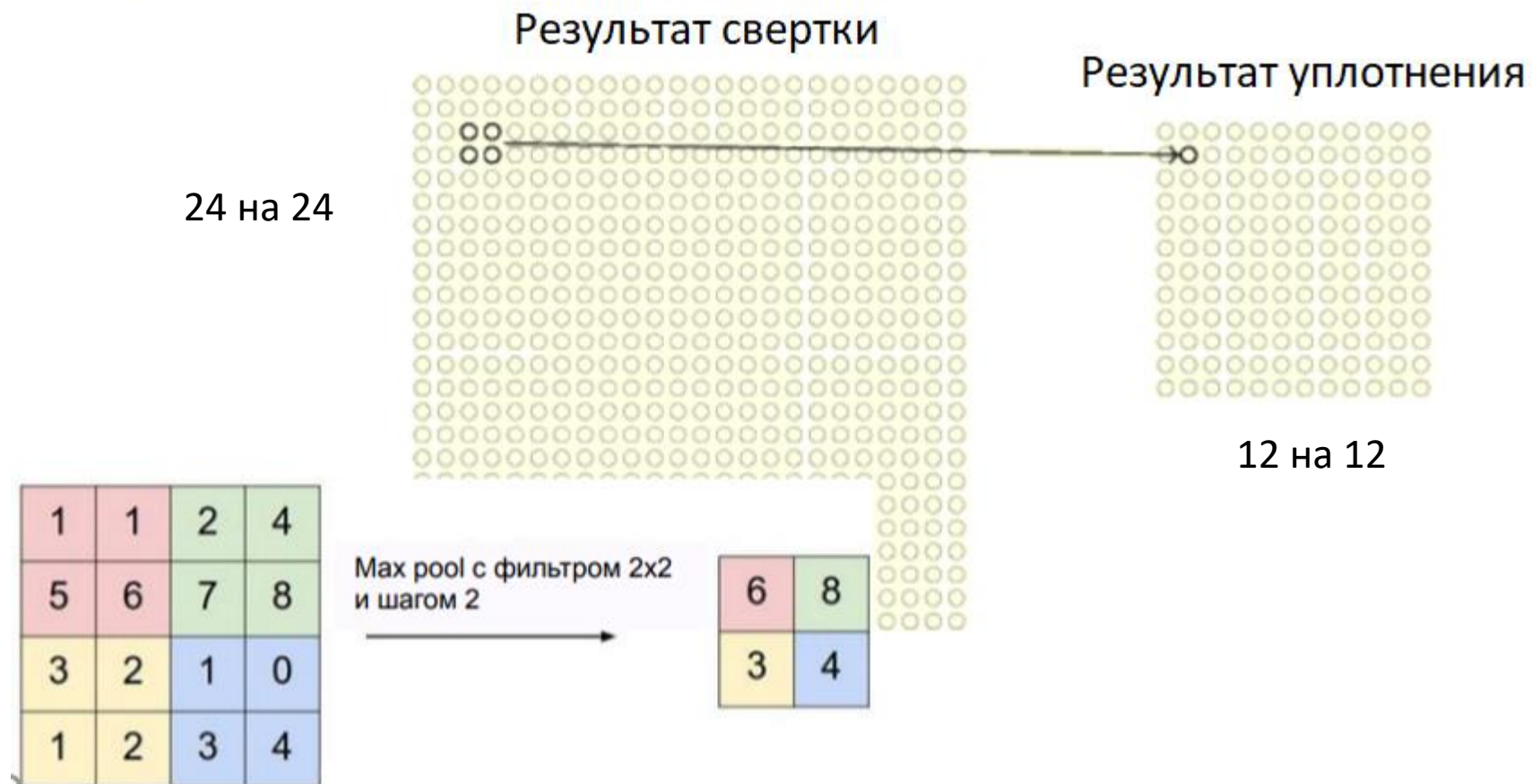


Фильтр 5x5

	$x \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$x \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$x \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

# Принципы работы CNN

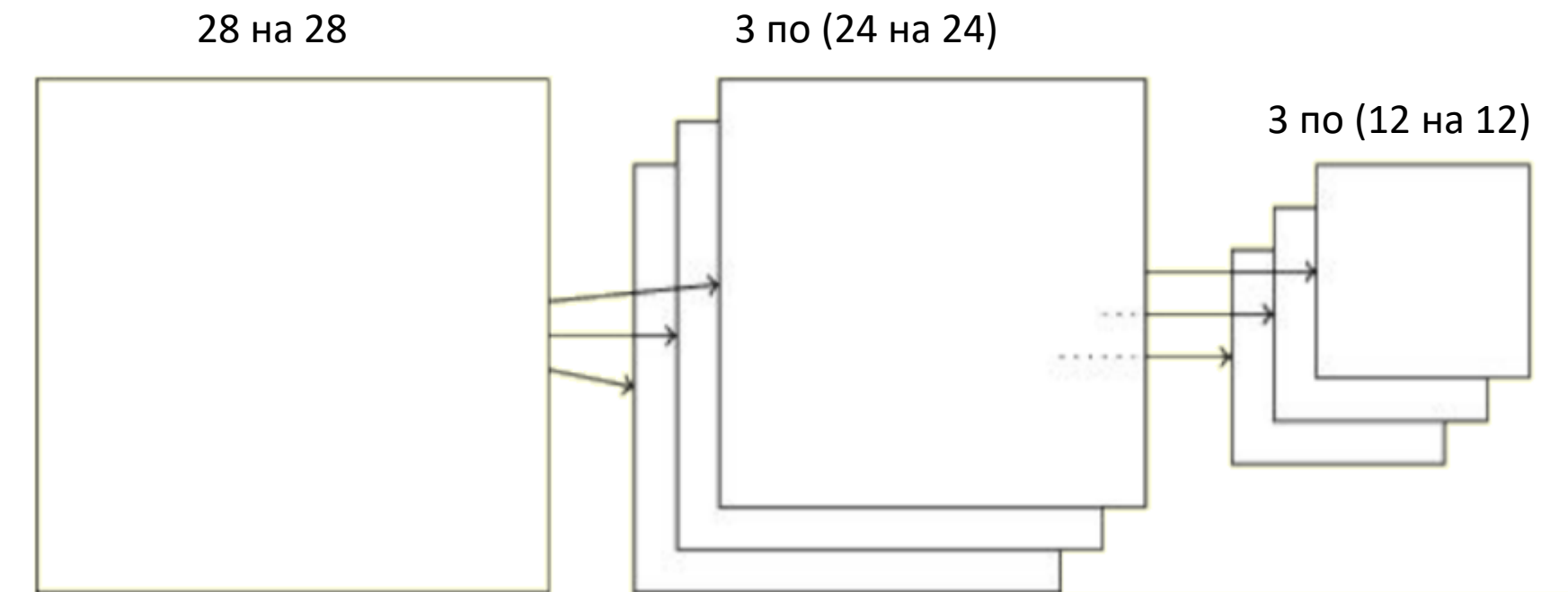
3. Уплотнение (объединение, pooling) карты признаков сверточного слоя => уплотненная карта



