

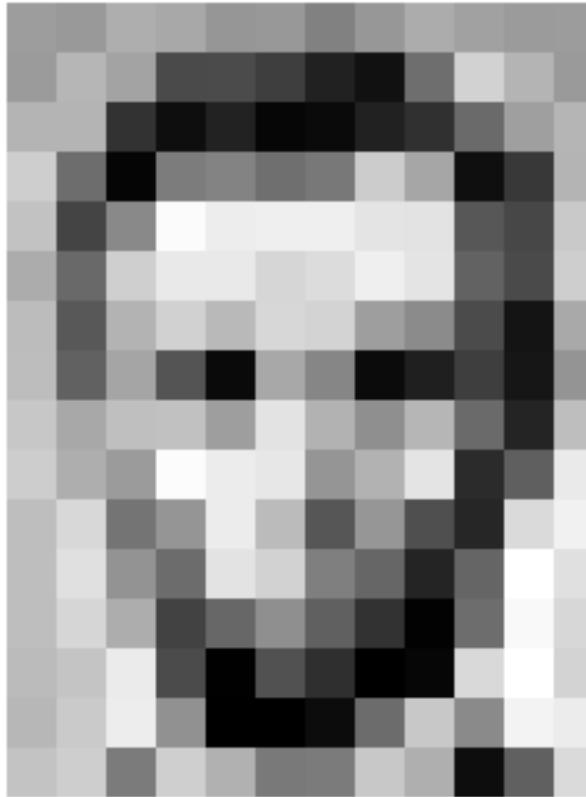
Фаза 2 • Неделя 1 • Среда

Сверточные слои • Convolutional layers

- как работать с изображениями?
- как представить изображение в виде матрицы?
- какие виды слоев нейронных сетей для этого подходят?
 - какие параметры есть у этих слоев?
- как реализовать нейронную сеть-классификатор изображений?

Как представить изображение в виде чисел?

Рассматриваем изображение как матрицу или тензор

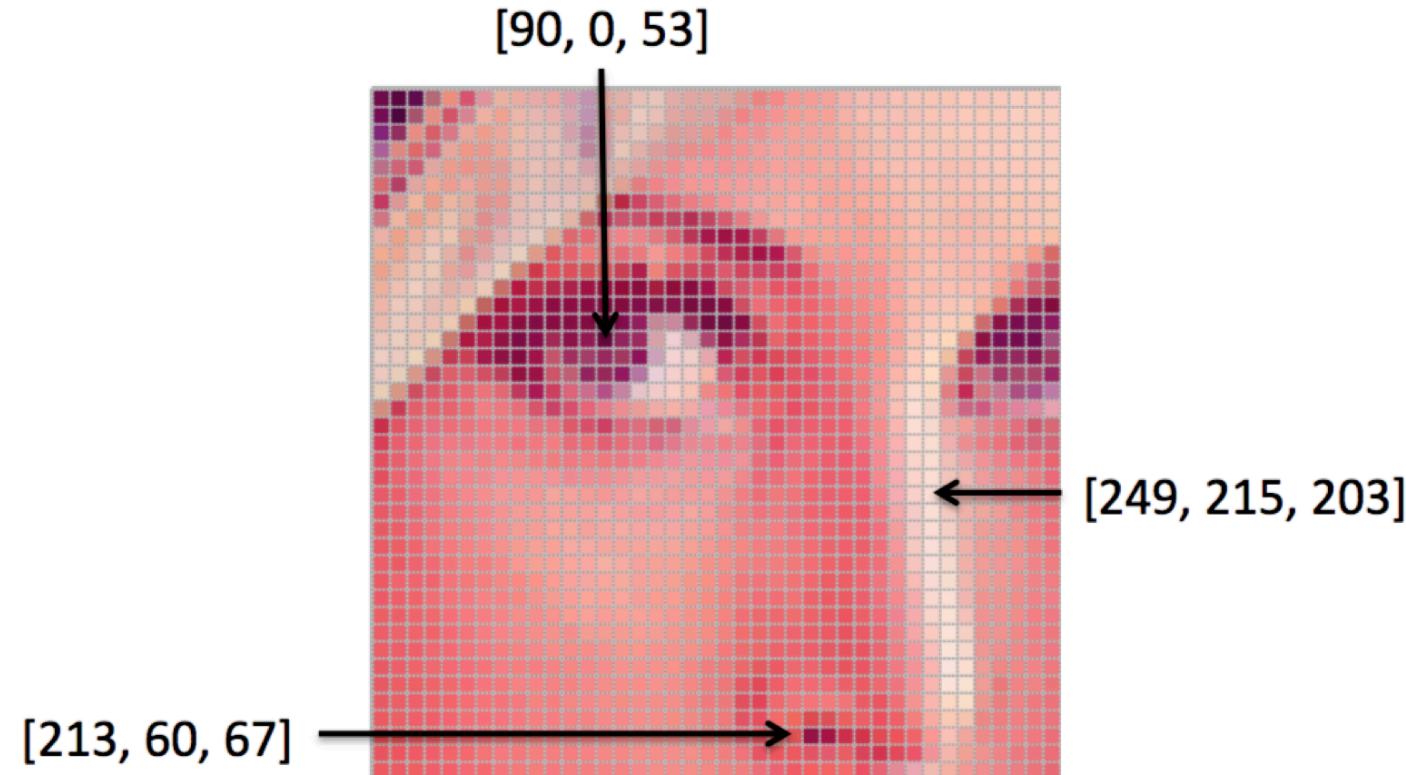


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
196	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	95	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
196	206	123	207	177	121	123	200	175	13	96	218

Как представить изображение в виде чисел?

