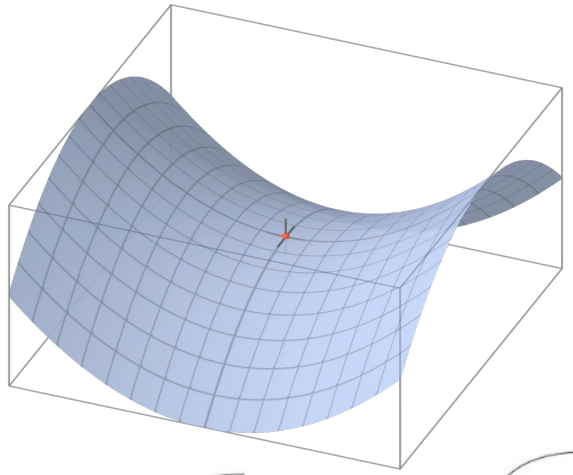
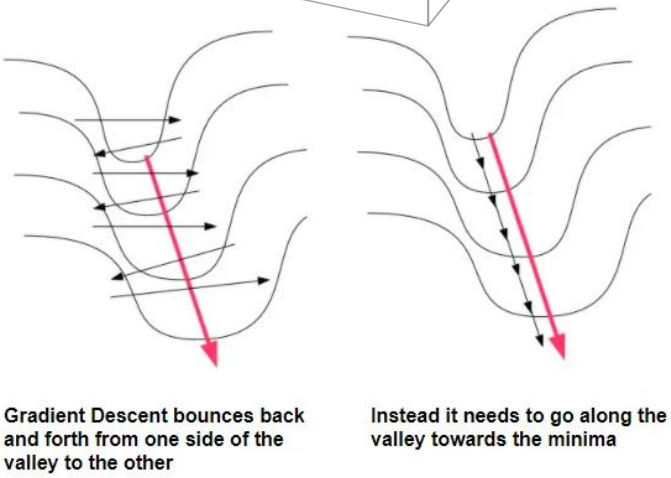
Badly conditioned regions

- regiony funkcji straty, w których metody pierwszego stopnia kiepsko sobie radzą, np. punkty siodłowe, wąwozy
- sposoby na poprawę zbieżności:
  - użycie metod drugiego rzędu
  - zmiana danych i architektury
  - modyfikacje metod pierwszego rzędu
- ponieważ największą część błędu aproksymacji zależy od Hesjanu, *conditioning* regionu można estymować za pomocą wartości własnych Hesjanu



