Основы практического использования нейронных сетей.

Лекция 3. Методы повышения эффективности алгоритмов обучения для глубоких НС.

Дмитрий Буряк.
к.ф.-м.н.
dyb04@yandex.ru

Batch normalization

- Нормализация + декорреляция входных данных → повышение эффективности обучения.
- Нарушение полученных свойств входных векторов для промежуточных данных во внутренних слоях (internal covariance shift).
- Идея: проводить предобработку входных данных для каждого внутреннего слоя.
- Оптимизация вычислительных затрат → нормализация внутренних данных (без декорреляции).

Sergey Ioffe, Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift, 2015

Batch normalization. Алгоритм

Input: Network N with trainable parameters Θ;
subset of activations $\{x^{(k)}\}_{k=1}^{K}$
Output: Batch-normalized network for inference, $N_{\text{BN}}^{\text{inf}}$

1: $N_{\text{BN}}^{\text{tr}} \leftarrow N$ // Training BN network
2: **for** $k = 1 \dots K$ **do**
3: Add transformation $y^{(k)} = \text{BN}_{\gamma^{(k)},\beta^{(k)}}(x^{(k)})$ to $N_{\text{BN}}^{\text{tr}}$ (Alg. 1)
4: Modify each layer in $N_{\text{BN}}^{\text{tr}}$ with input $x^{(k)}$ to take $y^{(k)}$ instead
5: **end for**
6: Train $N_{\text{BN}}^{\text{tr}}$ to optimize the parameters $\Theta \cup \{\gamma^{(k)}, \beta^{(k)}\}_{k=1}^{K}$
7: $N_{\text{BN}}^{\text{inf}} \leftarrow N_{\text{BN}}^{\text{tr}}$ // Inference BN network with frozen // parameters
8: **for** $k = 1 \dots K$ **do**
9: // For clarity, $x \equiv x^{(k)}, \gamma \equiv \gamma^{(k)}, \mu_{\mathcal{B}} \equiv \mu_{\mathcal{B}}^{(k)}$, etc.
10: Process multiple training mini-batches $\mathcal{B}$, each of size m, and average over them:

$$\text{E}[x] \leftarrow \text{E}_{\mathcal{B}}[\mu_{\mathcal{B}}]$$
$$\text{Var}[x] \leftarrow \frac{m}{m-1}\text{E}_{\mathcal{B}}[\sigma_{\mathcal{B}}^2]$$

11: In $N_{\text{BN}}^{\text{inf}}$, replace the transform $y = \text{BN}_{\gamma,\beta}(x)$ with $y = \frac{\gamma}{\sqrt{\text{Var}[x]+\epsilon}} \cdot x + \left(\beta - \frac{\gamma\,\text{E}[x]}{\sqrt{\text{Var}[x]+\epsilon}}\right)$
12: **end for**

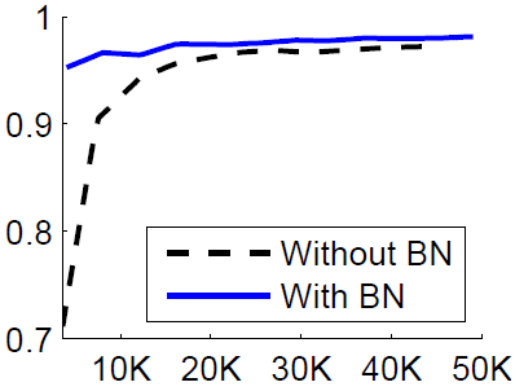
Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\dots m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \text{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$
$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$
$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \text{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

Batch normalization. Примеры.

- Классификация MNIST
- Сеть 784x100x100x100x10,

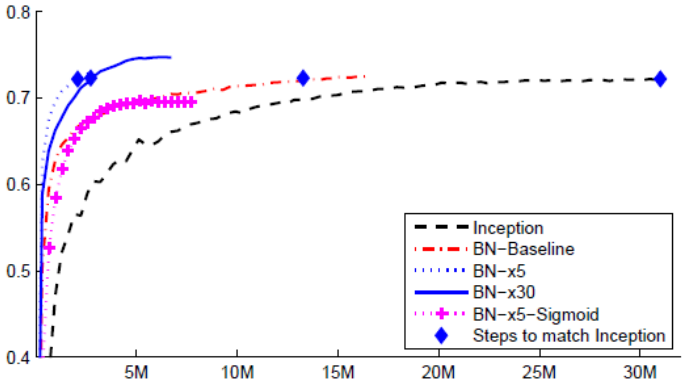
50000 обучающих примеров



- Классификация ImageNet
- Сеть $13.6*10^6$ параметров,

1000 классов

Model	Steps to 72.2%	Max accuracy
Inception	$31.0 \cdot 10^6$	72.2%
BN-Baseline	$13.3 \cdot 10^6$	72.7%
BN-x5	$2.1 \cdot 10^6$	73.0%
BN-x30	$2.7 \cdot 10^6$	74.8%
BN-x5-Sigmoid		69.8%



Регуляризация L2

- Регуляризация - метод предотвращения переобучения НС.
- Введение штрафа для больших весов.

$$C = C_0 + \frac{\lambda}{2n} \sum_w w^2$$

- λ - коэффициент регуляризации.