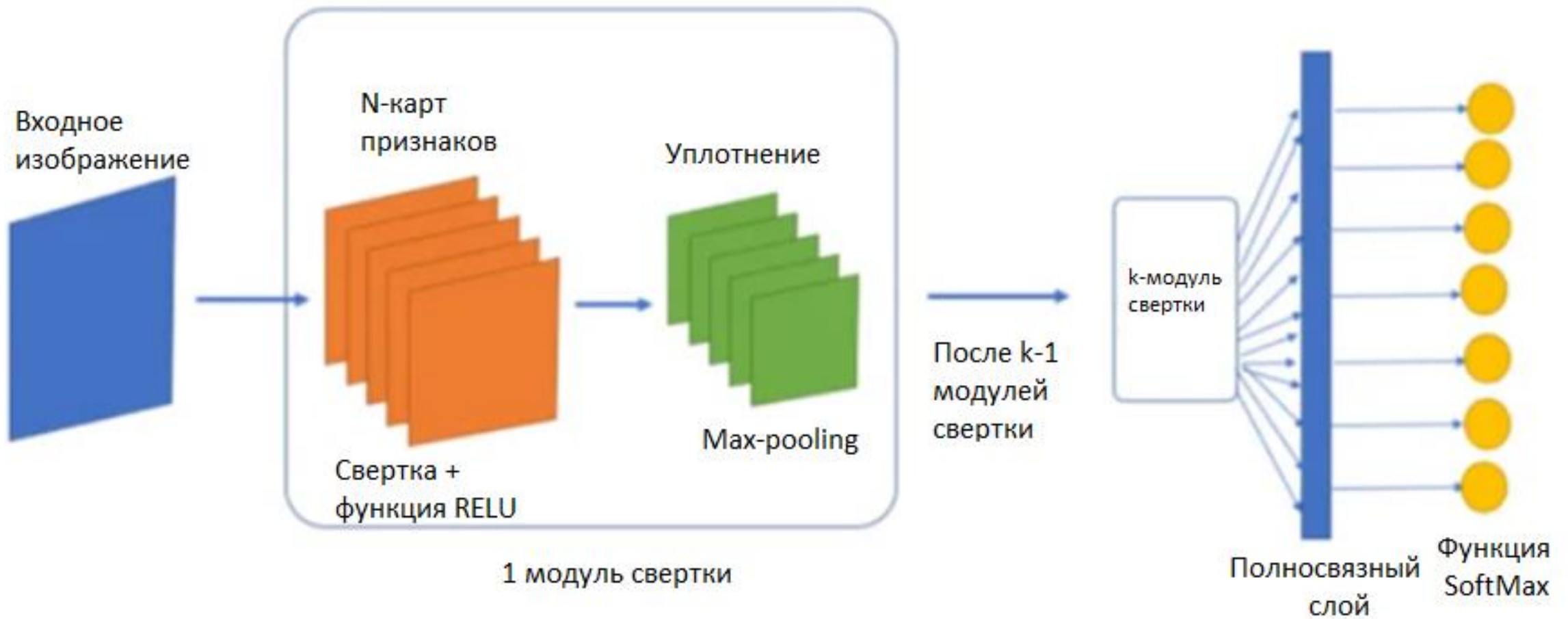
# Принцип работы CNN

---

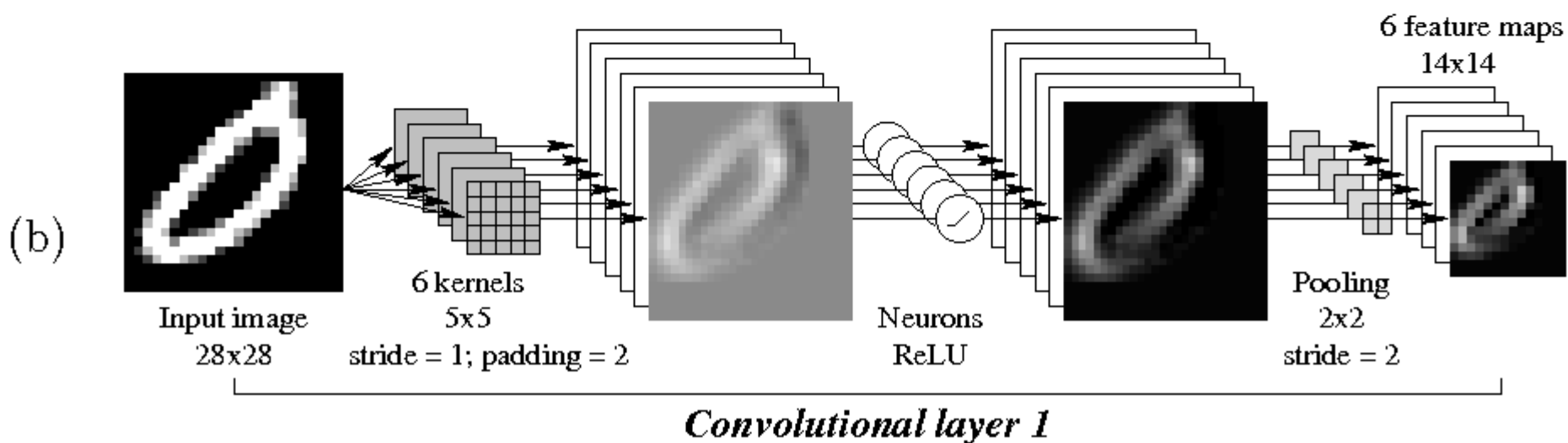
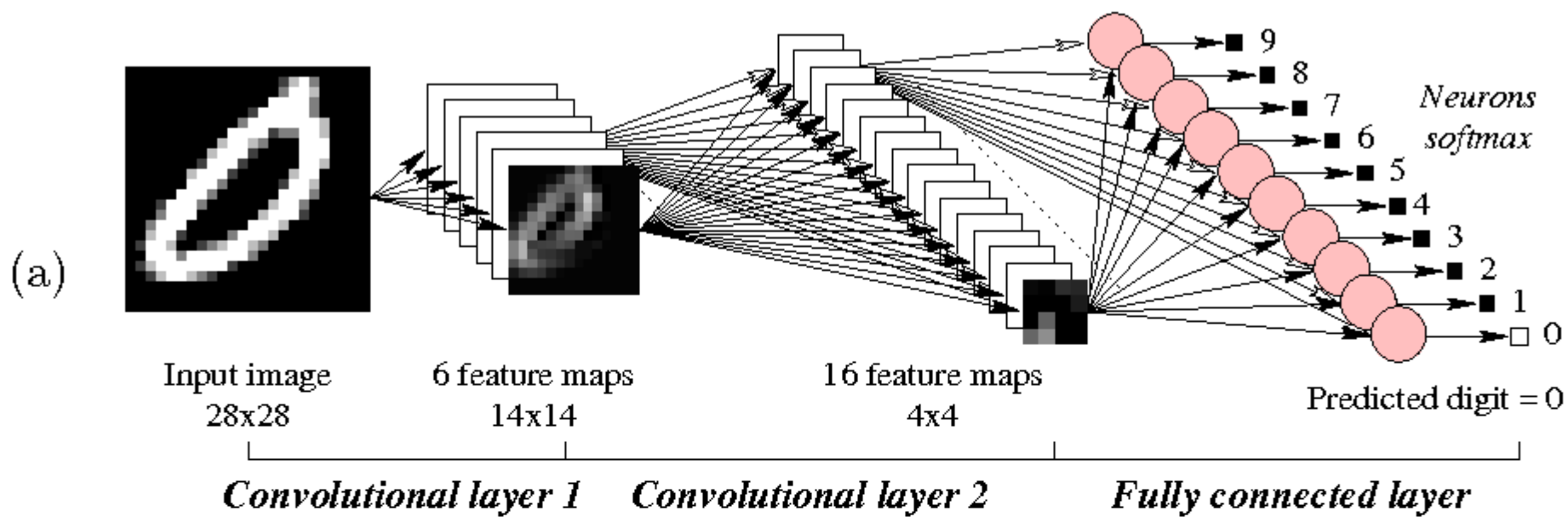
Несколько карт признаков