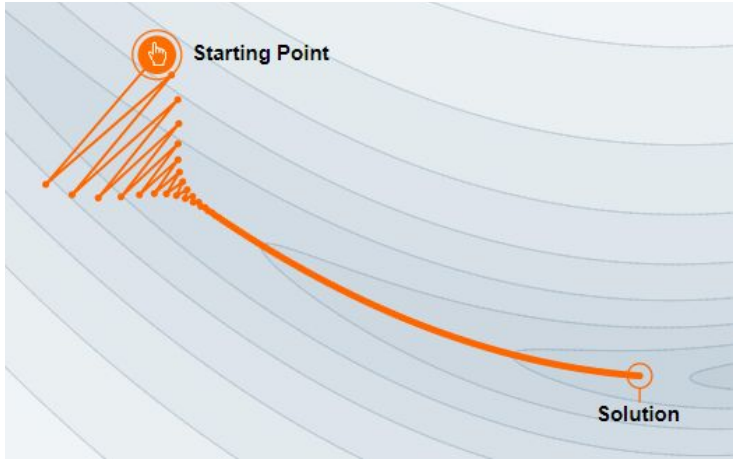


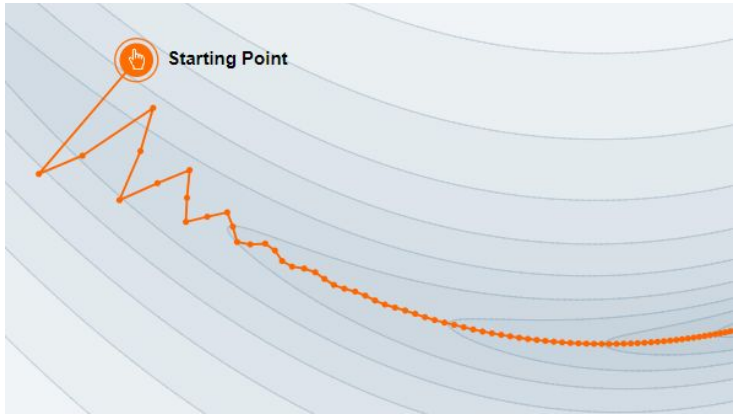
https://upload.wikimedia.org/wikipedia/commons/4/40/Saddle_point.png

https://towardsdatascience.com/neural-network-optimizers-made-simple-core-algorithms-and-why-they-are-needed-7fd072cd2788

Momentum

- *badly conditioned regions* mogą sprawić, że SGD przestanie robić postępy
- momentum zapobiega temu przy użyciu średniej kroczącej





krok SGD:

$$v_i = -\eta \nabla \mathcal{L}(w_i)$$

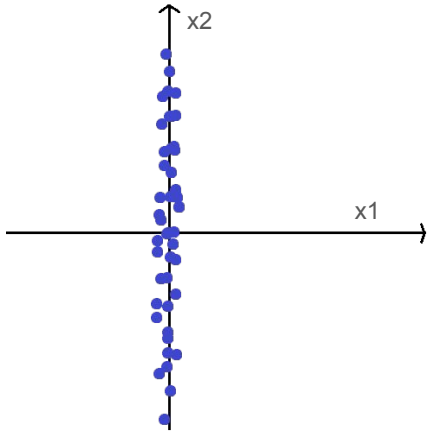
krok SGD z momentum:

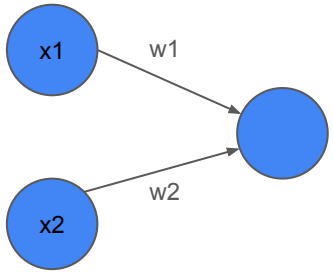
$$v_i = \beta v_{i-1} - \eta \nabla \mathcal{L}(w_i)$$

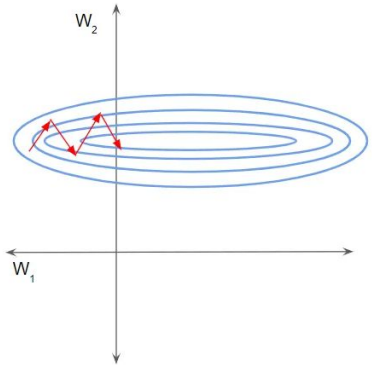
https://distill.pub/2017/momentum/

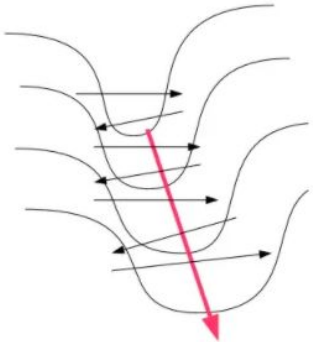
Normalizacja wejścia

- wejścia o różnych skalach samoistnie tworzą wąwozy
- gdy x_2 jest duże, gradient wzdłuż w_2 jest bardzo stromy, co nie pozwala SGD skoncentrować się na w_1
- aby temu zapobiec, wejście sieci normalizuje się odejmując średnią i dzieląc przed odchylenie standardowe
- gdy wejściem są obrazy, normalizacja oznacza zwykle skalowanie każdego kanału (a nie każdego piksela)









https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739

Inicjalizacja

- wybór punktu startowego optymalizacji wpływa na rezultaty
- losowe wagi (z jednego rozkładu) zwykle prowadzą do złych wyników
- pomysł: inicjalizacja wag takim rozkładem, że wariancja wyjścia warstwy jest taka sama, co wejścia
- dzięki temu wejście poszczególnych warstw jest znormalizowane, przynajmniej na początku uczenia, co częściowo zapobiega wąwozom