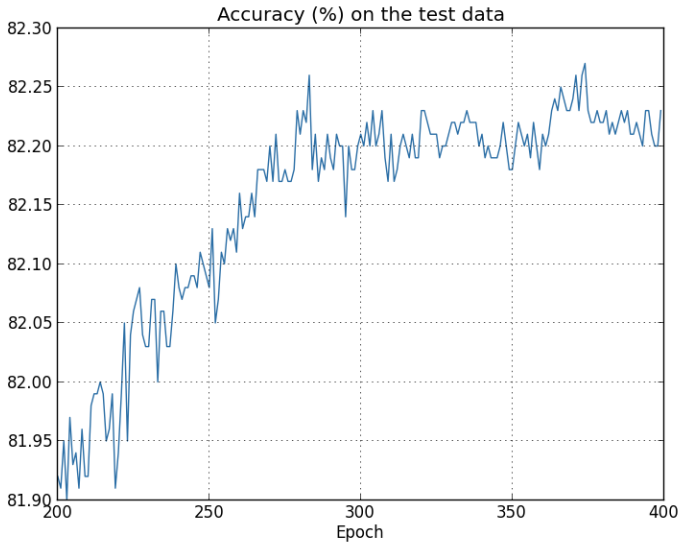
$$\frac{\partial C}{\partial w} = \frac{\partial C_0}{\partial w} + \frac{\lambda}{n} w$$
$$\frac{\partial C}{\partial b} = \frac{\partial C_0}{\partial b}.$$

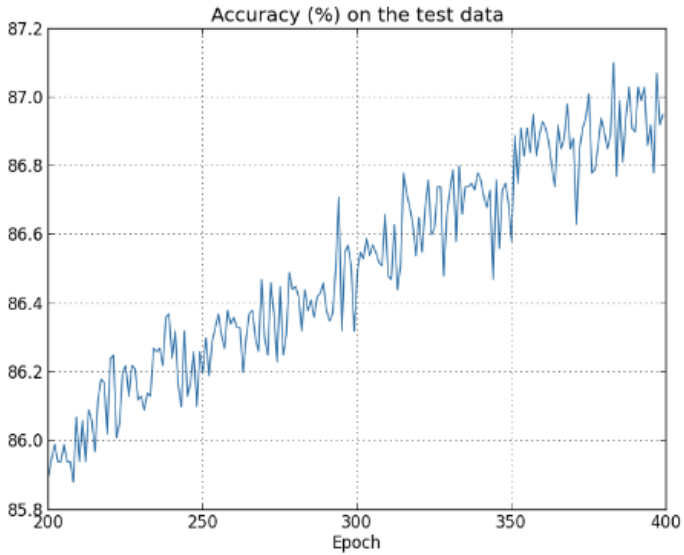
$$b \to b - \eta \frac{\partial C_0}{\partial b}$$
$$w \to w - \eta \frac{\partial C_0}{\partial w} - \frac{\eta\lambda}{n} w$$
$$= \left(1 - \frac{\eta\lambda}{n}\right) w - \eta \frac{\partial C_0}{\partial w}$$

- Масштабирование веса перед коррекцией по градиентному спуску.

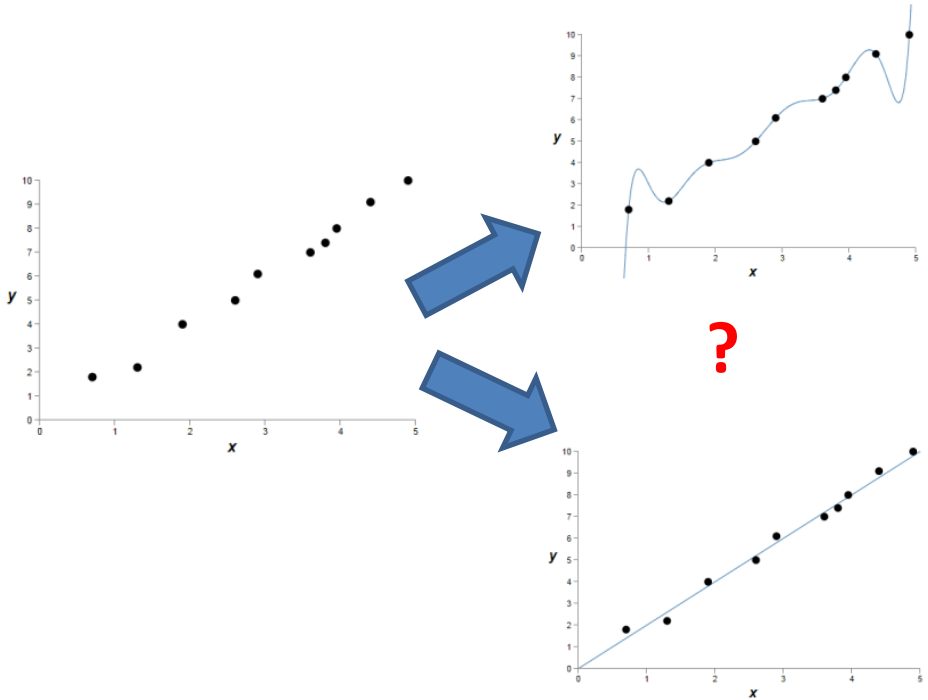
Пример применения регуляризации L2

- Классификация MNIST
- Сеть 784x30x10, 1000 обучающих примеров





Регуляризация → снижение переобучения



- Нет однозначного решения без дополнительной инфоормации.
- Большие значения параметров → увеличение чувствительности к шуму.