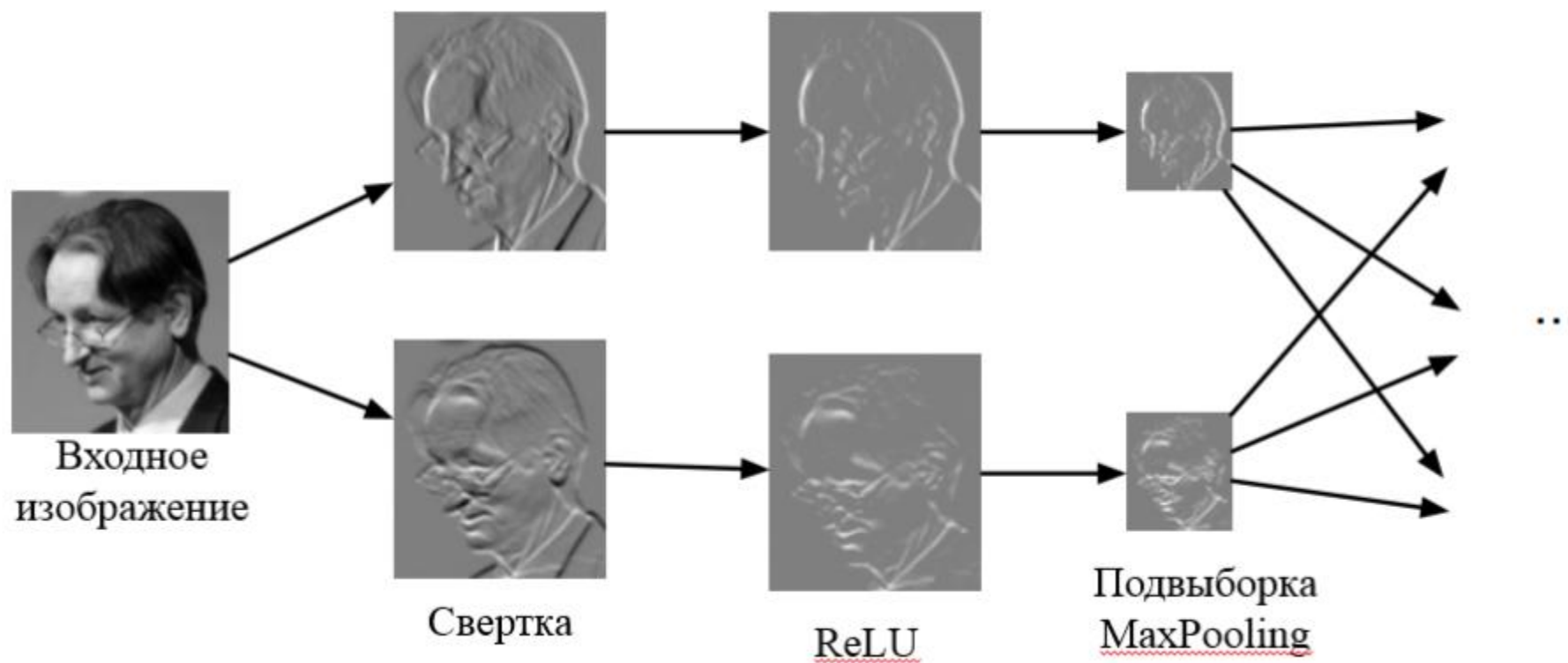
# Общая архитектура сверточной НС



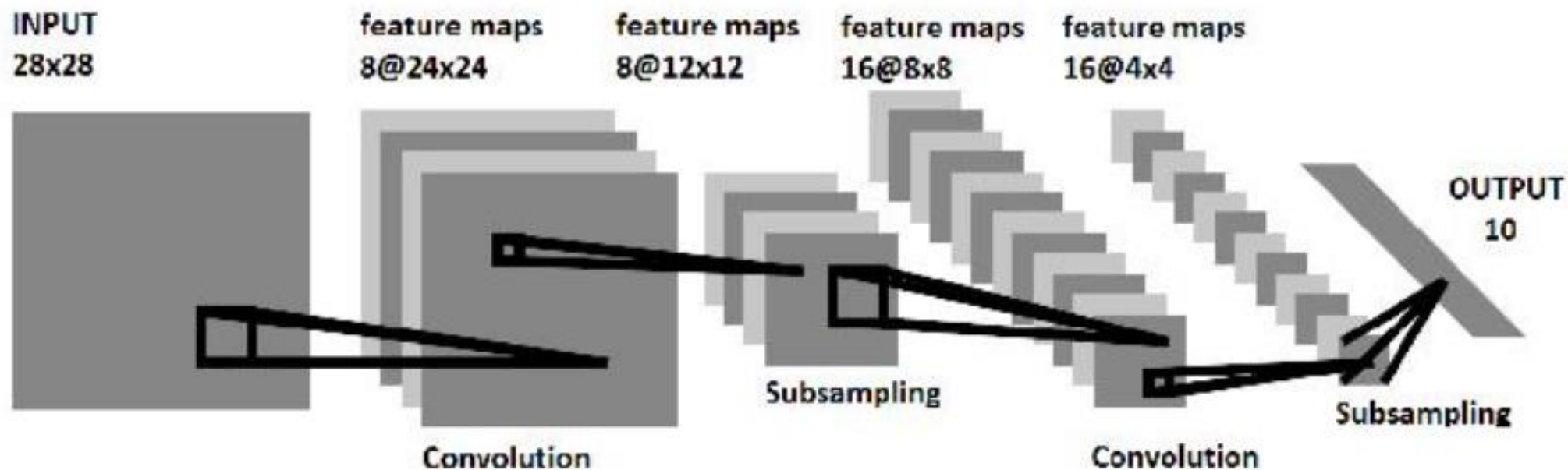




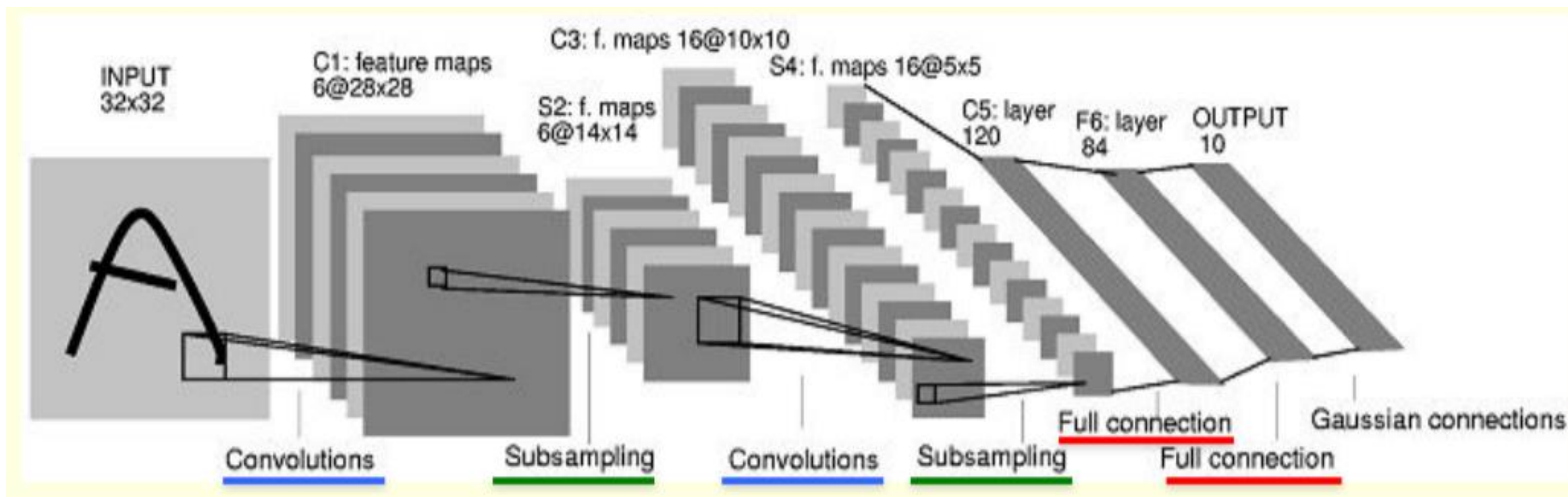
# Пример работы CNN



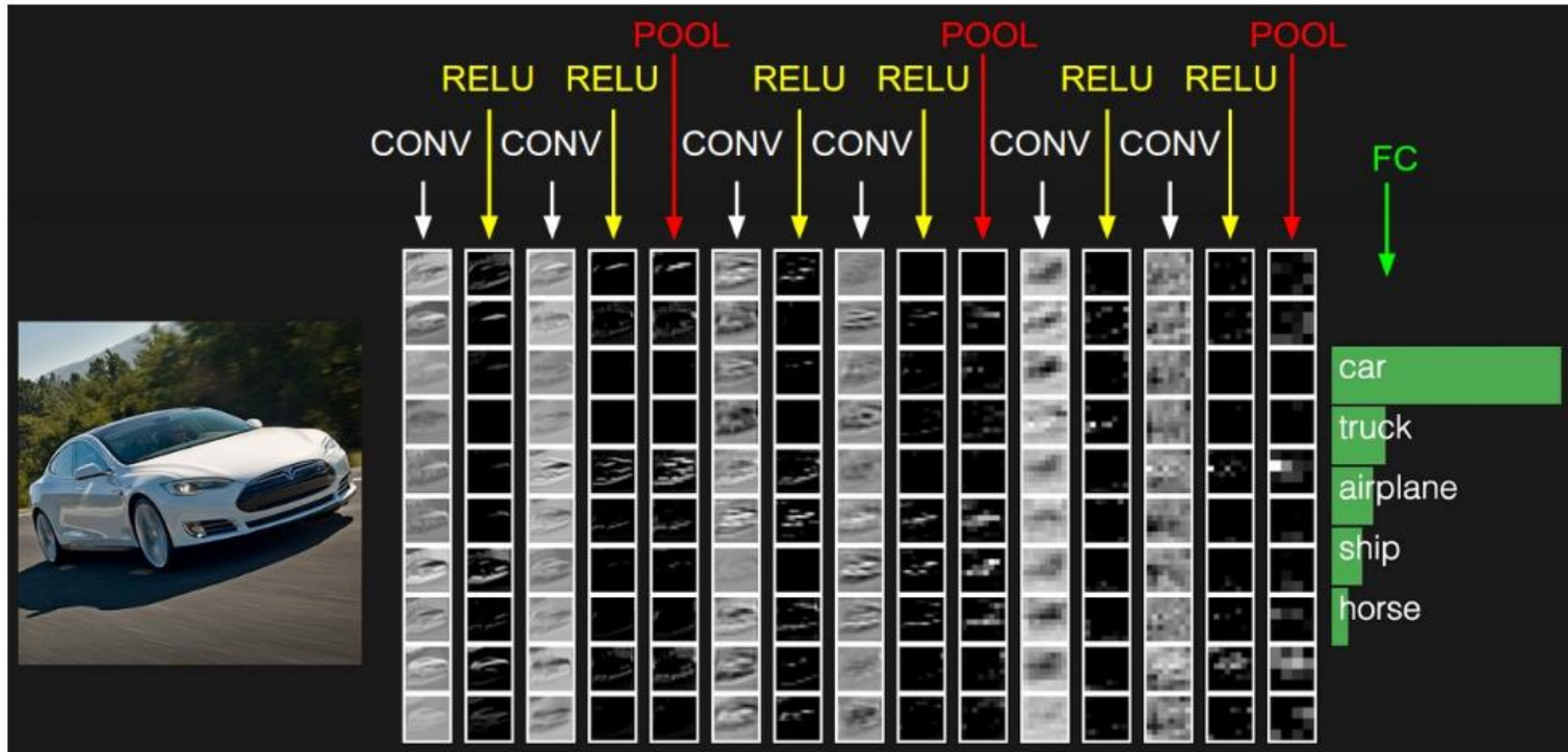
# Упрощенная архитектура сети Яна Лекуна для MNIST



# Архитектура сверточной нейронной сети Яна Лекуна



# Архитектура сверточной нейронной сети



INPUT  $\rightarrow$   $[[\text{CONV} \rightarrow \text{RELU}]^*N \rightarrow \text{POOL?