

CNN

Convolutional Neural Networks

CNN

Convolutional Neural Networks (CNN, ConvNet) wariant MLP inspirowany biologicznie, gdzie mnożenie macierzy wag i sygnału wejściowego zastąpione jest operacją splotu

Zdolne do generalizacji sygnału posiadającego relacje przestrzenne

- 1D sygnały czasowe
- 2D obrazy
- 3D fMRI, video, obrazy RGB
- sygnały wielokanałowe

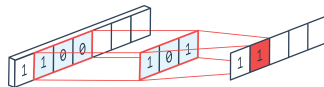
Potrafią być odporne na przestrzenne transformacje sygnału (skalowanie, obrót, przesunięcie)

Splot 1D

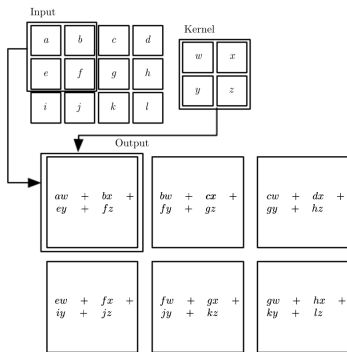
$$(x \star w)(t) = \int x(a)w(t - a)da$$

- wynikiem jest funkcja, np. uśredniona wartość x względem wszystkich wartości w jeśli w spełnia wymagania gęstości prawdopodobieństwa
- W sieciach splotowych CNN:
 x - sygnał wejściowy
 w - kernel (filtr), wagi połączeń neuronu, niezerowe tylko w ograniczonym obszarze (pole recepcyjne)
- wartości wyjściowe tworzą **mapę cech** (*feature map*)

$$s(i) = \sum_m x(m)w(i - m)$$



Splot 2D



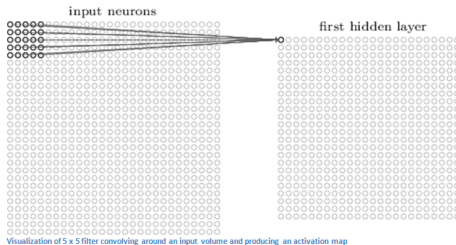
$$s(i, j) = \sum_l \sum_m w(l, m) \cdot x(i + l, j + m)$$

Filtry graficzne 2D – przykłady

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

<https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>

Pola recepcyjne i mapy cech

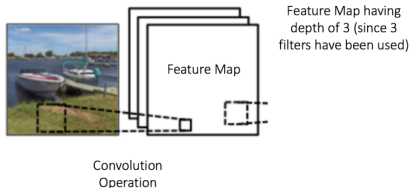


- **Pole recepcyjne** - obszar „widziany” przez neuron (rozmiar filtra)
- wyjście neuronu: spłot sygnału i liniowego filtra, ewentualny wyraz wolny (bias) i nieliniowa funkcja wyjściowa (ReLU, tanh), generują **mapę cech** (*feature map*)

$$s(i, j) = \sigma \left(b + \sum_l \sum_m w(l, m) \cdot x(i + l, j + m) \right)$$

Pola recepcyjne i mapy cech

- N neuronów (filtrów) tworzy N map (objętość, tensor n wymiarowy)

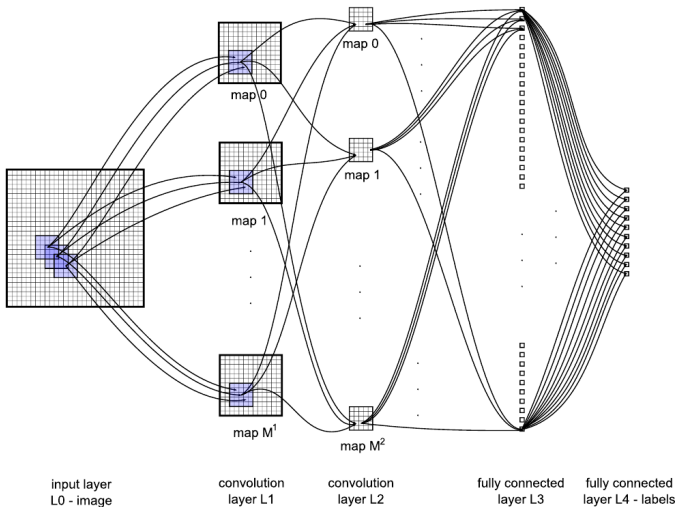


- Przy przetwarzaniu obrazów sygnał wejściowy x to tensor 3D: kanał (RGB), szerokość, wysokość