ReLU: Kaiming/He
rozkład normalny

$$W \sim \mathcal{N}\left(0, \frac{2}{n^l}\right)$$

sigmoid: Xavier/Glorot
rozkład normalny

$$W \sim \mathcal{N}\left(0, \frac{1}{n^l}\right)$$

gdzie n^l = liczba wejść

Momentum i dobra inicjalizacja umożliwiły użycie SGD

On the importance of initialization and momentum in deep learning

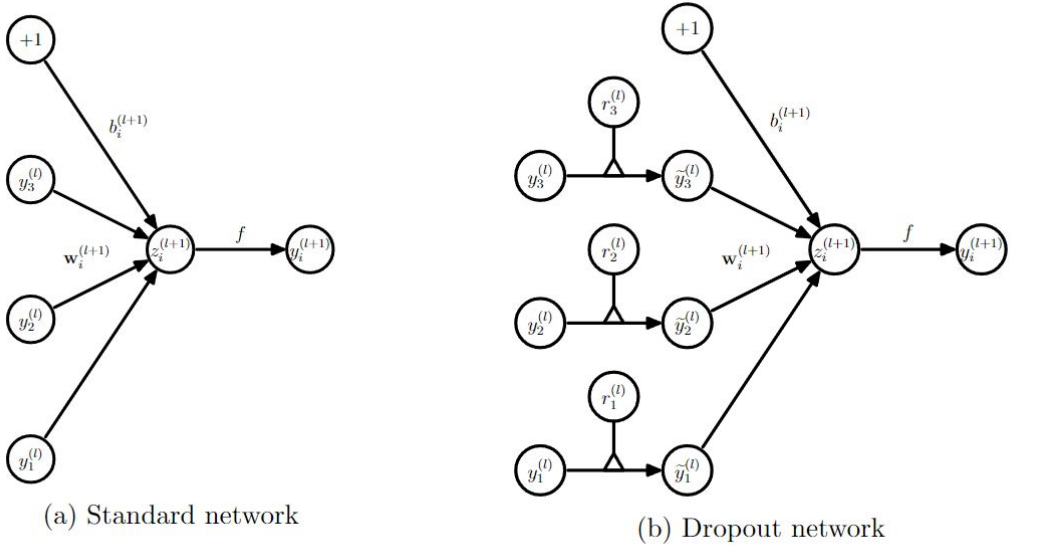
Proceedings of the 30th International Conference on Machine Learning 2013 · Ilya Sutskever, James Martens, George Dahl, Geoffrey Hinton · Edit social preview

Deep and recurrent neural networks (DNNs and RNNs respectively) are powerful models that were considered to be almost impossible to train using stochastic gradient descent with momentum. In this paper, we show that when stochastic gradient descent with momentum uses a well-designed random initialization and a particular type of slowly increasing schedule for the momentum parameter, it can train both DNNs and RNNs (on datasets with long-term dependencies) to levels of performance that were previously achievable only with Hessian-Free optimization. We find that both the initialization and the momentum are crucial since poorly initialized networks cannot be trained with momentum and well-initialized networks perform markedly worse when the momentum is absent or poorly tuned. Our success training these models suggests that previous attempts to train deep and recurrent neural networks from random initializations have likely failed due to poor initialization schemes. Furthermore, carefully tuned momentum methods suffice for dealing with the curvature issues in deep and recurrent network training objectives without the need for sophisticated second-order methods.