Рассматриваем изображение как матрицу или тензор



Сверточный слой

3	0	0	1	1
4	2	2	1	1
1	2	5	1	0
1	4	5	5	0
3	4	0	0	0

Исходная матрица

-1	0	1
-1	0	1
-1	0	1

Ядро / фильтр

Сверточный слой

3	0	0	1	1
4	2	2	1	1
1	2	5	1	0
1	4	5	5	0
3	4	0	0	0

Исходная матрица

-1	0	1
-1	0	1
-1	0	1

Ядро / фильтр

`kernel_size = 3` - обязательный параметр: размер фильтра. Обычно они квадратные, но не всегда

Сверточный слой • Операция свертки

$$\begin{array}{|c|c|c|c|c|} \hline
 3 & 0 & 0 & 1 & 1 \\ \hline
 4 & 2 & 2 & 1 & 1 \\ \hline
 1 & 2 & 5 & 1 & 0 \\ \hline
 1 & 4 & 5 & 5 & 0 \\ \hline
 3 & 4 & 0 & 0 & 0 \\ \hline
 \end{array}
 \times
 \begin{array}{|c|c|c|} \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 \end{array}
 =
 \begin{array}{|c|c|c|} \hline
 & & \\ \hline
 \end{array}$$

$$\begin{array}{|c|c|c|c|c|} \hline
 3 & 0 & 0 & 1 & 1 \\ \hline
 4 & 2 & 2 & 1 & 1 \\ \hline
 1 & 2 & 5 & 1 & 0 \\ \hline
 1 & 4 & 5 & 5 & 0 \\ \hline
 3 & 4 & 0 & 0 & 0 \\ \hline
 \end{array}
 \times
 \begin{array}{|c|c|c|} \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 \end{array}
 =
 \begin{array}{|c|c|c|} \hline
 -1 & -1 & -5 \\ \hline
 6 & & \\ \hline
 & & \\ \hline
 & & \\ \hline
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{|c|c|c|c|c|} \hline
 3 & 0 & 0 & 1 & 1 \\ \hline
 4 & 2 & 2 & 1 & 1 \\ \hline
 1 & 2 & 5 & 1 & 0 \\ \hline
 1 & 4 & 5 & 5 & 0 \\ \hline
 3 & 4 & 0 & 0 & 0 \\ \hline
 \end{array} & \times & \begin{array}{|c|c|c|} \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 \end{array} & = & \begin{array}{|c|c|c|} \hline
 -1 & -1 & -5 \\ \hline
 6 & -1 & -11 \\ \hline
 5 & & \\ \hline
 \end{array}
 \end{array}$$

$$\begin{array}{|c|c|c|c|c|} \hline & 3 & 0 & 0 & 1 & 1 \\ \hline \end{array}
 \times
 \begin{array}{|c|c|c|c|c|} \hline 4 & 2 & 2 & 1 & 1 \\ \hline 1 & 2 & 5 & 1 & 0 \\ \hline 1 & 4 & 5 & 5 & 0 \\ \hline 3 & 4 & 0 & 0 & 0 \\ \hline \end{array}
 =
 \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline -1 & 0 & 1 \\ \hline \end{array}
 \quad
 \begin{array}{|c|c|c|} \hline -1 & -1 & -5 \\ \hline 6 & -1 & \\ \hline \end{array}$$

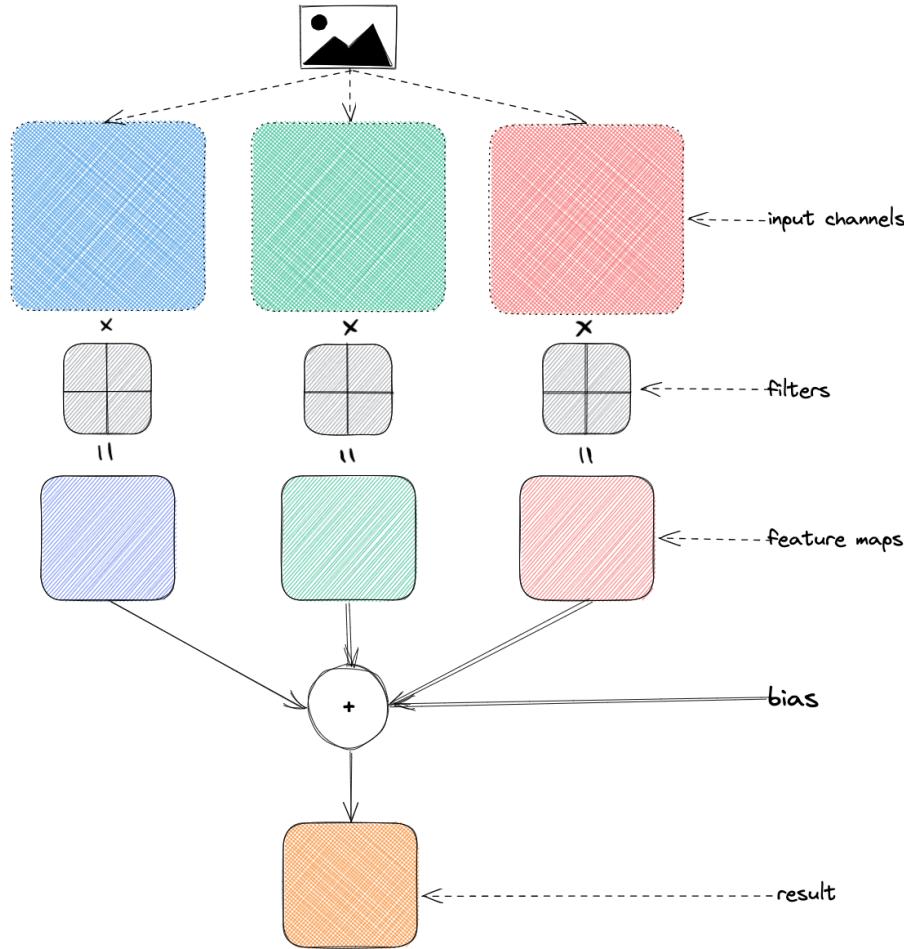
$$\begin{array}{c}
 \begin{array}{ccccc} 3 & 0 & 0 & 1 & 1 \end{array} \\
 \times \quad \begin{array}{ccccc} 4 & 2 & 2 & 1 & 1 \\ 1 & 2 & 5 & 1 & 0 \\ 1 & 4 & 5 & 5 & 0 \\ 3 & 4 & 0 & 0 & 0 \end{array} \\
 = \quad \begin{array}{ccc} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \\ 6 & -1 & -11 \\ 5 & -4 & \end{array}
 \end{array}$$