}]^*M \rightarrow [\text{FC} \rightarrow \text{RELU}]^*K \rightarrow \text{FC}$



# Дополнительные слои

---

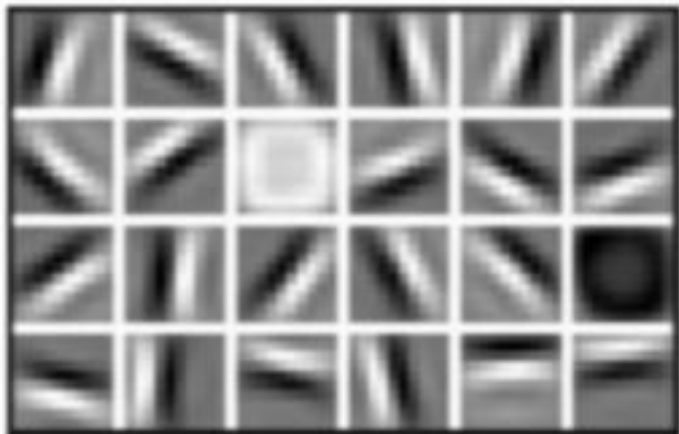
**Слой пакетной нормализации** - это приём, который помогает упростить обучение очень глубоких нейронных сетей путём стандартизации входов в слой для каждого мини-пакета. Стандартизация входов стабилизирует процесс обучения и таким образом уменьшает количество эпох обучения глубоких нейросетей.