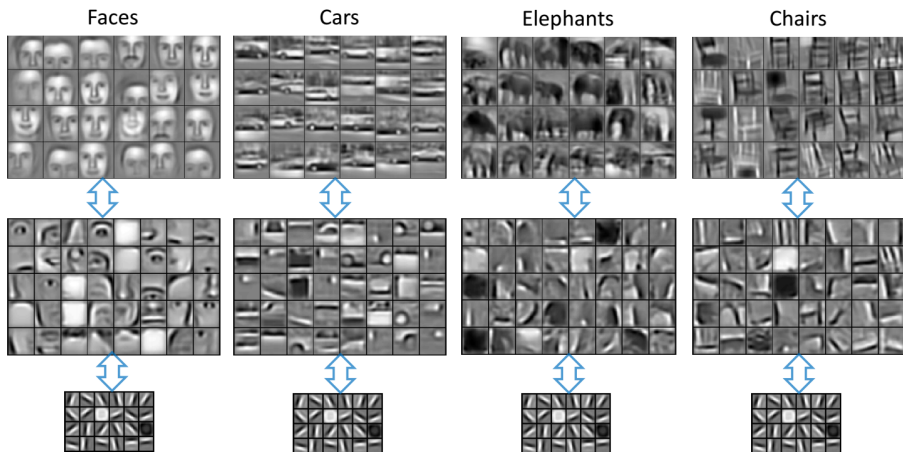
$$s(i, j, k) = \sum_{lmn} x(l, j + m, k + n)w(i, l, m, n)$$

- tensor wyjściowy warstwy zawierającej N filtrów (N, M_x, M_y) , gdzie M_x, M_y - rozmiary mapy wyjściowej
- w praktyce x rozszerzony do 4D o wymiar związany z rozmiarem mini-batcha

Typowa architektura CNN



Przykłady filtrów



Gooffellow, 2016 [?]

Rozmiar mapy cech

- Warstwa zawierająca N neuronów (filtrów) tworzy N map (objętość, tensor n wymiarowy)
- Rozmiar mapy wyjściowej (M_x^n, M_y^n) w warstwie n :

$$M_x^n = \frac{M_x^{n-1} - K_x^n + 2P_x^n}{S_x^n} + 1 \quad M_y^n = \frac{M_y^{n-1} - K_y^n + 2P_y^n}{S_y^n} + 1$$

K_x, K_y rozmiar filtra (szerokość i wysokość)

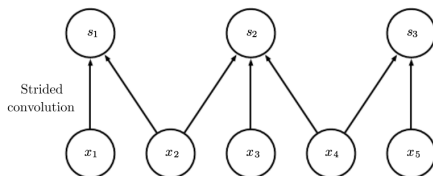
S_x, S_y przesunięcie (*stride*) w każdym z wymiarów,
splot 2D

$$s(i, j, k) = \sum_{lmn} x(l, j \times S_x + m, k \times S_y + n) w(i, l, m, n)$$

P_x, P_y wielkość rozszerzenia brzegu (*zero padding*),

Stride - krok przesunięcia

Przykład w 1D: S=2, K=3, P=0

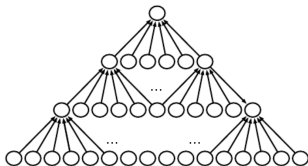


$$M^n = \frac{M^{n-1} - 3}{2} + 1$$

- redukcja wymiaru - zmniejszenie wymogów obliczeniowych i pamięciowych
- zmniejszenie rozdzielczości sygnału
- kosztem mniej dokładnej reprezentacji

Zero padding

valid zero padding, brak rozszerzenia na brzegach ($P = 0$)



$$M^n = \frac{M^{n-1} - K^n}{S^n} + 1$$

- rozmiar map maleje w kolejnych warstwach
- ogranicza to możliwość budowania głębokich sieci i wymusza stosowanie małych filtrów
- sygnał wejściowy na brzegach ma mniejszy wpływ na sygnał wyjściowy

Zero padding

same zero padding, brzegi wypełnione dodatkowymi wartościami (zerami) aby zapewniać odtworzenie wymiaru sygnału wejściowego $P = \lfloor \frac{K}{2} \rfloor, S = 1$



- pozwala budować bardzo głębokie sieci o dowolnych wielkościach filtrów $M_i^n = \lceil \frac{M_i^{n-1}}{S_i} \rceil$
- sygnał wejściowy na brzegach ma mniejszy wpływ na sygnał wyjściowy (na brzegach 0)

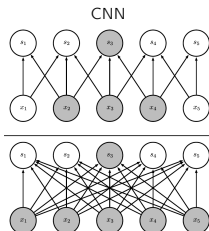
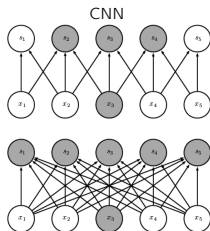
Właściwości CNN

- **rzadka reprezentacja** - rozmiar filtra jest dużo mniejszy od rozmiaru sygnału wejściowego, pojedyncze wejście oddziałuje tylko na grupę neuronów
- **współdzielenie parametrów** - warstwa splotowa może być widziana jako w pełni połączona warstwa ze współdzielonymi wagami (dużo mniejsza liczba parametrów w stosunku do MLP)
- **równoważność względem przesunięcia** sygnału - ta sama cecha znajdująca się w różnych miejscach obrazu będzie aktywowała ten sam filtr
- możliwość użycia sygnału wejściowego o zmiennym rozmiarze - większy obraz wejściowy wygeneruje większe mapy, w przypadku MLP zwiększenie rozmiaru wektora wejściowego wymaga rozbudowy architektury