$$y = a_0 x^9 + a_1 x^8 + \dots$$

$$y = a_0 x + a_1$$

Регуляризация L1

- Введение штрафа для больших весов.

$$C = C_0 + \frac{\lambda}{n} \sum_w |w|$$

- λ - коэффициент регуляризации.

$$\frac{\partial C}{\partial w} = \frac{\partial C_0}{\partial w} + \frac{\lambda}{n} \operatorname{sgn}(w) \qquad w \to w' = w - \frac{\eta\lambda}{n} \operatorname{sgn}(w) - \eta \frac{\partial C_0}{\partial w}$$

- Уменьшение веса на фиксированную величину

- Для регуляризации L2 значение уменьшения веса зависит от его величины.

$$w \to w' = w \left(1 - \frac{\eta\lambda}{n}\right) - \eta \frac{\partial C_0}{\partial w}$$

Регуляризация Max-norm

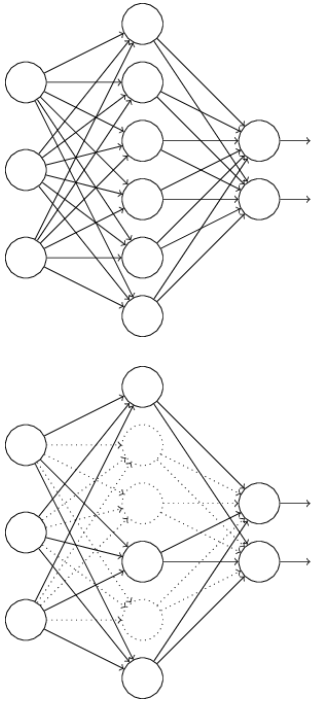
- Ограничения нормы вектора весов для каждого нейрона.

$$||\mathbf{w}||_2 \leq c$$

- c - гиперпараметр.
- Реализация через нормировку вектора весов при невыполнении неравенства.
- Эффективна при совместном использовании с Dropout

Dropout

- Инструмент регуляризации.
- Модификация архитектуры сети в процессе обучения.
- Упрощенная схема Dropout
    1. Временно удалить из НС половину случайно выбранных внутренних нейронов с соответствующими связями.
    2. Провести итерацию обучение на пакете: обновление связей оставшихся нейронов.
    3. Восстановить удаленные нейроны и их связи.
    4. Повторить п. 1 – 3.
- Перед применением сети уменьшить внутренние веса в 2 раза.



N. Srivastava et al. Dropout: A simple way to prevent neural networks from overfitting, 2014.

Dropout. Пример использования.

- MNIST.

Входной вектор 784 элемента.

10 классов.

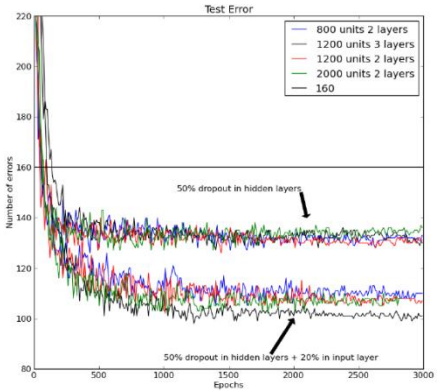
10000 тестовых изображений.

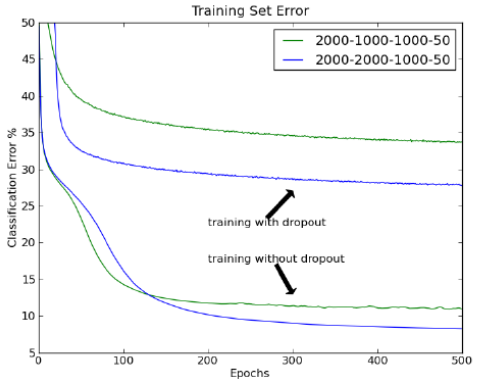
- Reuters. Классификация документов.

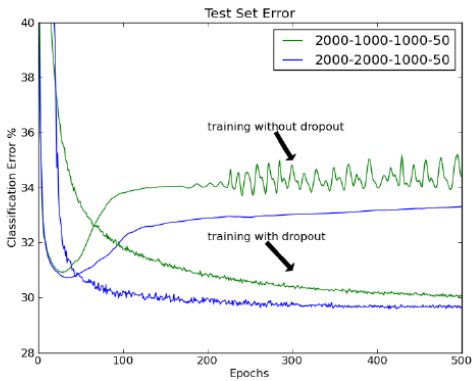
Входной вектор 2000 элементов.

50 классов.

~200000 тестовых документов.