$$\begin{array}{|c|c|c|c|c|} \hline
 3 & 0 & 0 & 1 & 1 \\ \hline
 4 & 2 & 2 & 1 & 1 \\ \hline
 1 & 2 & 5 & 1 & 0 \\ \hline
 1 & 4 & 5 & 5 & 0 \\ \hline
 \end{array}
 \times
 \begin{array}{|c|c|c|} \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 -1 & 0 & 1 \\ \hline
 \end{array}
 =
 \begin{array}{|c|c|c|} \hline
 & & \\ \hline
 & & \\ \hline
 & & \\ \hline
 & & \\ \hline
 \end{array}$$

$$\begin{array}{c|ccccc}
 3 & 0 & 0 & 1 & 1 \\
 \hline
 4 & 2 & 2 & 1 & 1 \\
 \hline
 1 & 2 & 5 & 1 & 0 \\
 \hline
 1 & 4 & 5 & 5 & 0 \\
 \hline
 3 & 4 & 0 & 0 & 0
 \end{array} \times \begin{array}{c|ccc}
 -1 & 0 & 1 \\
 \hline
 -1 & 0 & 1 \\
 \hline
 -1 & 0 & 1
 \end{array} = \begin{array}{c|ccc}
 -1 & -1 & -5 \\
 \hline
 6 & -1 & -11 \\
 \hline
 & &
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{ccccc} 3 & 0 & 0 & 1 & 1 \end{array} \\
 \times \\
 \begin{array}{ccccc} 4 & 2 & 2 & 1 & 1 \\ 1 & 2 & 5 & 1 & 0 \\ 1 & 4 & 5 & 5 & 0 \\ 3 & 4 & 0 & 0 & 0 \end{array}
 \end{array}
 =
 \begin{array}{ccc} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{array}
 =
 \begin{array}{ccc} -1 & -1 & -5 \\ 6 & -1 & -11 \\ 5 & -4 & 10 \end{array}$$

Сверточный слой • Операция свертки

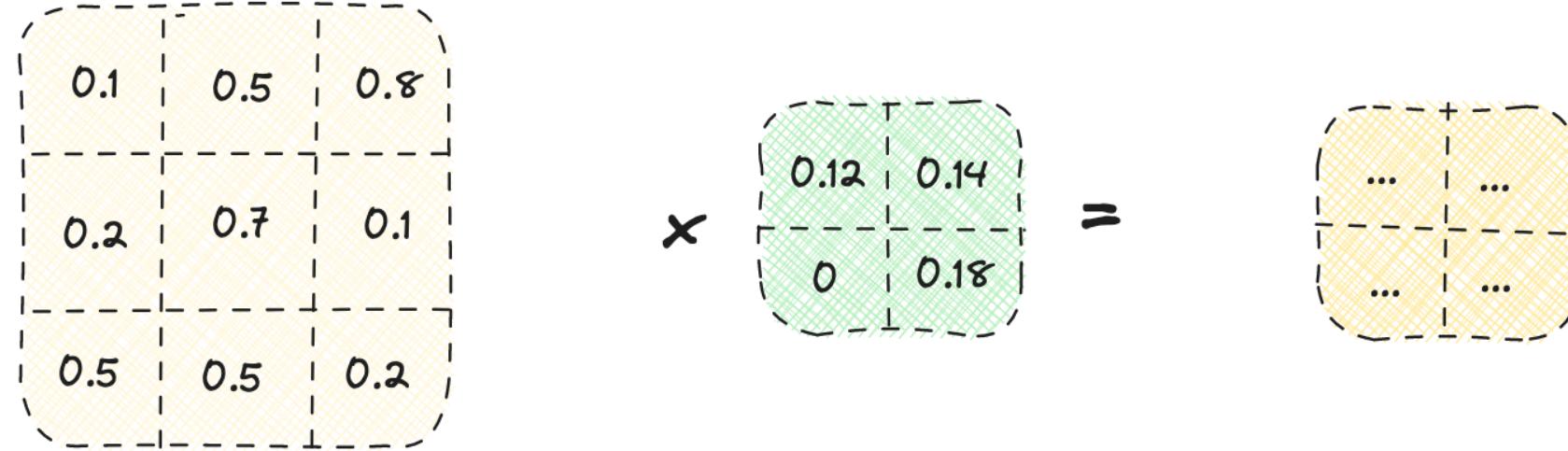


```

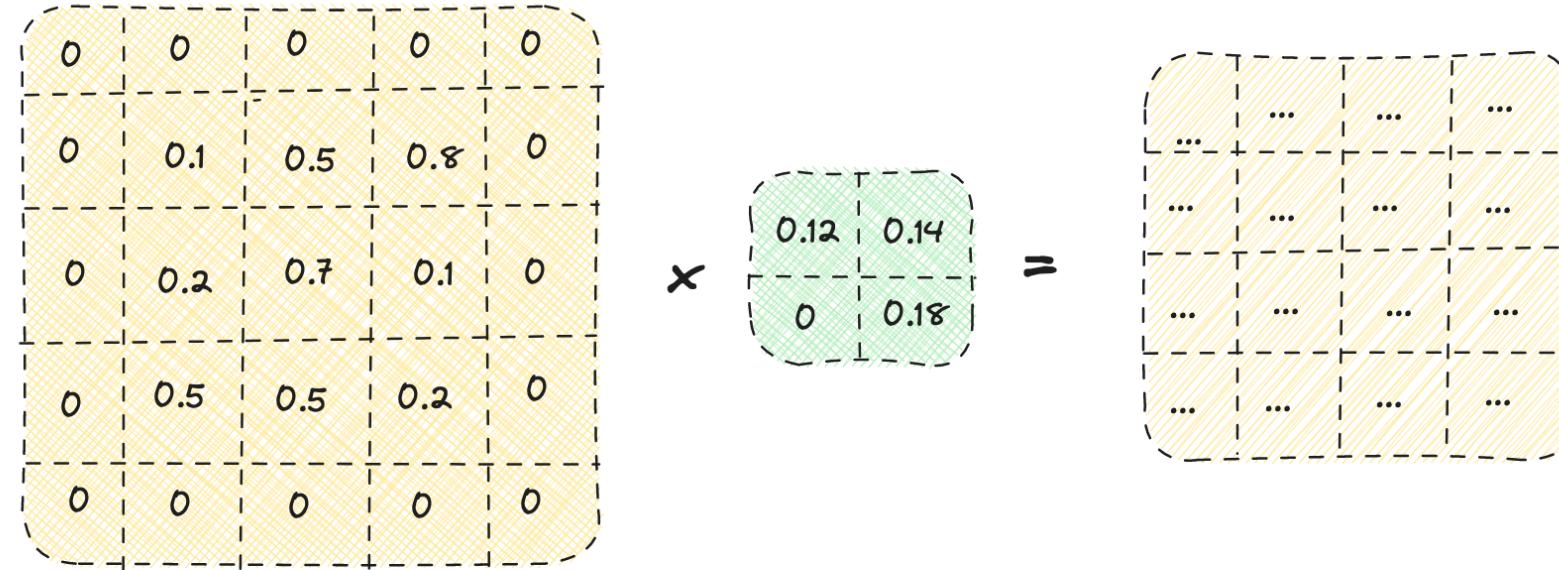
● ● ●
model = nn.Conv2d(
    in_channels=3,
    out_channels=1,
    kernel_size=2
)
model.weight, model.bias

Parameter containing:
tensor([[[[ 0.2278,  0.2484],
          [ 0.1522, -0.2209]],
         [[-0.0611,  0.2156],
          [-0.1626, -0.0988]],
         [[-0.2752,  0.1702],
          [ 0.1131,  0.0462]]]], requires_grad=True),
Parameter containing:
tensor([-0.0321], requires_grad=True)
    
```