Rzadka reprezentacja

pojedyncze wejście aktywuje tylko grupę neuronów w małym obszarze

wyjście neuronu zależy od małego obszaru sygnału wejściowego

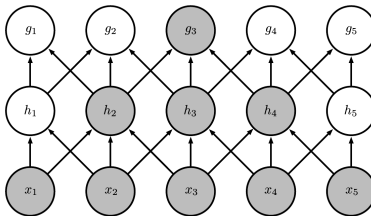


MLP: mnożenie macierzy $O(m \times n)$ parametrów
CNN: warstwa splotowa $O(k \times n)$, gdzie $k \ll m$

Źródło grafiki: Goofellow, 2016 [?]

Efektywne pole recepcyjne

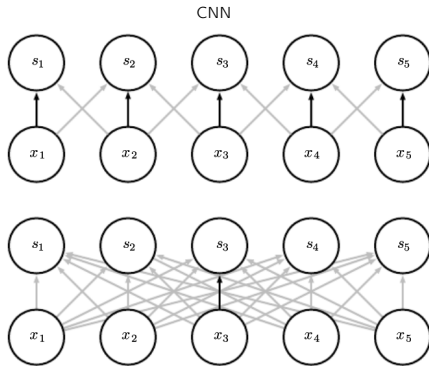
- **Efektywne pole recepcyjne** - obszar sygnału wejściowego pokryty przez neurony w wyższych warstwach, rośnie z głębokością



- stride, pooling i dilation (rozrzedzony spłot) dodatkowo zwiększają efektywne pole recepcyjne
- pomimo rzadkich połączeń sieć jest w stanie w ten sposób modelować złożone zależności

Współdzielenie wag

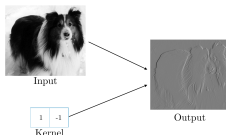
- ta sama waga jest używana przy przetwarzaniu każdego punktu wejściowego
- pojedynczy filtr pozwala wykryć tę samą cechę w różnych położeniach obrazu wejściowego (*equivariance to translation*)



Źródło grafiki: Goofellow, 2016 [?]

Rzadkie połączenia i współdzielenie wag

Przykład: wykrywanie krawędzi dla obrazu $n \times n$ w poziomie



- MLP: mnożenie pełnej macierzy $n^2 \times n^2 = n^4$
- CNN: splot filtrem 1x2 (2 wagi) wymaga $n^2 \times 3$ operacji (2 mnożenia i dodawanie)
- splot drastycznie bardziej wydajny w mapowaniu powtarzających się relacji w małych, lokalnych rejonach sygnału wejściowego

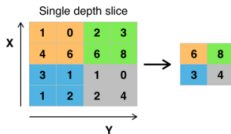
Źródło grafiki: Goofellow, 2016 [?]

Równoważność przy przesunięciu

- przesunięcie sygnału wejściowego powoduje identyczne przesunięcie sygnału wyjściowego na mapie cech
- własność pożądana w rozpoznawaniu obrazów, gdzie lokalne cechy (np. krawędzie) mogą wystąpić w każdym miejscu na obrazie
- współdzielenie wag dla całego wejścia nie zawsze jest pożądane (różne filtry w różnych obszarach obrazu), np. rozpoznawanie twarzy ze zdjęć paszportowych, posiadają charakterystyczne cechy występujące wyłącznie w określonych rejonach sygnału wejściowego
- spłot nie jest niezmienniczy względem zmiany skali lub obrotu obrazu, osiąga się to poprzez rozszerzenie zbioru treningowego o przypadki zdeformowane (szum, skalowanie, obrót, zmiana kontrastu, etc..)

Pooling

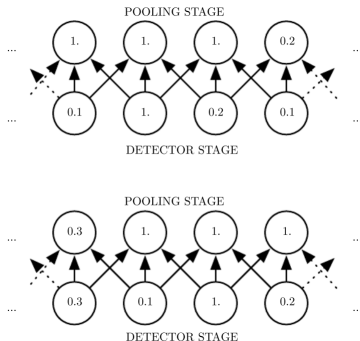
- typowa warstwa w sieciach CNN:
 - liniowa aktywacja (splot)
 - detekcja (nieliniowość np. ReLU)
 - funkcja redukcji (pooling) - „uogólnienie” wartości sąsiadujących wyjść
- **Max pooling** - maksimum z pewnego podobszaru



- redukcja wymiarowości - przesunięcie S filtra typowo równe jego wielkości K (maksimum z rozłącznych obszarów)
- inne podejścia: *avg. pooling* (średnia z sąsiedztwa), *norm pooling* (norma z sąsiadujących wyjść), ważona średnia od centrum, niektóre architektury rezygnują z tej warstwy

Pooling względem obrazu wejściowego

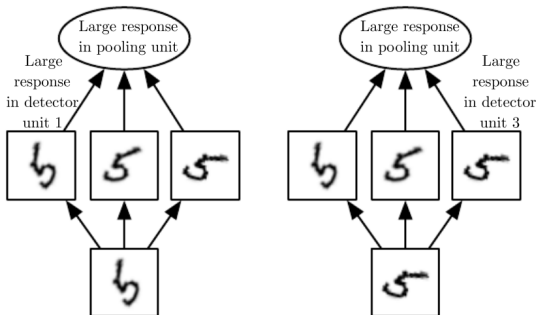
Pooling zapewnia niezmienniczość względem drobnego przesunięcia obrazu wejściowego



Źródło grafiki: Gooffellow, 2016 [?]