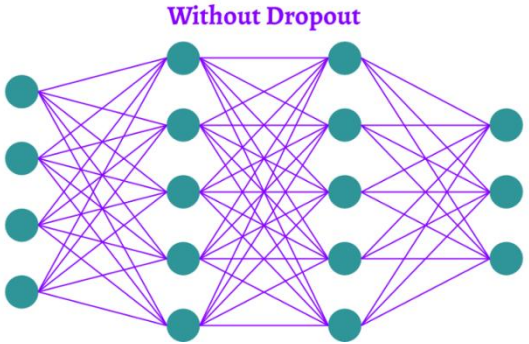


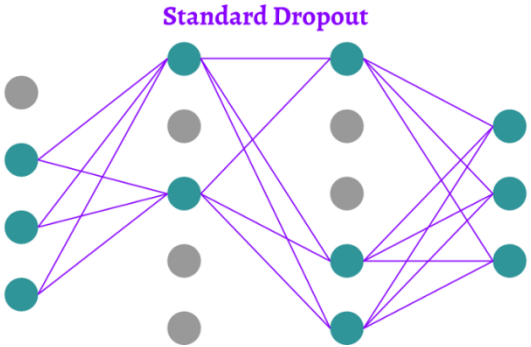


G.E. Hinton et al, Improving neural networks by preventing co-adaptation of feature detectors, 2012

Варианты Dropout. DropConnect.

- Удаление связей.



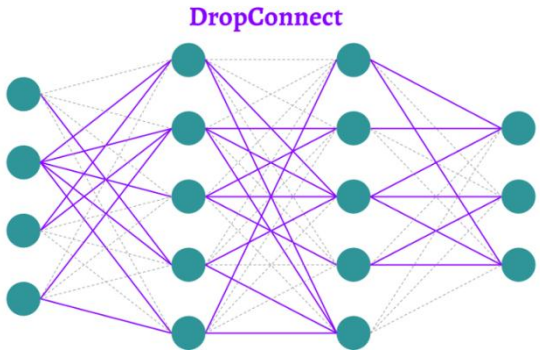


Training Phase :

$$\mathbf{y} = f(\mathbf{W}\mathbf{x}) \circ \mathbf{m}, \quad m_i \sim Bernoulli(p)$$

Testing Phase :

$$\mathbf{y} = (1-p)f(\mathbf{W}\mathbf{x})$$



Training Phase :

$$\mathbf{y} = f((\mathbf{W} \circ \mathbf{M})\mathbf{x}), \quad M_{i,j} \sim Bernoulli(p)$$

Testing Phase :

$$\mathbf{y} = (\mathbf{W}\mathbf{x}) \circ \hat{\mathbf{m}}(\mathbf{Z})$$

$$\text{where } \hat{m}_i(Z) = \frac{1}{Z}\sum_{z=0}^{Z} f(\hat{x}_{i,z}), \quad \hat{x}_{i,z} \sim \mathcal{N}(\mu_i, \sigma_i^2)$$

$$\text{and } \boldsymbol{\mu} = p\mathbf{W}\mathbf{x}, \quad \boldsymbol{\sigma}^2 = p(1-p)(\mathbf{W} \circ \mathbf{W})(\mathbf{x} \circ \mathbf{x}), \quad Z \in \mathbb{N}^+$$

https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Варианты Dropout. Standout.

- Вероятность удаления нейрона зависит от величины весов.
- Пример.

Training Phase :

$$\mathbf{y} = f(\mathbf{W}\mathbf{x}) \circ \mathbf{m}, \quad m_i \sim Bernoulli(g(\mathbf{W_s}\mathbf{x}))$$

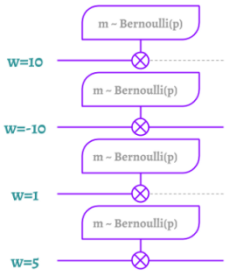
Testing Phase :

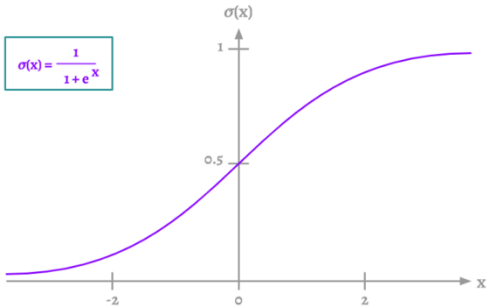
$$\mathbf{y} = (1 - g(\mathbf{W_s}\mathbf{x})) \circ f(\mathbf{W}\mathbf{x})$$

where $\mathbf{W_s}$ is the belief network's weights
and g is the belief network's activation function

$$g(x) = |\sigma(x)|$$

$$\mathbf{W_s} = \alpha\mathbf{W} + \beta$$





https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Варианты Dropout. Gaussian Dropout.

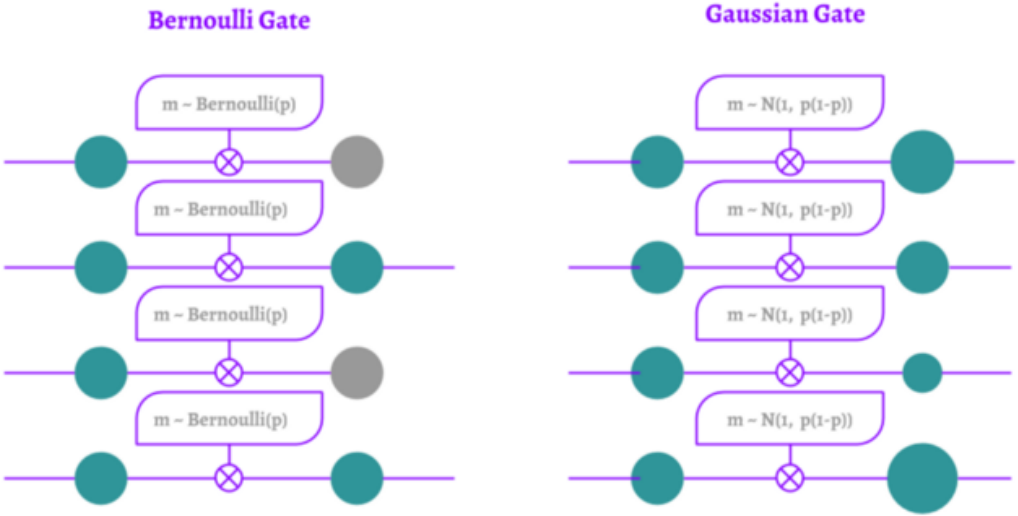
- ❑ Нейроны не удаляются, а «завешиваются» с помощью нормального распределения.
- ❑ Выше скорость сходимости.