Сверточный слой • padding

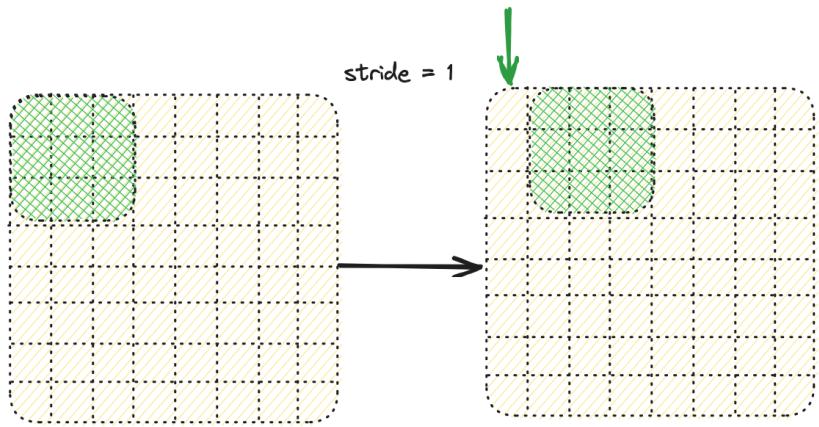


Сверточный слой • padding



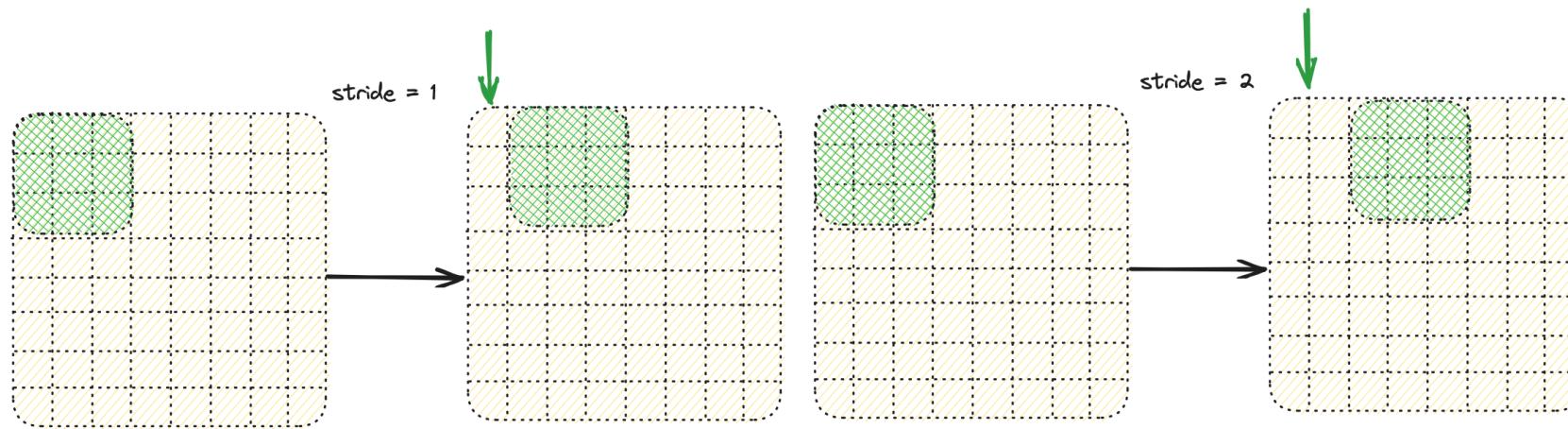
- `padding=1` - по умолчанию 0, можно задавать произвольным, в том числе не нулевым
- Нужен для контроля выходной размерности и предоставления возможности всем участкам фильтра обработать края изображения