Eksperymenty:
https://wandb.ai/podcast-o-rybach-warsaw-university-of-technology/iml_lab2/reports/IML-Lab-2--Vmlldzo5OTk2OTA0?accessToken=podfkih4w73w3doiz9hjnd8ayb9fg0kw9jz8pfmut6y8h9wh39o1y6brzplvqoxh

Dropout

- technika zapobiegania *overfitting*'owi
- losowy podzbiór wejść do wybranej warstwy zostaje zamieniony na zera
- w teorii zmusza to warstwę do nauczenia się wielu sposobów na otrzymanie tego samego wyniku



(a) Standard network (b) Dropout network

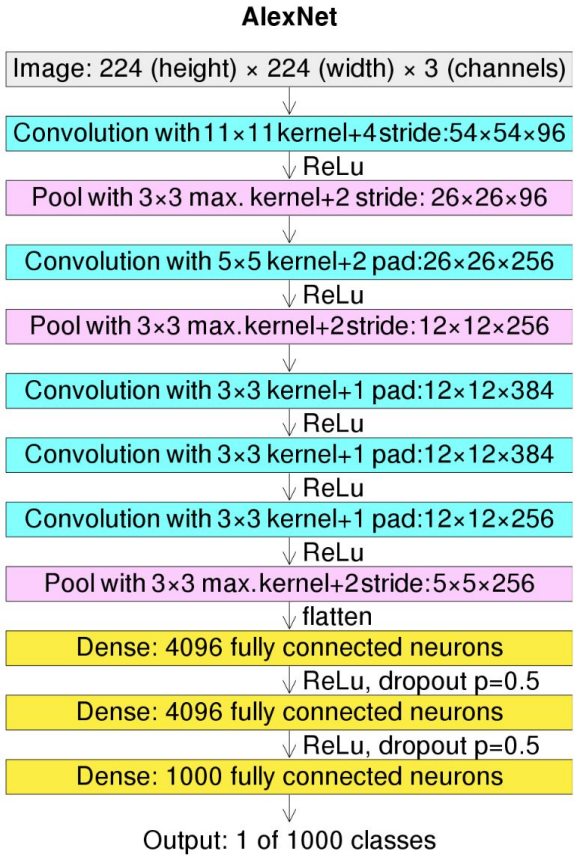
Figure 3: Comparison of the basic operations of a standard and dropout network.

- podczas *inference* wejścia nie są zerowane, są za to skalowane
- znacząco wydłuża uczenie. Używany głównie przed warstwami FC

Dropout: A Simple Way to Prevent Neural Networks from Overfitting, N. Srivastava et al. (2014)

AlexNet

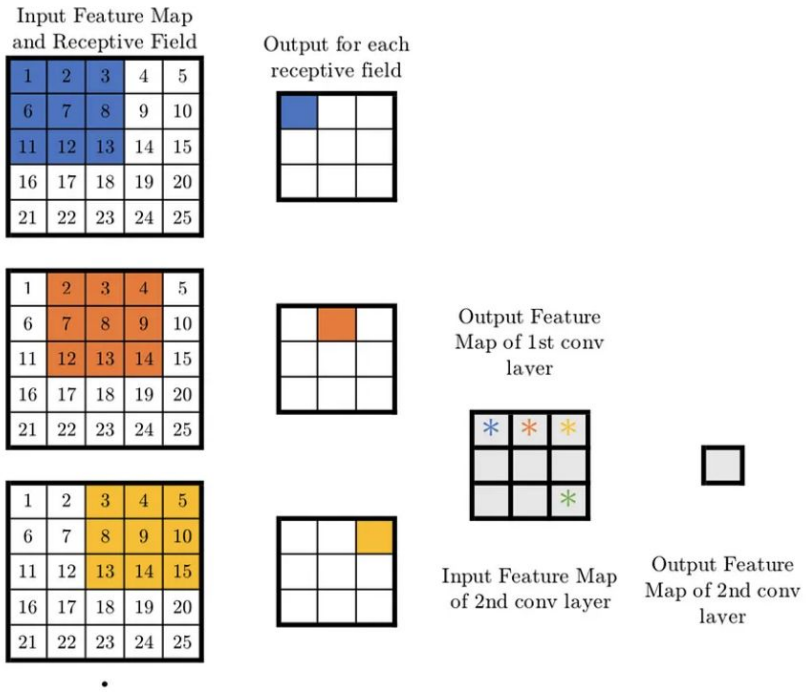
- jedno z pierwszych udanych zastosowań sieci konwolucyjnych
- używa ReLU + *dropout* + nie wszystkie warstwy splotowe są przedzielone warstwami łączącymi
- osiągnęło *top-5 error* równy 15.3% na ImageNet w 2012, o 10.8 punktów procentowych lepiej niż rozwiązanie z drugiego miejsca
- 62M parametrów w 8 uczących się warstwach