Training Phase :

$$\mathbf{y} = f(\mathbf{W}\mathbf{x}) \circ \mathbf{m}, \quad m_i \sim \mathcal{N}(1,\ p(1-p))$$

Testing Phase :

$$\mathbf{y} = f(\mathbf{W}\mathbf{x})$$



https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Варианты Dropout. Pooling Dropout.

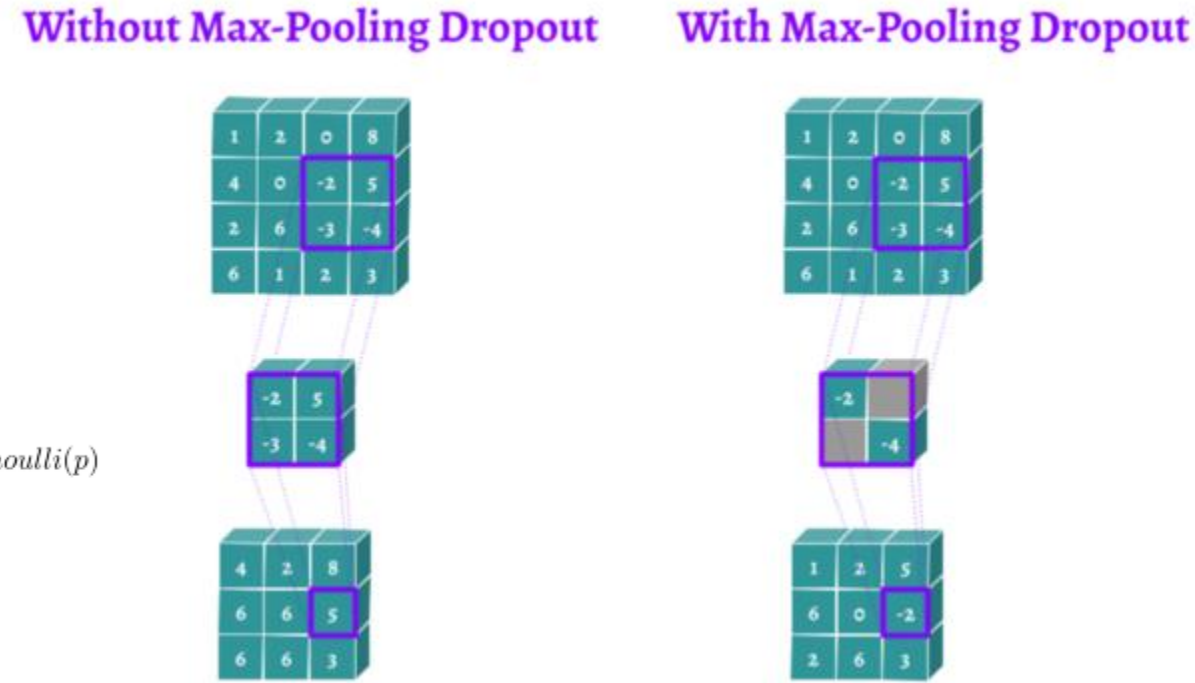
- ❑ Стандартный dropout не эффективен для изображений
- ❑ Pooling Dropout применяют для сверточных сетей

Training Phase :

$$\mathbf{Y} = max\{Pool_{size}(\mathbf{Y}) \circ \mathbf{M_{size}}\} \quad M_{ij} \sim Bernoulli(p)$$

Testing Phase :

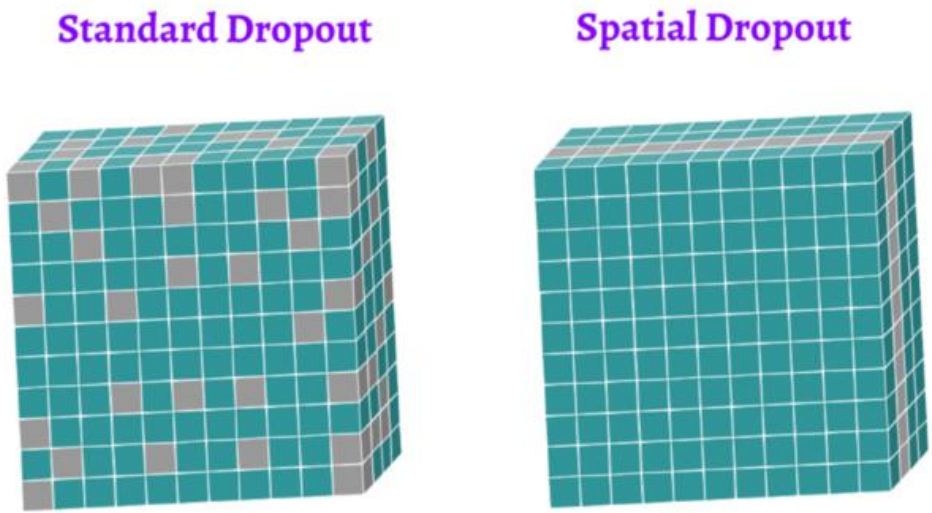
$$\mathbf{Y} = (1-p)\ max\{Pool_{size}(\mathbf{Y})\}$$



https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Варианты Dropout. Spatial Dropout.