Сверточный слой: stride



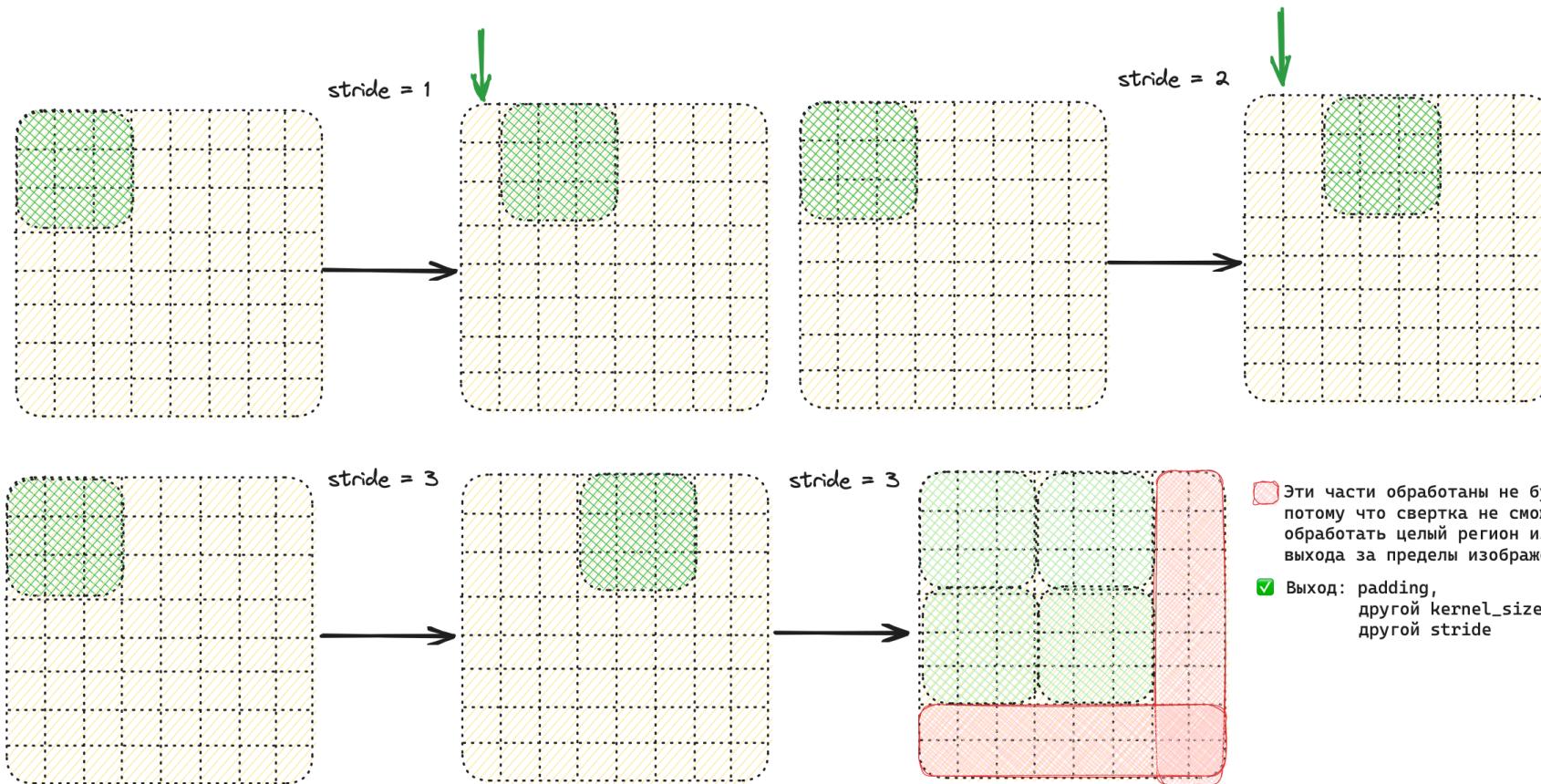
stride = 1 - по умолчанию 1

Сверточный слой: stride



stride = 1 - по умолчанию 1

Сверточный слой: stride



stride = 1 - по умолчанию 1

Выходной размер свертки: формула

W - размер картинки (как правило, подразумевают квадрат)

K - размер ядра

P - padding

S - stride

Выходной размер свертки: формула

W - размер картинки (как правило, подразумевают квадрат)

K - размер ядра

P - padding

S - stride

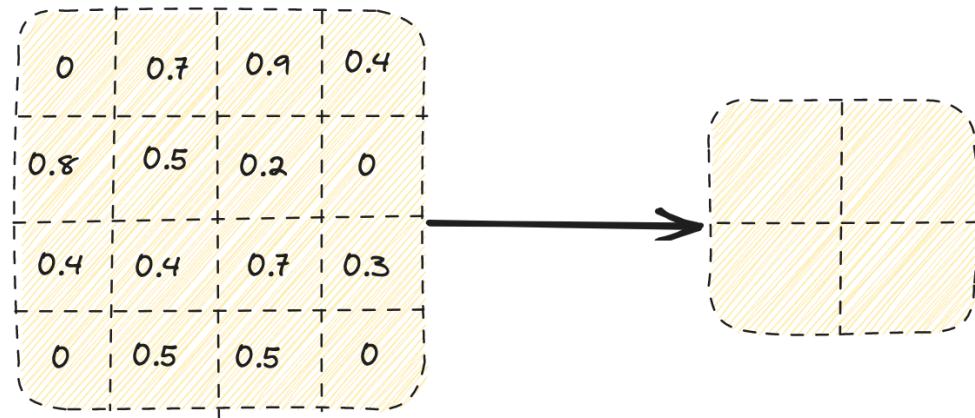
$$\text{out}_{wh} = \frac{(W - K + 2P)}{S} + 1$$

Дополнительно:

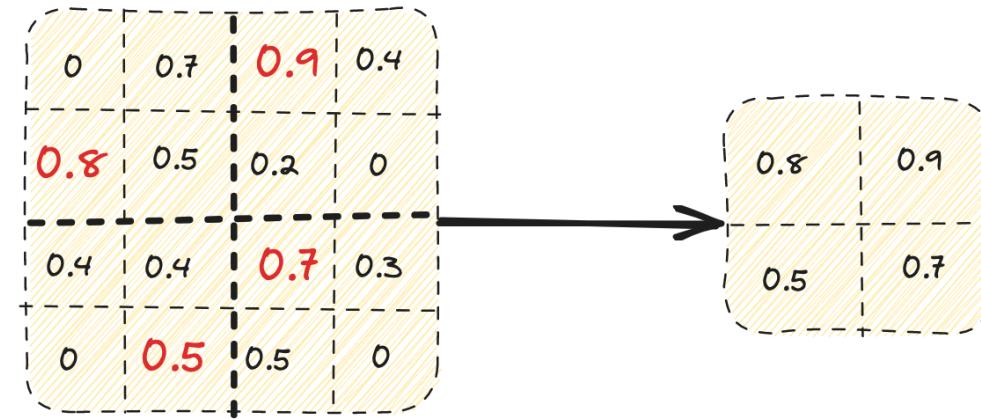
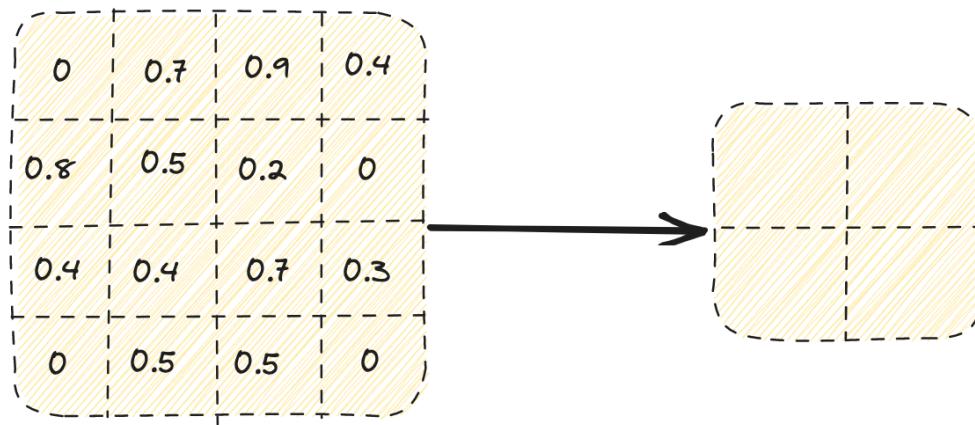
 A guide to convolution arithmetic for deep learning - подробное руководство