**Слой исключения (Dropout)** - это приём регуляризации, который справляется с переобучением и чрезмерным обобщением.

# Примеры признаков

---

Низкоуровневые признаки



Линии и границы

Признаки средних уровней



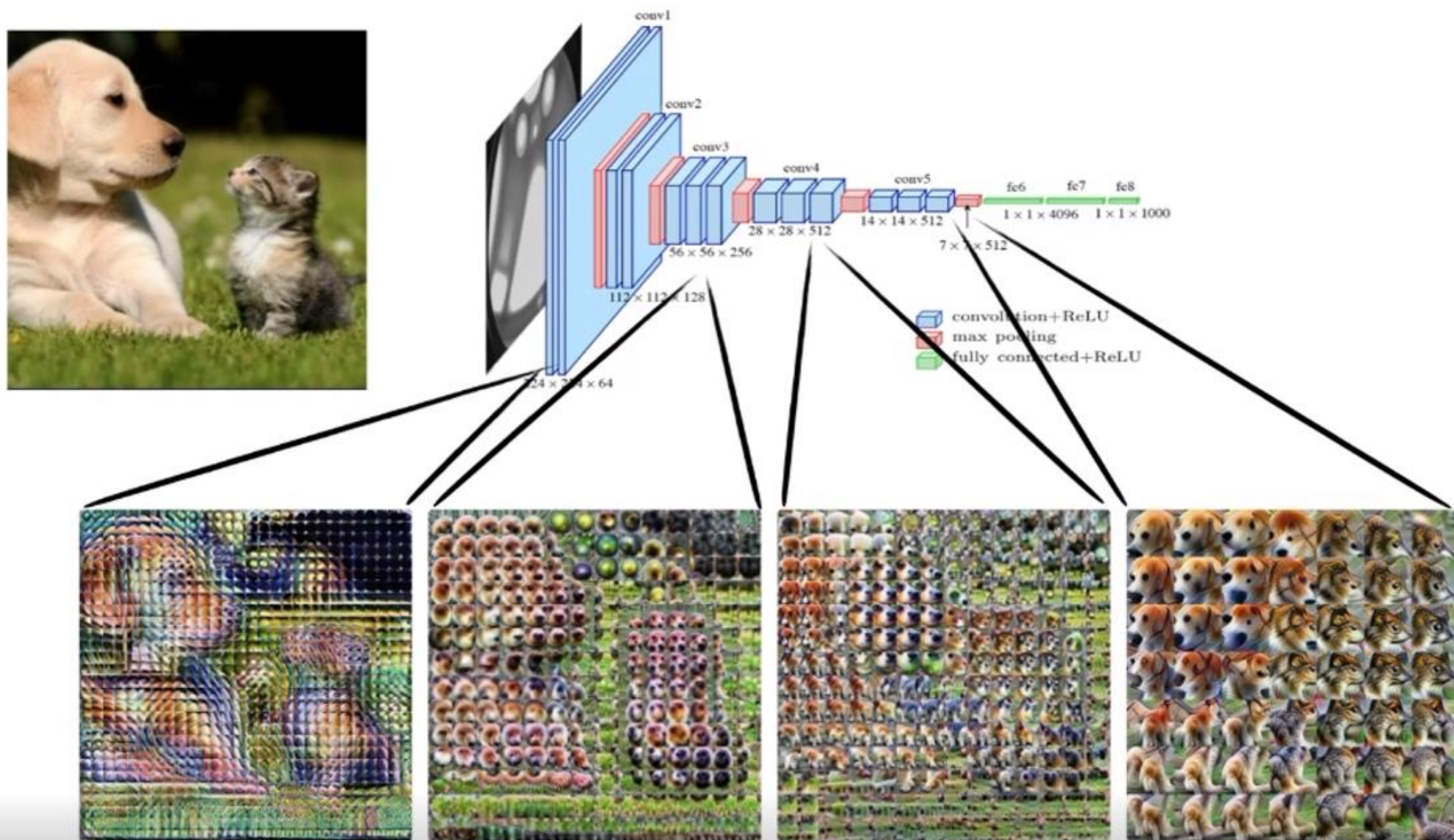
Глаза, носы, уши

Высокоуровневые признаки



Строение лица

# Примеры признаков



# Ключевые вопросы

---

## Какие фильтры использовать?

- фильтры со случайными значениями на основе нормального или какого-либо другого распределения. При достаточном обучении и объёме данных нейросеть сама создаёт подходящие фильтры для извлечения наиболее значимых признаков

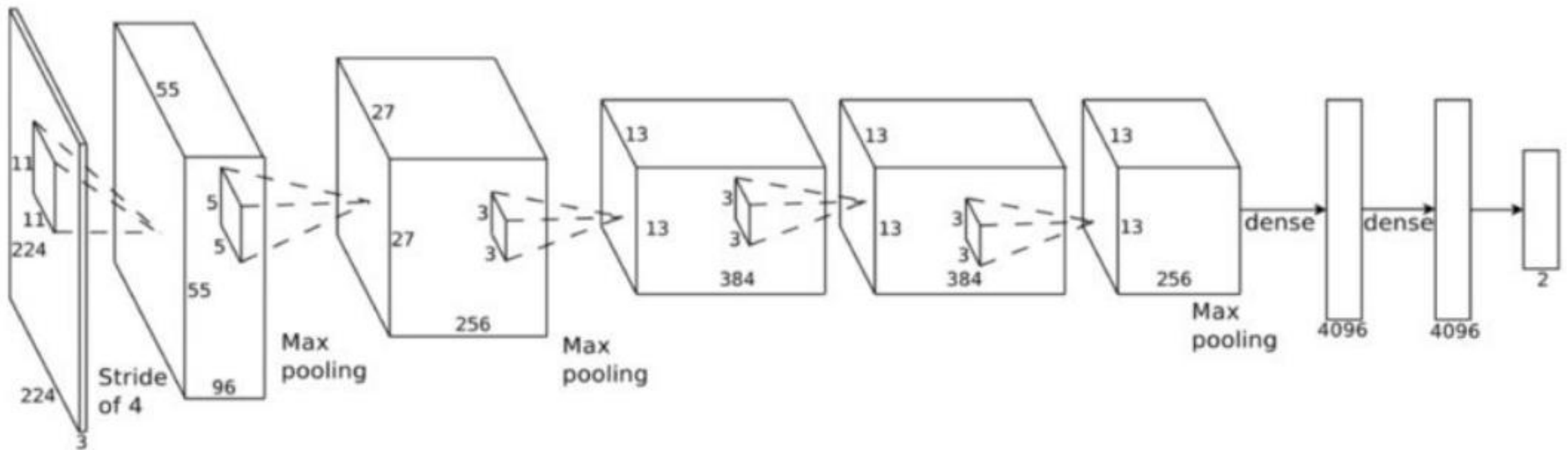
## Сколько фильтров в каждой свертке?

- универсальное правило — использовать фильтры с нечётными размерами (3×3, 5×5, 7×7). Также крупным фильтрам обычно предпочитают маленькие, но возможны и компромиссные соотношения, которые надо вычислять эмпирически.