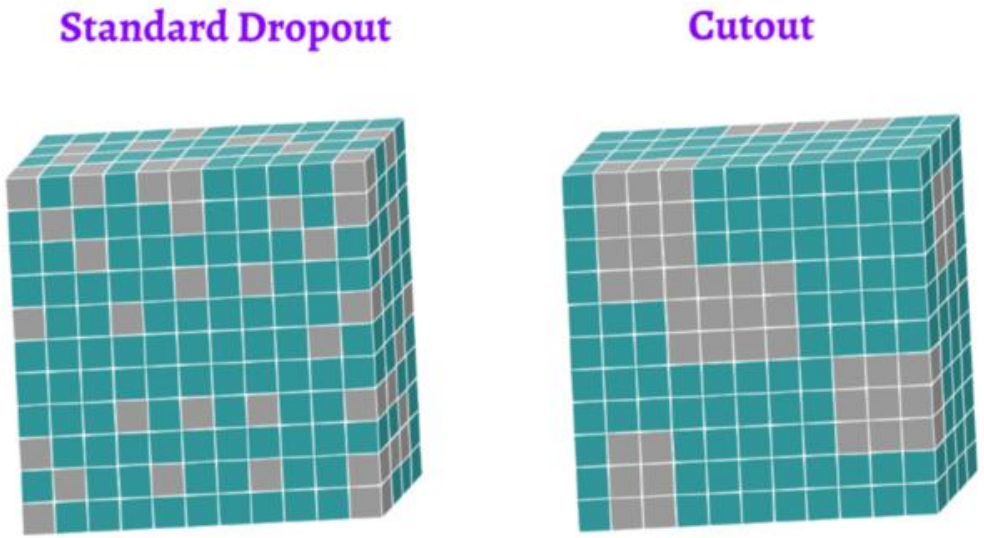
- ❑ Удаление карт признаков
- ❑ Spatial Dropout применяют для сверточных сетей



https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Варианты Dropout. Cutout.

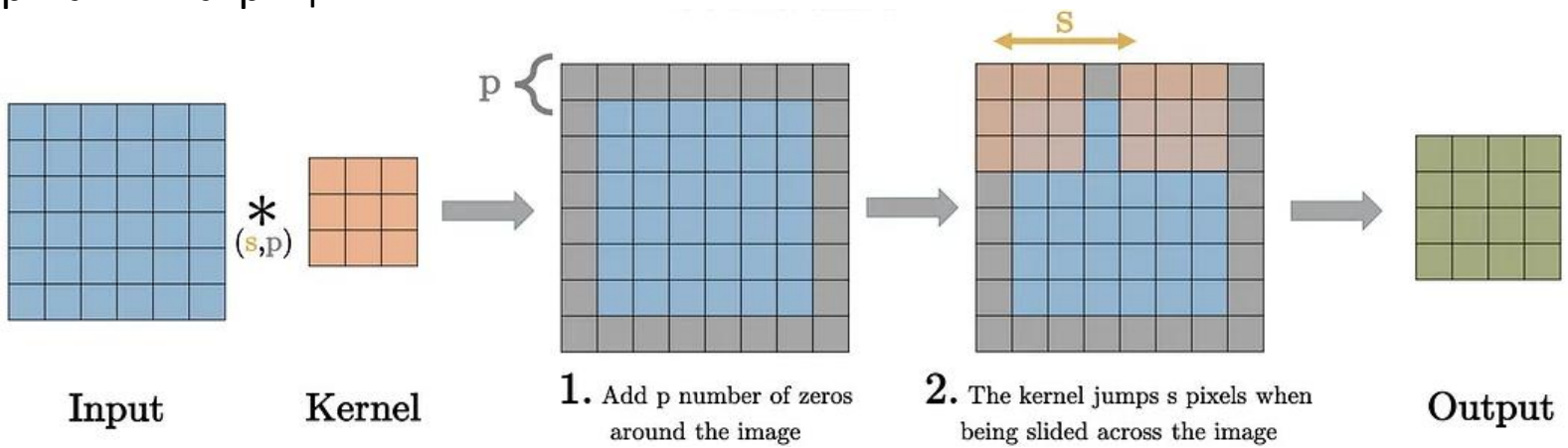
- Удаление фрагментов карт
- Cutout применяют для сверточных сетей



https://towardsdatascience.com/12-main-dropout-methods-mathematical-and-visual-explanation-58cdc2112293