 Транспонированная и обратная свертка - сверточные операции на русском

MaxPooling • AveragePooling

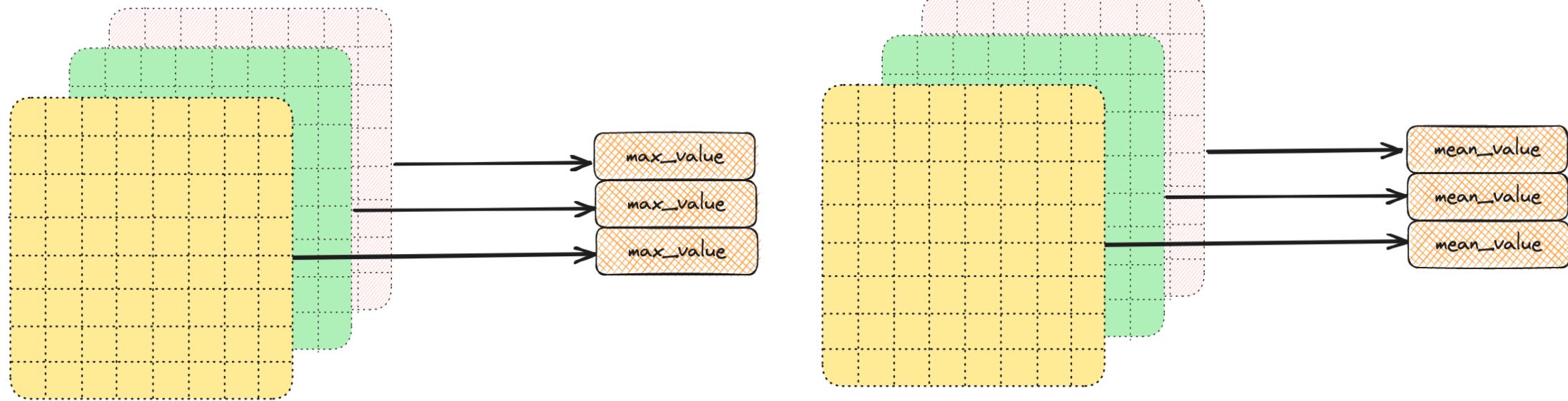


MaxPooling • AveragePooling



- 🔥 `nn.MaxPool2d(kernel_size=2, stride=2, padding=0)`
- Пулинг нужен для понижения размерности

GlobalMaxPooling • GlobalAveragePooling



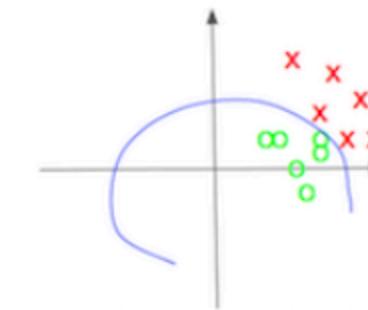
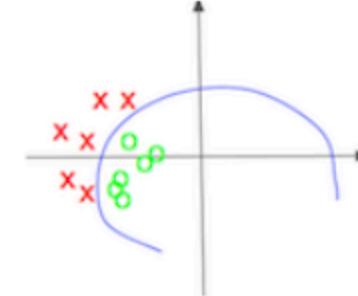
🎥 Convolutional Neural Network Visualization by Otavio Good

📊 А если изображение цветное? (ru)

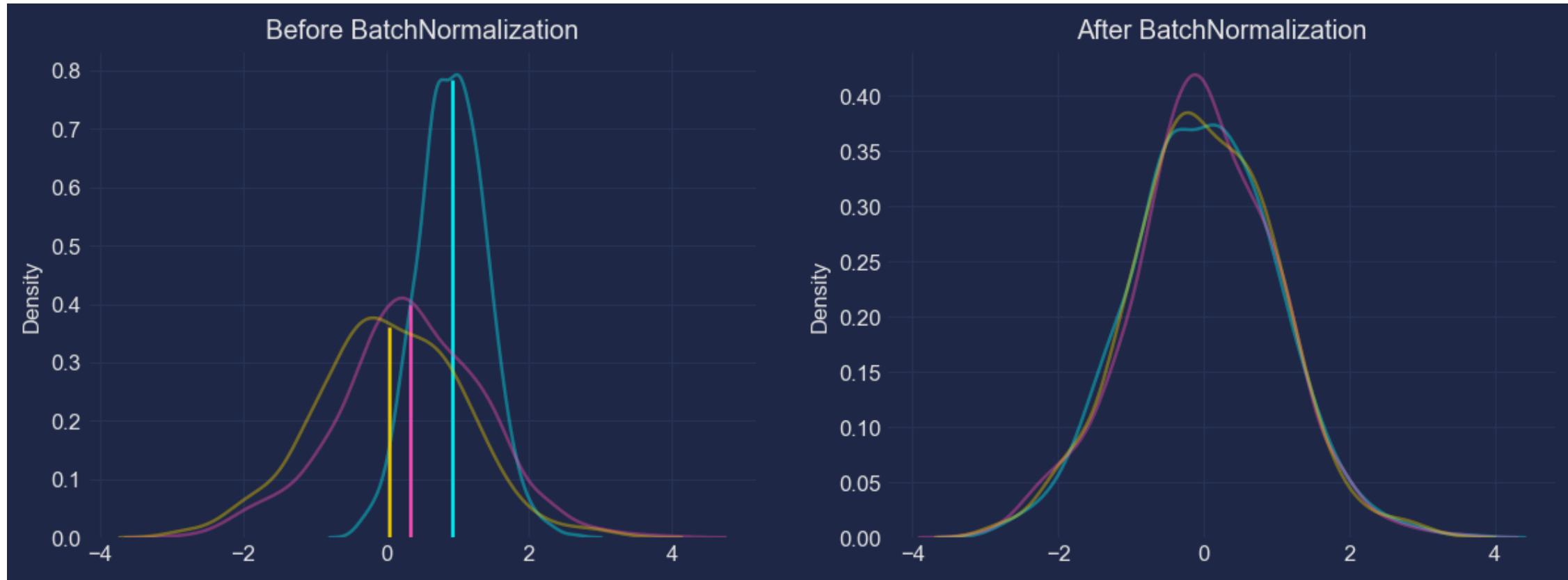
📊 Conv2d: Finally Understand What Happens in the Forward Pass (en)