Pooling względem map cech

Redukcja względem wyjść różnych splotów umożliwia wprowadzenie do modelu niezmienniczości względem pewnych transformacji sygnału wejściowego (np. obrotu)



Źródło grafiki: Gooffellow, 2016 [?]

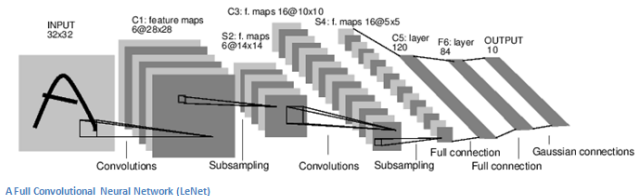
LeNet5 (LeCun 1998)

klasyfikacja cyfr pisanych ręcznie
(odczytywanie czeków)

👉 MNIST

trening: 60k cyfr, 250 osób,

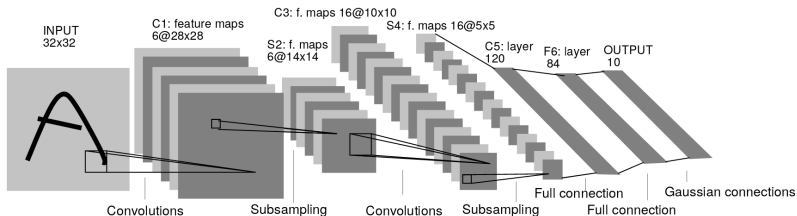
test: 10k cyfr



👉 LeNet-5, convolutional neural networks

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. *Gradient-based learning applied to document recognition. Proceedings of the IEEE, november 1998*

LeNet-5



Layer	kernels	size	output	connections	parameters
Input			1x32x32		
C1	6	5x5	6x28x28	122304	156
S2		2x2	6x14x14	5880	12
C3	16	5x5	16x10x10	151600	1516
S4		2x2	16x5x5	2000	32
C5	120	5x5	120x1x1	48120	48120
F6	84		84x1x1	10164	10164
Output	10		10x1x1		

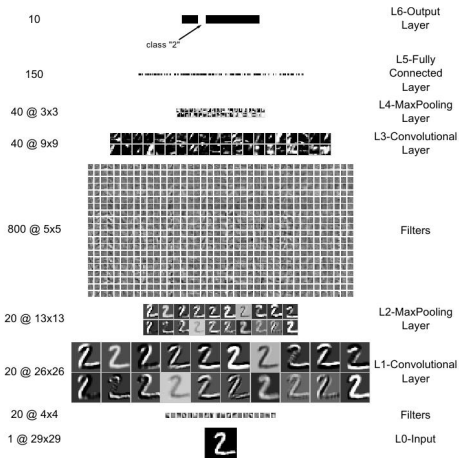
LeNet-5 połączenia S2-C3

Mapy C3 połączone wyłącznie z wybranymi mapami z S2

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

- redukcja liczby połączeń
- przełamanie symetrii sieci - różne sygnały wejściowe pozwalają uzyskać różnorodne (być może komplementarne) zestawy detektorów cech