Операция свертки

- Свертка 2D матриц



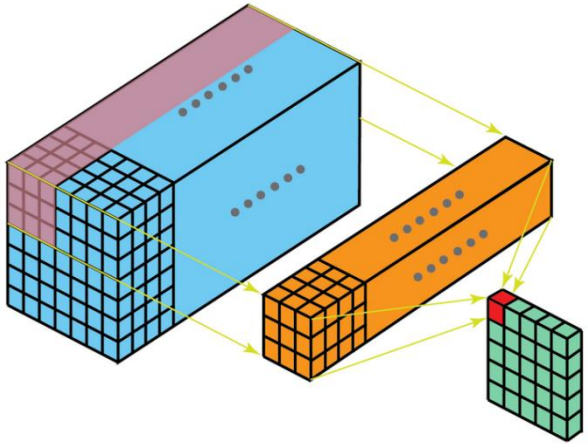
- P – число элементов вдоль границ, заполняемых нулями
- S – ширина сдвига ядра

$$o = \frac{i + 2p - k}{s} + 1$$ - Размер выходного массива (i- размер входного массива)

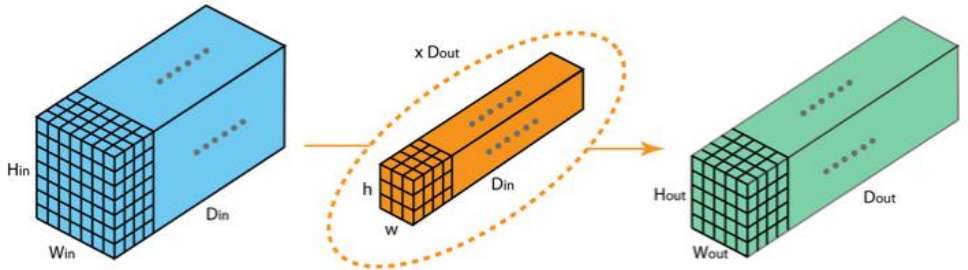
https://towardsdatascience.com/what-is-transposed-convolutional-layer-40e5e6e31c11

Типы сверток (1)

- 2D свертка



2D свертка: получение одного выходного элемента



Выполнение 2D свертки, число входных карт Din, выходных - Dout

- 1D свертка (аналогично для 2D тензора)

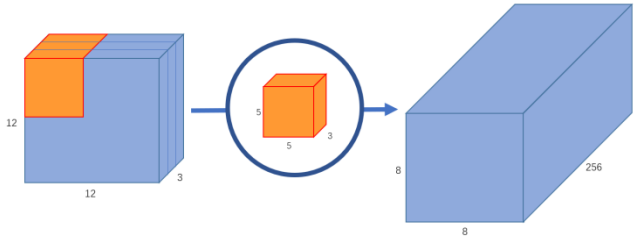
https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728

Типы сверток (2)

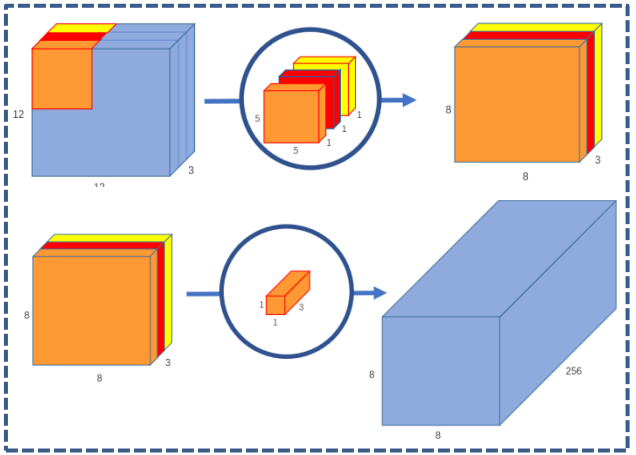
- Входной тензор: $WxHxC_I$;
- Ядро: K_1xK_2;
- Выходной тензор: $WxHxC_O$;
- Обычная свертка
  - Число операций: $K_1xK_2xC_IxWxHxC_O$;
- Свертка Depthwise Separable
  - Depthwise, число операций: $K_1xK_2xC_IxWxH$;
  - Pointwise, число операций: $C_IxWxHxC_O$;
  - Общее число операций: $K_1xK_2xC_IxWxH+ C_IxWxHxC_O$;

  +: Вычислительная эффективность

  -: Потеря точности



Обычная свертка, 1228800 операций

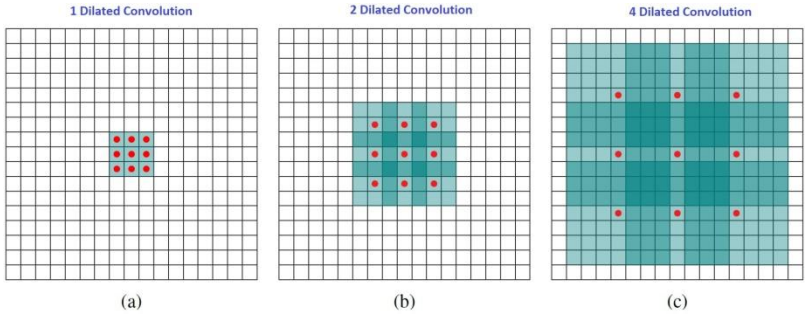


Depthwise Separable, 52952 операций

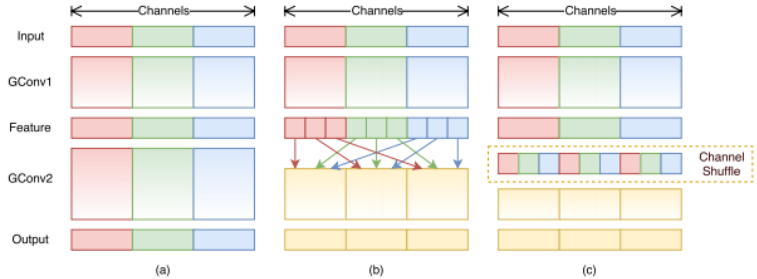
https://towardsdatascience.com/a-basic-introduction-to-separable-convolutions-b99ec3102728