# Известные сверточные НС

---

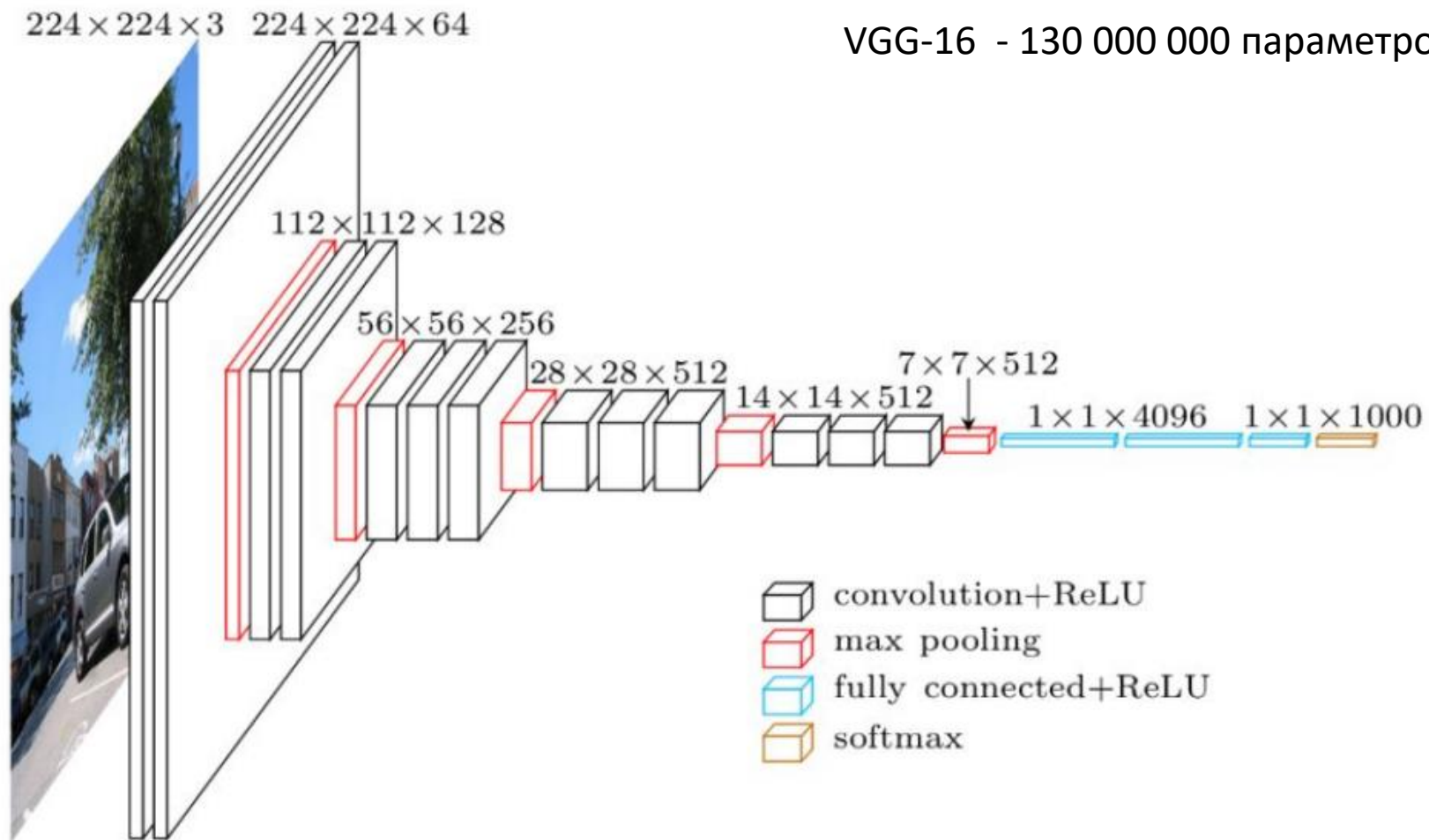
# AlexNet



60 000 000 параметров



# VGG



# ResNet (остаточные сети)

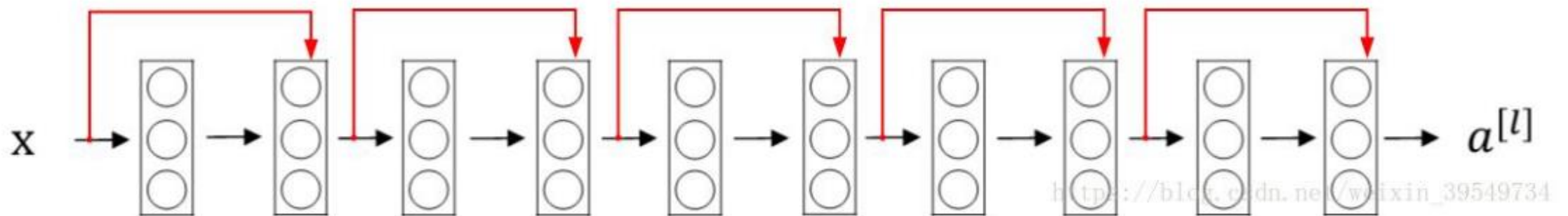
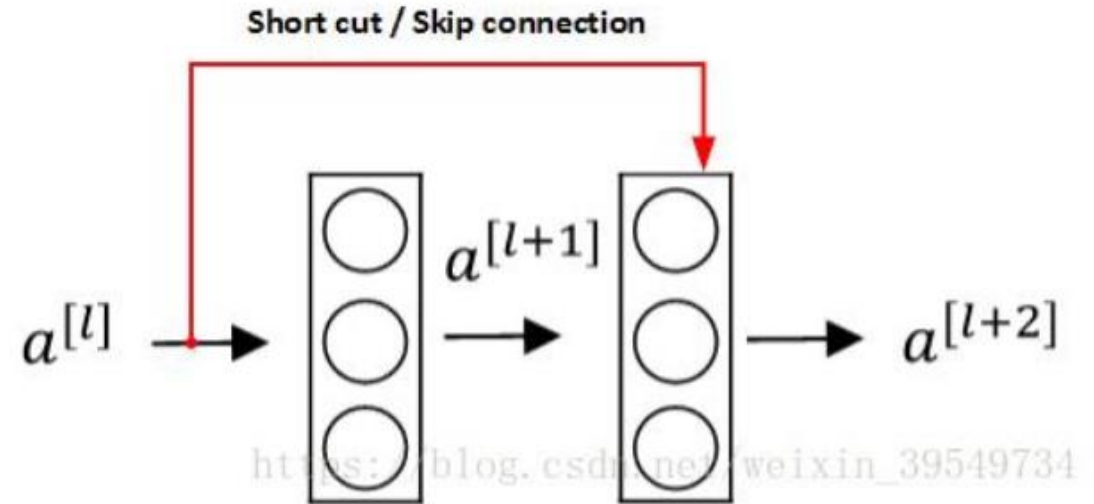
2015 – 2016 года