Przykład wizualizacji aktywacji sieci

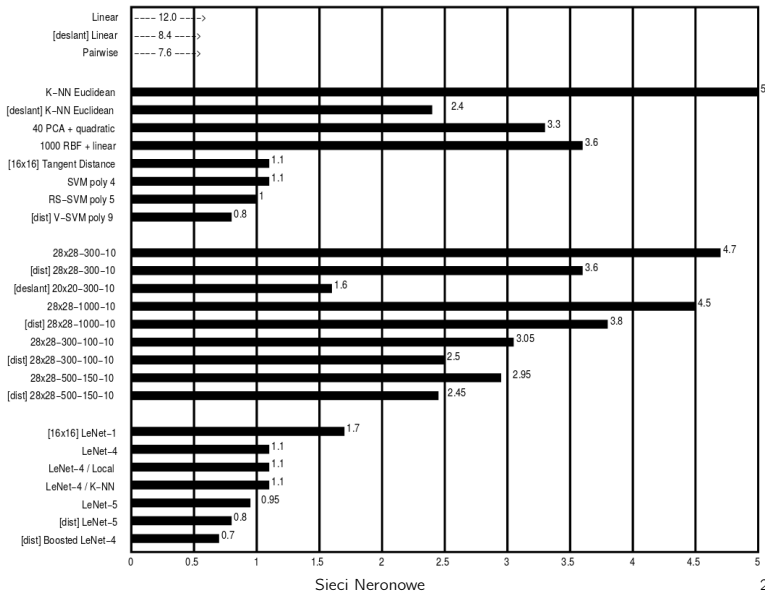


👉 Wizualizacja sieci splotowej

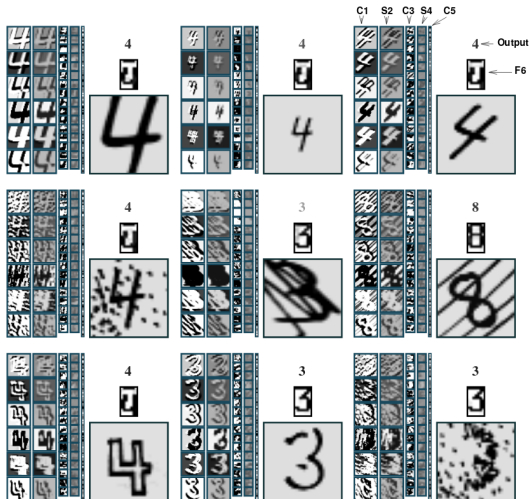
LeNet-5 błędy klasyfikacji



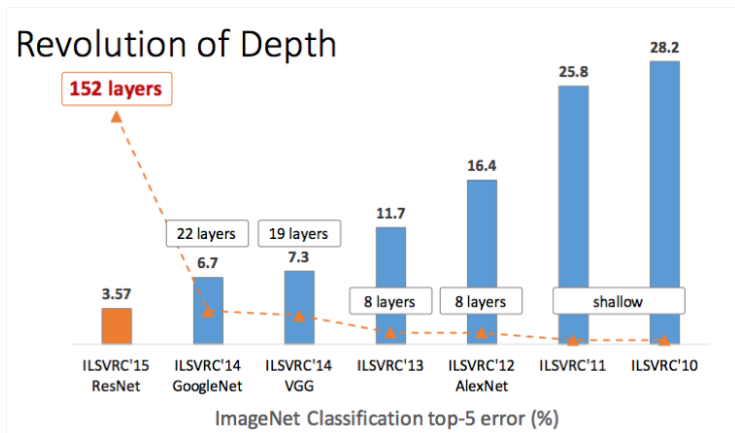
LeNet-5 porównanie z innymi metodami



LeNet-5 odporność na szum, zniekształcenia i nietypowe przypadki



ImageNet



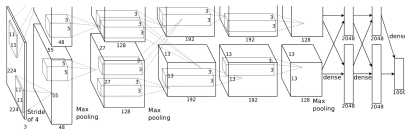
- 👉 CNNs Architectures used on ImageNet

AlexNet (A. Krizhevsky, 2012)

- Zwycięzca 🖱️ ImageNet w 2012 z poprawnością 15.3% (poprawa z 26%)
- 1.2M obrazów, 1000 klas



mushroom

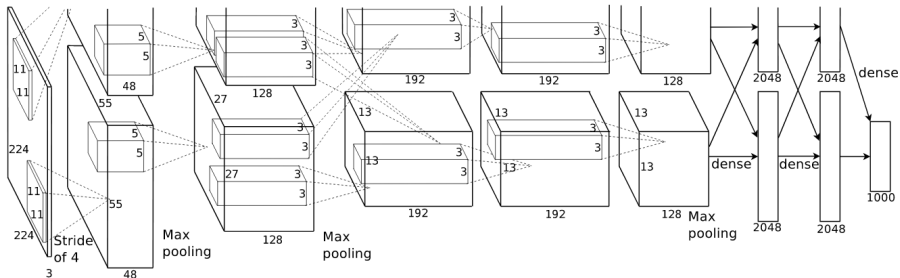


- architektura wzorowana LeNet5 ale głębsza (8 warstw) i więcej filtrów na warstwę (11x11, 5x5, 3x3)
- sekwencje warstw splotowych z jednostkami ReLU
- regularyzacja: dropout, L2, rozszerzanie danych, normalizacja wyjść wybranych warstw
- SGD z momentem, 20 epok, 6 dni treningu (2x NVIDIA GTX 580 3GB GPUs), 60M parametrów

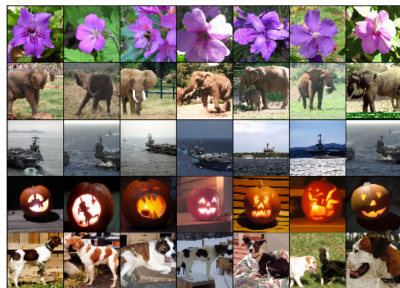
A. Krizhevsky, I. Sutskever, G.E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, 2012