!! Возможно, лучшие анимации: <https://animatedai.github.io/>

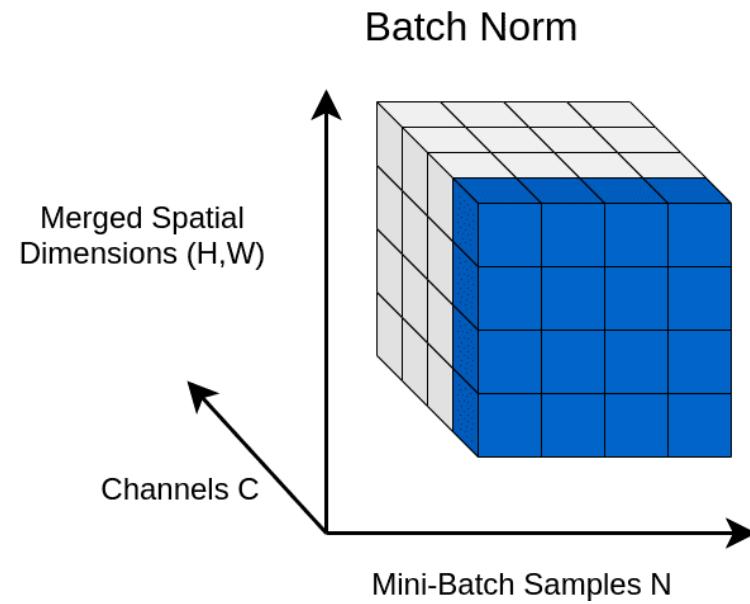
Ковариантный сдвиг



Batch Normalization



Batch Normalization



? до или после активации использовать нормализацию? 🌐 непонятно

Типичная последовательность: Conv2d → BatchNormalization → Activation ...



`nn.BatchNorm2d(num_features)`



In-layer normalization techniques for training very deep neural networks !! Batch-normalization

Аугментации изображений

- Аугментация - процесс (часто случайных) преобразований исходной картинки для повышения устойчивости сети к небольшой изменчивости признаков.
- Сеть должна «понимать», что признаки объектов могут быть повернуты, перемещены, затемнены и т.д.

Аугментации изображений

- Аугментация - процесс (часто случайных) преобразований исходной картинки для повышения устойчивости сети к небольшой изменчивости признаков.
- Сеть должна «понимать», что признаки объектов могут быть повернуты, перемещены, затемнены и т.д.

Зачем:

- «увеличение» датасета
- повышение устойчивости модели к искаженным данным
- есть встроенные в `keras` , `pyTorch` , но лучше всего - `Albumentations` 