Распознавание изображений

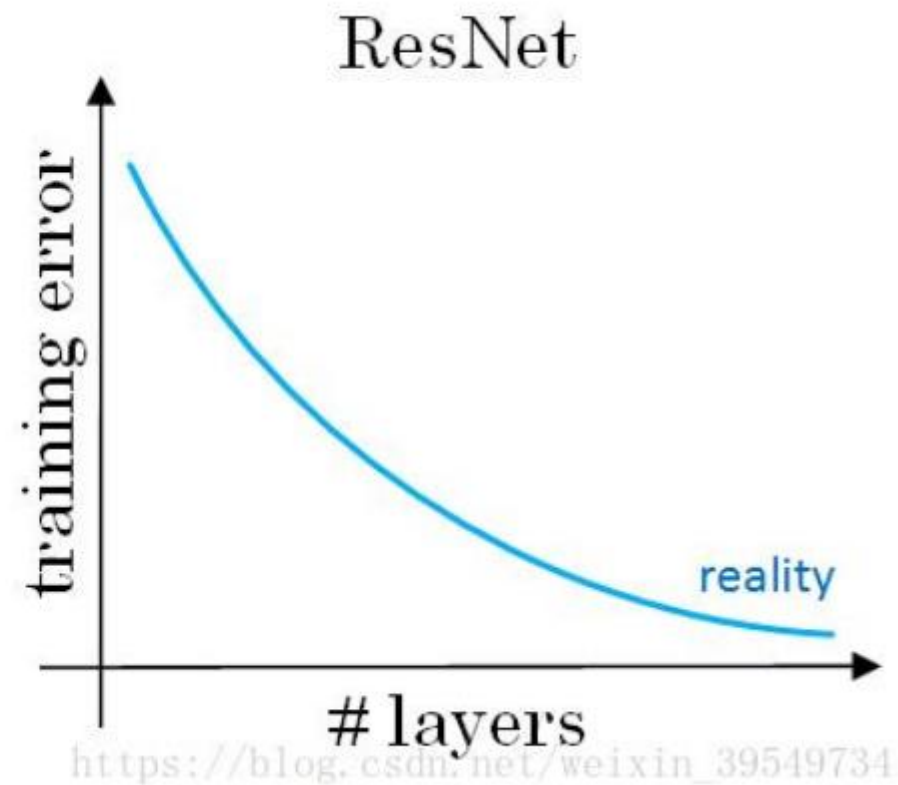
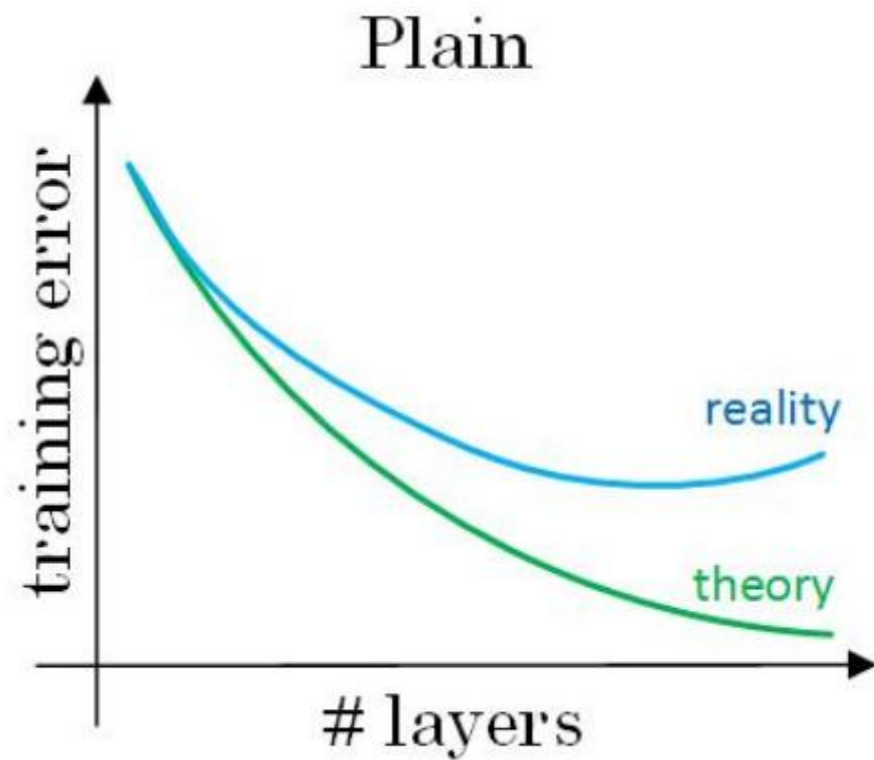
Исчезающий градиент

Добавление входных данных к выходным, чтобы градиенты не исчезали так быстро.

Широкое использование пакетной нормализации

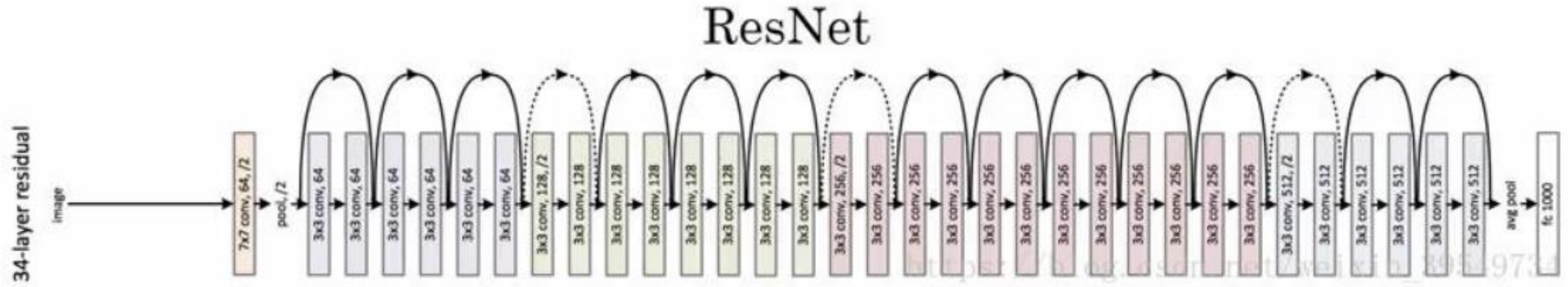


# ResNet



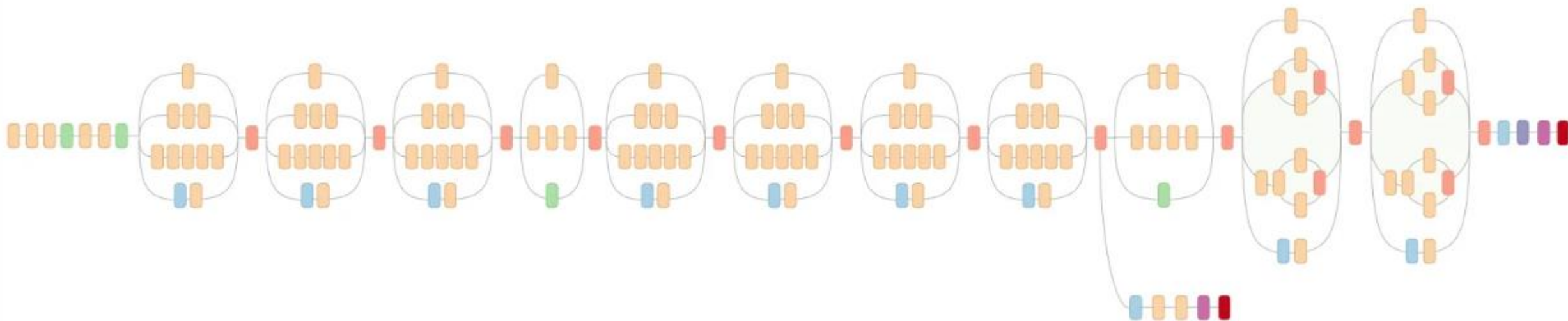
Более 100 слоев

# ResNet



# Пример сверточной НС: Inception V3 для ImageNet

Ошибка: 5,6% (одна модель), 3.6% (ансамбль), 25000000 параметров, учится 1 неделю на GPU