AlexNet

Layer	kernels	size	stride	output
Input				224x224x3
C1 + LRN	96	11x11	4x4	
MaxPooling		3x3	2x2	55x55x96
C2 + LRN	256	5x5		
MaxPooling		3x3	2x2	27x27x256
C3	384	3x3		13x13x384
C4	384	3x3		13x13x384
C5	256	3x3		
MaxPooling		3x3	2x2	13x13x256
F6 + dropout($p = 0.5$)	4096			
F7 + dropout($p = 0.5$)	4096			
Output softmax	1000			

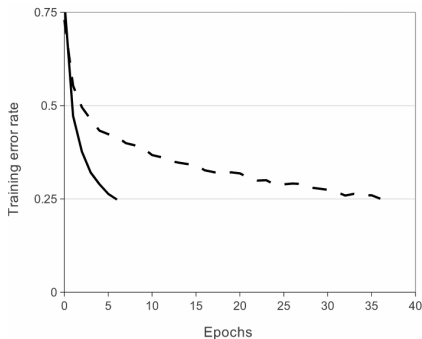


AlexNet



obraz wejściowy i 6 najbliższych wzorców względem odległości Euklidesowej wektora pobudzeń ostatniej warstwy ukrytej

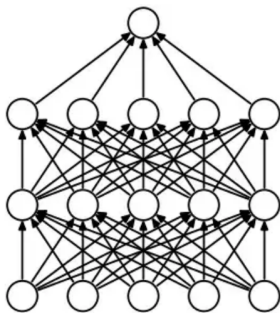
5 najsilniejszych odpowiedzi



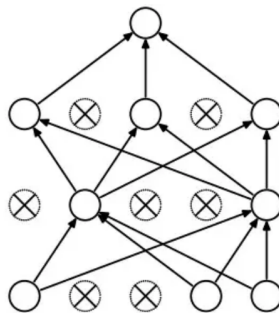
- Porównanie szybkości zbieżności 4 warstwowej sieci splotowej z jednostkami ReLU i tanh na danych ImageNet
- Zastosowanie ReLU do kilkukrotnego przyspieszenia zbieżności
- Inicjalizacja: wagi z rozkładu Gaussa $N(0, 0.01)$, obciążenia (bias) $b = 1$, stąd większość ReLU aktywnych na początku treningu

Dropout

Dropout (Srivastava et al., 2014) dla każdego wzorca treningowego z prawdopodobieństwem p „wyrzuca” jednostki (zeruje ich aktywacje), odrzucone jednostki nie biorą udziału w treningu



(a) Standard Neural Net



(b) After applying dropout.

Rys:  ResNet, AlexNet, VGGNet, Inception: Understanding various architectures of Convolutional Networks by Koustubh Sinhal

Dropout

- W czasie ewaluacji dropout jest wyłączany ($p = 1$)
- Każdy krok uczenia odbywa się ze zmienioną losowo architekturą sieci ale wagi są współdzielone,
- Dla n jednostek mamy 2^n możliwych architektur, trening w mini-batchu uśrednia gradient względem różnych , wylosowanych sieci
- Przeciwdziała powstawaniu złożonych relacji pomiędzy wieloma neuronami, stąd pojedynczy neuron jest zmuszony wykrywać bardziej wartościowe cechy, niezależne od wpływu innych neuronów
- Wolniejsza zbieżność (AlexNet 2 x wolniej) ale silnie zapobiega przeuczeniu

Local response normalization

$$b_{x,y}^i = a_{x,y}^i / \left(k + \alpha \sum_{j=\max(0, i-n/2)}^{\min(N-1, i+n/2)} (a_{x,y}^j)^2 \right)^\beta$$

dla kanału i oraz pikseli w pozycji x, y

- normalizacja wartości wyjściowych ReLU
- $k = 2, n = 5, \alpha = 10^{-4}, \beta = 0.75$ dobrane heurystycznie ze zbioru walidacyjnego
- realizuje normalizację jasności dla n sąsiadujących kolejnych
- w AlexNet poprawa o 1.4% (top 1) oraz 1.2% (top 5)

Dodatkowa regularyzacja

Data augmentation

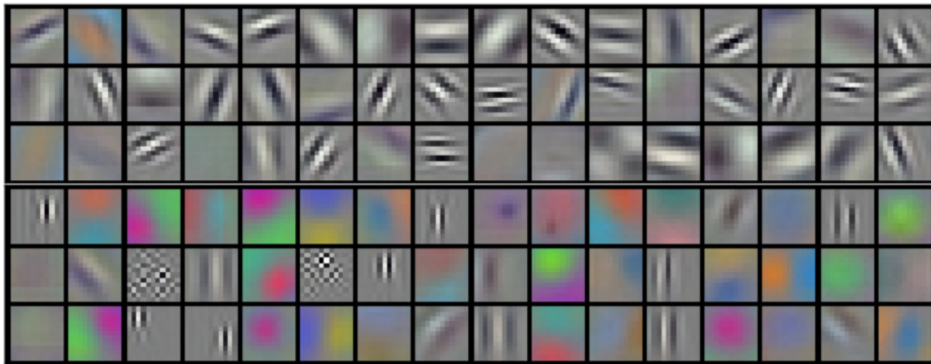
- przesunięcie i odbicia: losowanie obrazków 224×224 z 256×256 , zwiększenie zbioru 2048 razy niezbędne do uniknięcia przeuczenia przy tak dużej sieci
- losowa modyfikacja intensywności kanałów RGB
- poprawa ponad 1% (top 1, top 5)