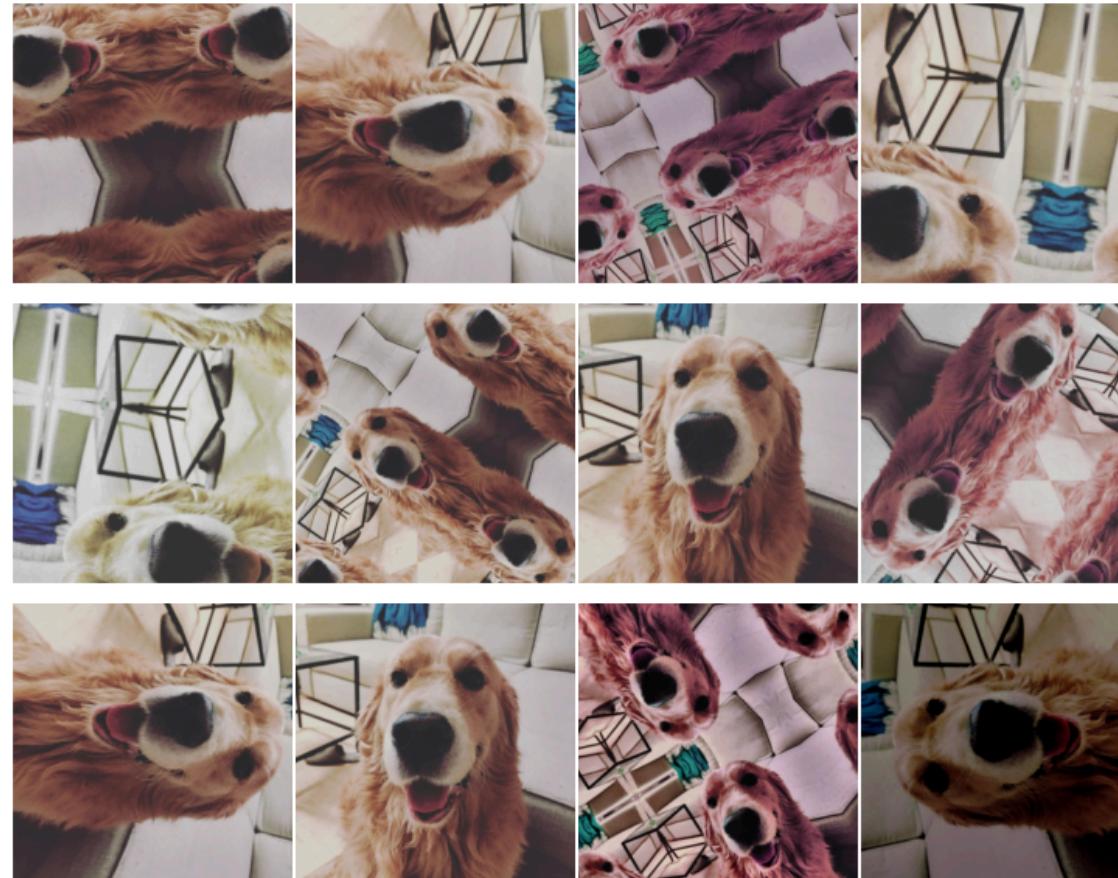
Аугментации изображений

! Аугментируем **только** обучающую выборку – валидационная выборка должна остьаться без изменений

Это обычный Альф

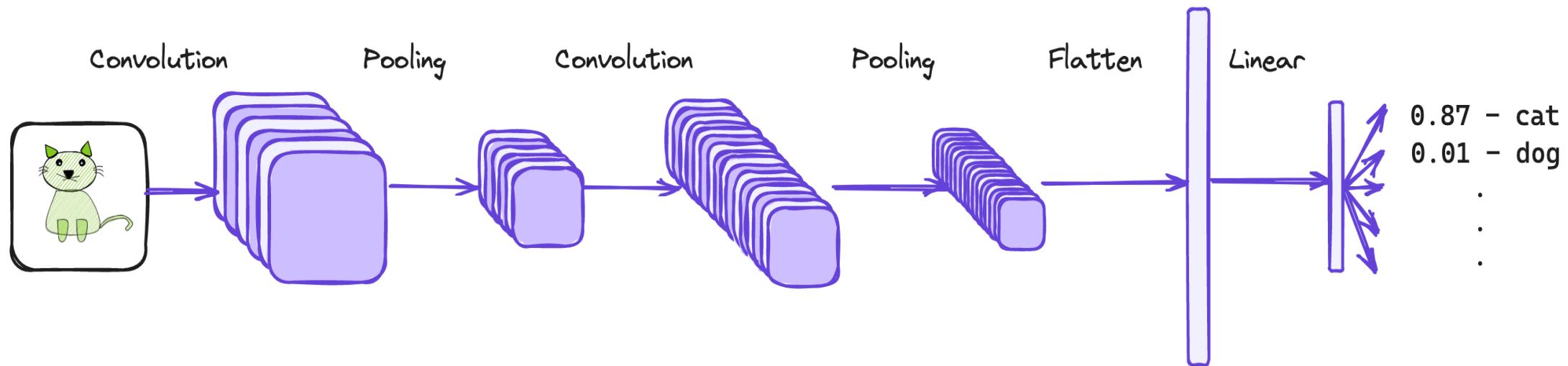


Это аугментированный Альф

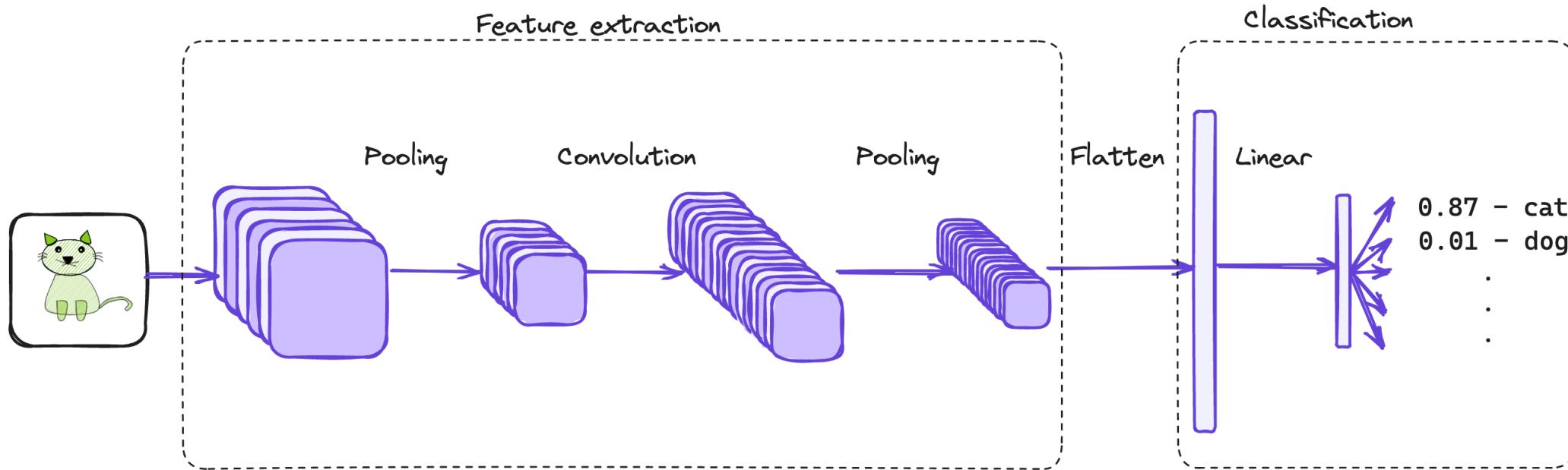


🔥 `import torchvision.transforms as T`

Базовая архитектура классификатора изображений



Базовая архитектура классификатора изображений



Итоги

- главные параметры сверточного слоя `nn.Conv2d`:
 - `in_channels` - число входящих матриц (для цветного изображения 3, для ч/б - 1)
 - `out_channels` - сколько карт признаков (feature maps) сгенерировать на выходе
 - `kernel_size` - размер фильтра
 - `padding` - «рамка» из нулей
 - `stride` - шаг перемещения фильтра
- `Max/AveragePooling` – сокращение размерности
- чтобы обучалось быстрее, нужно использовать **батч-нормализацию**
- сверточные сети могут быть применены при обработке временных рядов, текста, трехмерных объектов и пр.