https://en.wikipedia.org/wiki/LeNet

VGG: wiele filtrów 3x3 symuluje filtry 5x5 i 7x7

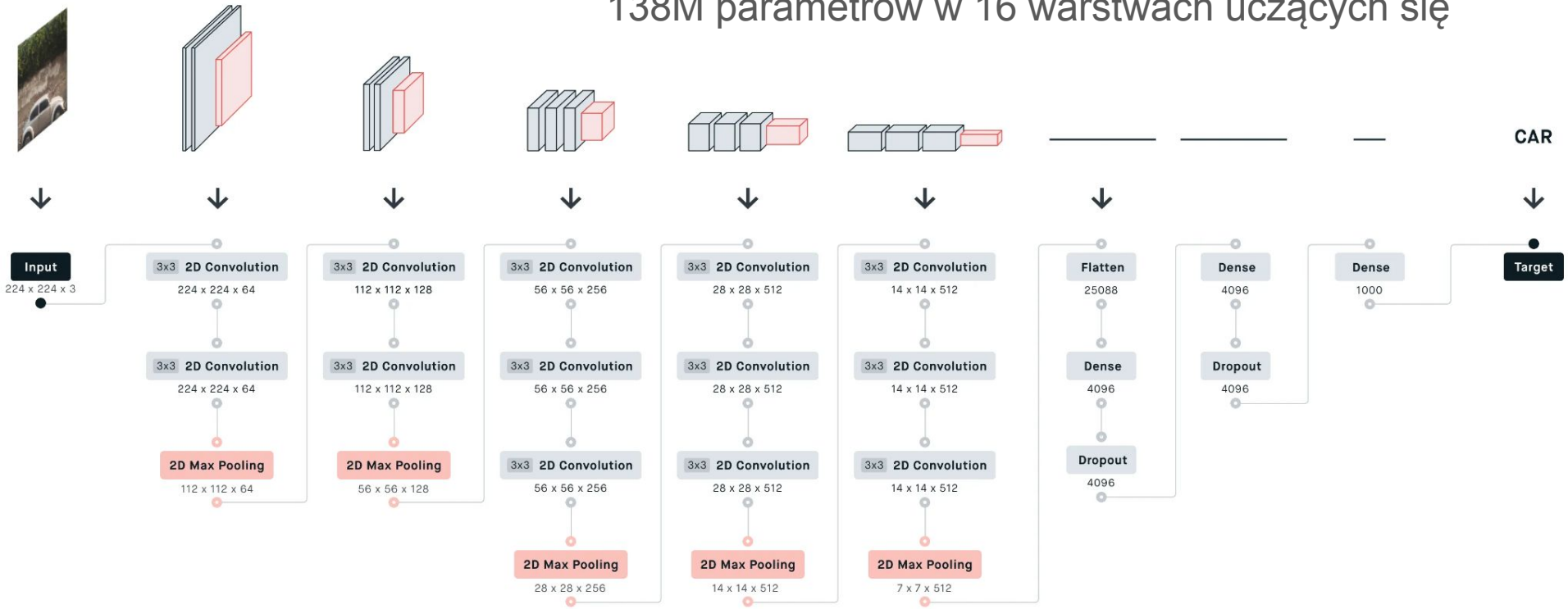
- VGG, opublikowane w 2014, poprawiło wynik AlexNet z 15.3% do 7.7% *top-5 error*
- dwie następujące po sobie warstwy splotowe o filtrach 3x3 mają to samo pole widzenia co pojedynczy filtr 5x5, zawierając przy tym mniej parametrów
- pozwala to zachować te same możliwości modelujące, przy niższej tendencji do *overfitting*