MaxPooling z nakrywaniem

- kernel 3×3 , stride 2×2
- poprawa 0.4% (top 1) i 0.3% (top 5) względem próbkowania bez nakrywania (kernel 2×2)
- obserwacja: AlexNet rzadziej ulegał przeuczeniu

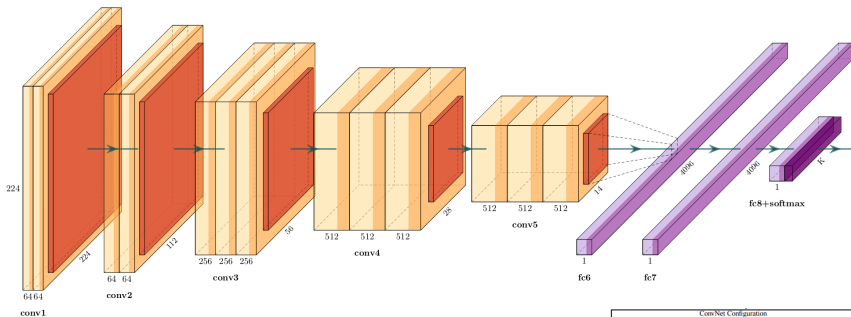
Zastosowanie 2 rdzeni GPU pozwoliło na szkolenie sieci o większej liczbie filtrów co zaowocowało poprawą 1.7% (top-1) oraz 1.2% (top-5) w porównaniu z mniejszą siecią na 1 rdzeniu

Filtry w pierwszej warstwie



96 filtrów pierwszej warstwy splotowej, osobne kolumny warstw splotowych uczą się wykrywania innego typu relacji, obserwowane przy każdym treningu

VGGNet (Simonyan and Zisserman, 2014)



- bloki warstw splotowych + max pooling
- wielkość map zmniejszana o połowę po każdym bloku
- ilość filtrów zwiększana 2 krtnie po każdym bloku

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 x 224 RGB image)					
conv3-64	conv3-64	conv3-64	conv3-64	conv3-64	conv3-64
	LRN	conv3-64	conv3-64	conv3-64	conv3-64
conv3-128	conv3-128	conv3-128	conv3-128	conv3-128	conv3-128
		conv3-128	conv3-128	conv3-128	conv3-128
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
conv3-256	conv3-256	conv3-256	conv3-256	conv3-256	conv3-256
		conv3-256	conv3-256	conv3-256	conv3-256
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
		conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
conv3-512	conv3-512	conv3-512	conv3-512	conv3-512	conv3-512
		conv3-512	conv3-512	conv3-512	conv3-512
maxpool					
FC-4096					
FC-4096					
FC-1000					
softmax					

VGGNet

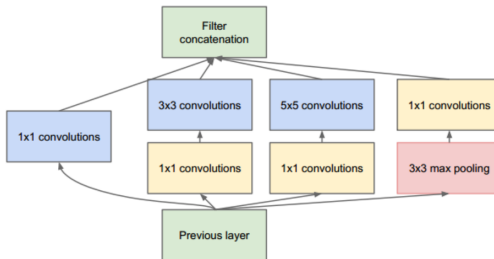
- zamiast dużych filtrów (11x11, 7x7, 5x5) stosuje we wszystkich warstwach filtry 3x3 zwiększając ich efektywne pola recepcyjne sekwencjami warstw splotowych
- małe filtry 3x3 zdolne do wykrycia bardziej subtelnych relacji w obrazach
- zwiększenie głębokości pozwala trenować bardziej złożone cechy
- stride=1, brak utraty informacji, gęsta konwolucja
- kilka architektur od 11 do 19 warstw (VGG16, VGG19)
- 138M parametrów
- trening 2-3 tygodnie (4x GPU), duże wymagania pamięciowe i obliczeniowe

K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv technical report, 2014

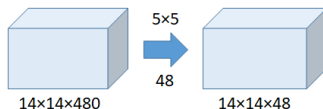
GoogLeNet/Inception

Inception module

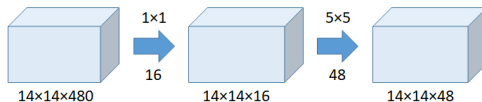
- mały stosunek rozmiaru mapy do ilości filtrów, np. 128 filtrów 3x3 i 32 filtry 5x5
- równoległe sploty o różnej wielkości filtrów 1x1, 3x3, 5x5 - detektory cech o różnej skali
- każdy większy filtr poprzedzony splotem 1x1 w celu redukcji liczby parametrów (redukcja liczby kanałów do 1)



Redukcja złożoności splotem 1x1



liczba operacji $(14 \times 14 \times 48) \times (5 \times 5 \times 480) = 112.9\text{M}$