Типы сверток (3)

- Dilated convolution
  - Прореживание рецептивного поля
  - Увеличение рецептивного поля с сохранением числа параметров и операций.
- Shuffle convolution
  - Поканальные свертки с дальнейшим перемешиванием результирующих карт.



Примеры dilated convolution

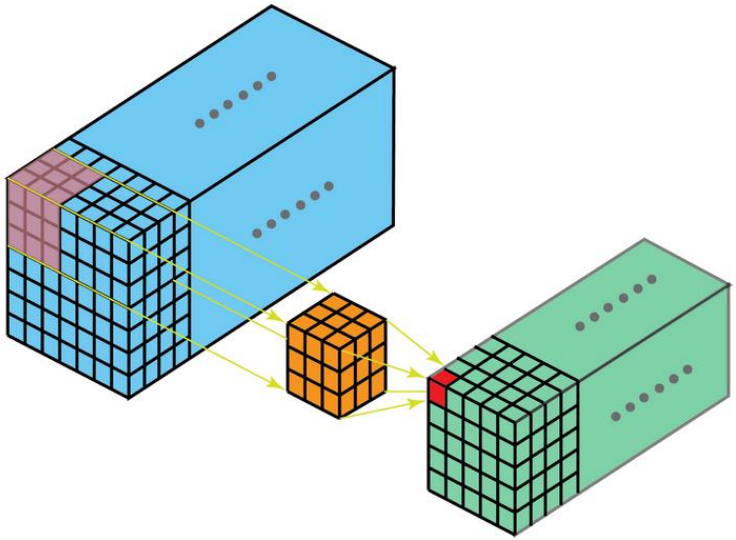


a. Поканальная свертка
b, c. Shuffle convolution

https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215

Типы сверток (4)

- 3D свертка
- Применяют для обработки 3D данных

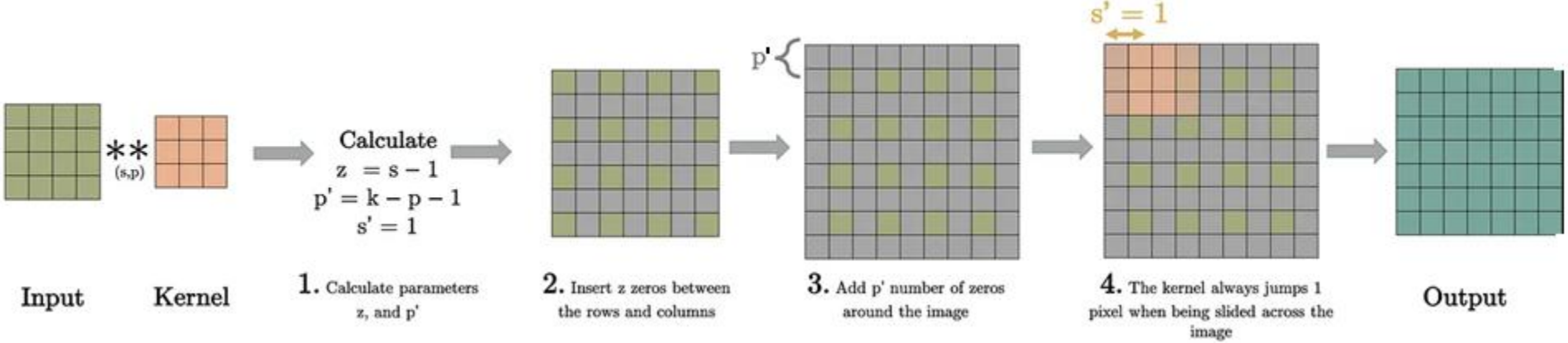


Выполнение 3D свертки

https://towardsdatascience.com/a-comprehensive-introduction-to-different-types-of-convolutions-in-deep-learning-669281e58215

Типы сверток (5)

- ❑ Транспонированная свертка (Transposed convolution) ≠обратная свертка (deconvolution)
- ❑ Применяют для получения выходных данных большего размера (автоэнкодеры, GAN)



- (S,P) – параметры
- z – число нулей, вставляемых между столбцами и строками)
- p’ – число нулей вокруг исходного массива
- s’ – шаг сдвига ядра

$o = (i-1) \times s + k - 2p$ - Размер выходного массива (i- размер входного массива)

https://towardsdatascience.com/what-is-transposed-convolutional-layer-40e5e6e31c11

Вопросы

- ❑ Почему Batch Normalization повышает эффективность обучения?
- ❑ Какие типы Dropout применяют для сверточных сетей?
- ❑ Почему Depthwise Separable свертка является вычислительно более эффективной во сравнению с обычной операцией свертки в НС?
- ❑ Как выполняется transposed convolution?