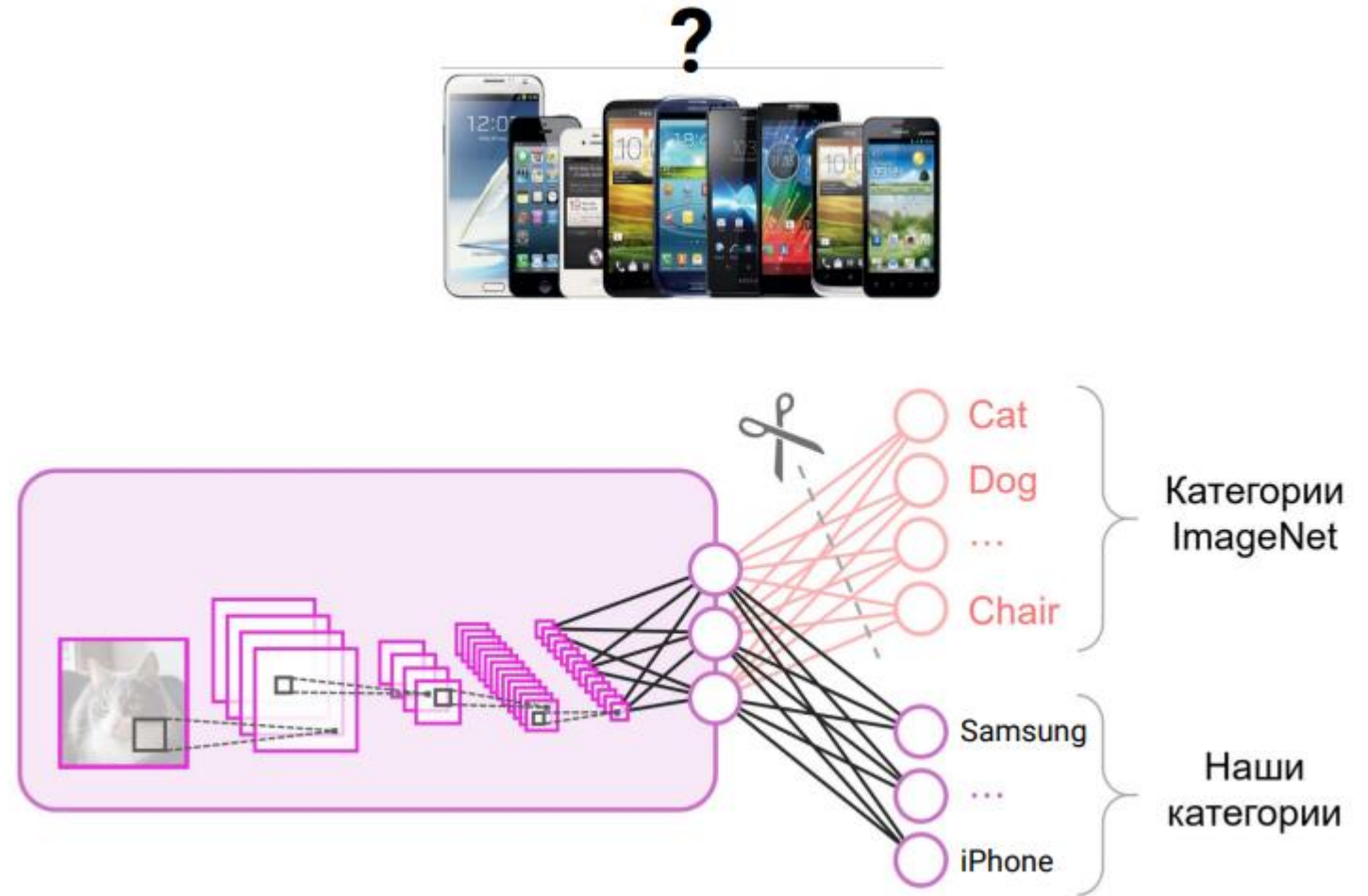


- Convolution
- AvgPool
- MaxPool
- Concat
- Dropout
- Fully connected
- Softmax

GoogLeNet

# Трансферное обучение (Transfer Learning)

мощный метод обучения глубоких нейронных сетей, который позволяет использовать знания, полученные об одной проблеме глубокого обучения, и применять их к другой, но со схожей задачей.





# CNN в Keras

---

**keras.layers.Conv2D(filters, kernel\_size, strides=(1, 1), ...)**

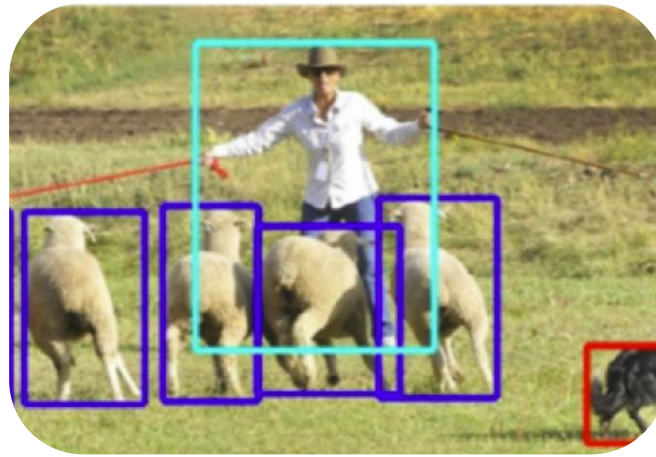
```
model = keras.Sequential([
    Conv2D(32, (3,3), padding='same', activation='relu', input_shape=(28, 28, 1)),
    MaxPooling2D((2, 2), strides=2),
    Conv2D(64, (3,3), padding='same', activation='relu'),
    MaxPooling2D((2, 2), strides=2),
    Flatten(),
    Dense(128, activation='relu'),
    Dense(10, activation='softmax')
])
```

# Задачи, решаемые сверточными НС

---



Классификация



Детекция



Сегментация

Распознавание рукописного текста	Точность человека — 97.5% CNN — 99.8%		
Компьютерное зрение	CNN распознает не только простые объекты на фото, но и эмоции, действия, а еще анализирует видео для автопилотов (семантическая сегментация).	Фото	Улучшение качества, оцветнение
3D реконструкция	Создание 3D моделей по видео	Медицина	
Развлечение	Стилизация и генерация картинок	Безопасность	Обнаружение аномального поведения (Свертка + Рекуррентность)
		Игры	В итоге сеть играет круче профессионала, выбивая дырку и специально загоняя туда шар.

Благодарю за  
ВНИМАНИЕ

---

[HTTPS://CS.STANFORD.EDU/PEOPLE/KARPATHY/CONVNETJS/DEMO/  
CIFAR10.HTML](https://cs.stanford.edu/people/karpathy/convnetjs/demo/cifar10.html)