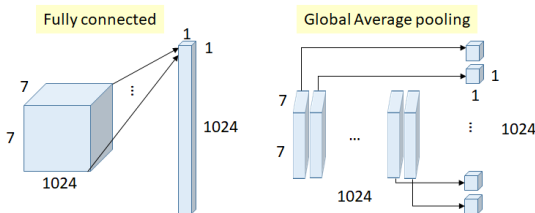


liczba operacji

$$(14 \times 14 \times 16) \times (1 \times 1 \times 480) + (14 \times 14 \times 48) \times (5 \times 5 \times 16) = 1.5\text{M} + 3.8\text{M} = 5.3\text{M}$$

Global average pooling

Globalne uśrednienie względem kanałów (global average pooling) zamiast pełnej połączonej warstwy za ostatnią warstwą splotową



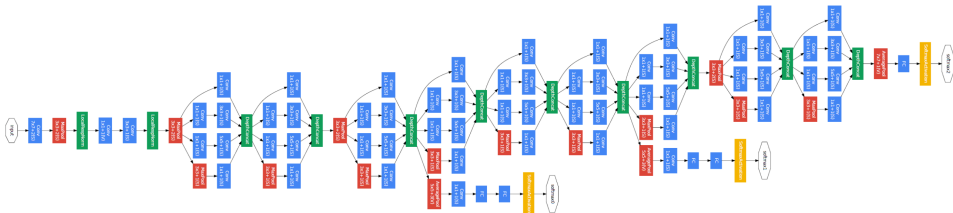
$$7 \times 7 \times 1024 \times 1024 = 51.3\text{M}$$

- Warstwy w pełni połączone zawierają najwięcej parametrów (w AlexNet 90% parametrów) w sieci
- global average pooling: 0 wag, uśrednienie wartości dla poszczególnych kanałów

Rys:  Review: *GoogLeNet (Inception v1)*— Winner of ILSVRC 2014, (Image Classification) by Sik-Ho Tsang

- GoogleNet poprawa poprawności top-1 o 0.6%

GoogLeNet/Inception



- 22 warstwy
- warstwy przewężające (bottleneck) - dodatkowo redukują złożoność sieci
- kilka wyjść softmax
- poprawność 93.3% top-5 na ImageNet, dużo szybszy w treningu od VGG

ResNet (Residual Neural Network, Kaiming He et. al 2015)

Bloki ze skrótami (residual module)

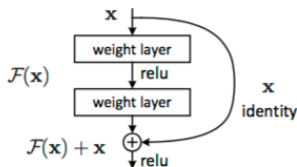
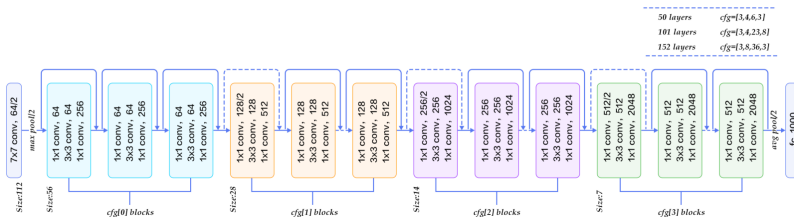


Figure 2. Residual learning: a building block.

- „skrót” łączy wejścia bloku z wyjściem poprzez odwzorowanie jednostkowe
- przejścia „skrótowe” pomagają uczyć bardzo głębokie sieci (152 warstwy)
- zapobiegają zanikaniu gradientu (sygnał może być propagowany skrótami)

ResNet



- wiele bloków zawierających sploty 1x1, 3x3, 1x1
- regularyzacja: batch normalization

Batch normalization

Batch normalization (Ioffe, 2015) normalizuje wejścia $x_i^{(k)}$ warstwy względem wartości średniej $\mu^{(k)}$ i wariancji $\sigma^{(k)}$ wzdłuż wymiaru k dla mini pakietu

$$y_i^{(k)} = \gamma^{(k)} \hat{x}_i^{(k)} + \beta^{(k)}$$

gdzie γ i β podlegają adaptacji w czasie treningu
Czynnik normalizujący każde wejście

$$\hat{x}_i^{(k)} = \frac{x_i^{(k)} - \mu_B^{(k)}}{\sqrt{\sigma_B^{(k)^2} + \epsilon}}$$

średnia i odchylenie dla minipakietu o rozmiarze m

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i, \quad \sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2$$

Ioffe, Sergey; Szegedy, Christian (2016). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift".

Batch normalization

- przyspiesza trening, poprawia stabilność treningu oraz polepsza generalizację
- rozwiązuje problem przesunięcia kowariancji - gdy zmienia się wyjście warstwy poprzedniej wówczas warstwa następna musi dopasować się do nowego rozkładu danych
- można stosować większe kroki uczenia bez obawy znikającego lub eksplodującego gradientu
- większa odporność na wpływ różnorodnej specjalizacji i metod treningu
- wygładza powierzchnię błędu (Santurkar, 2018)