https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96

Architektura VGG16

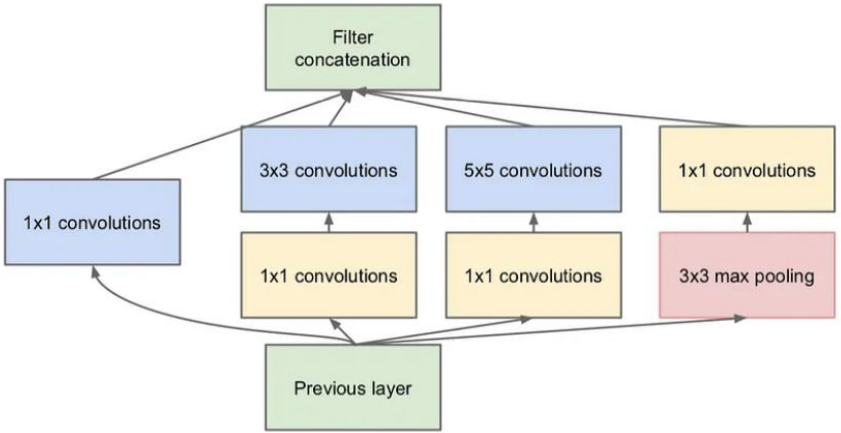
138M parametrów w 16 warstwach uczących się



https://medium.com/@siddheshb008/vgg-net-architecture-explained-71179310050f

Moduł Inception

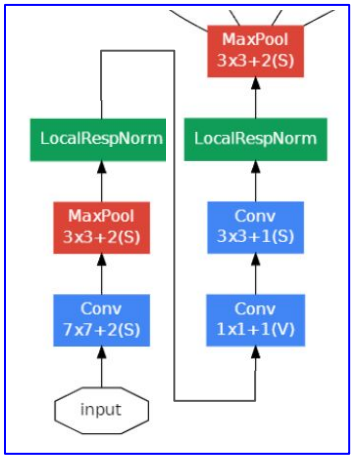
- warstwa, która jednocześnie analizuje różne skale na obrazie
- zawiera filtry różnych rozmiarów
- wyniki działania filtrów są sklejane w jedno wyjście, w którym kanały pochodzą z poszczególnych filtrów
- filtry 1x1 z liczbą kanałów na wyjściu mniejszą niż na wejściu zmniejszają liczbę parametrów uczących się
- po publikacji VGG filtr 5x5 został zamieniony na dwa filtry 3x3

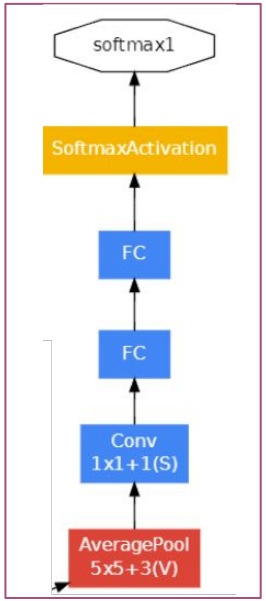


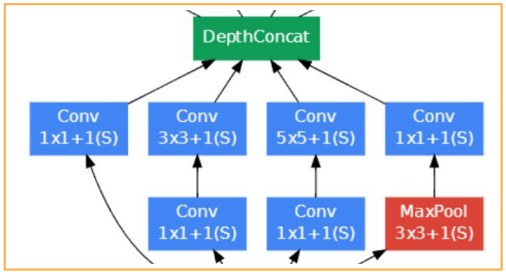
Going Deeper with Convolutions, Szegedy et al. (2014)

GoogleNet

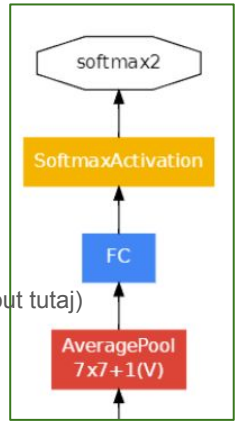
jedynie 6.4M parametrów
w 10+9x6 warstwach
uczących się. 6.4% *top-5 error*

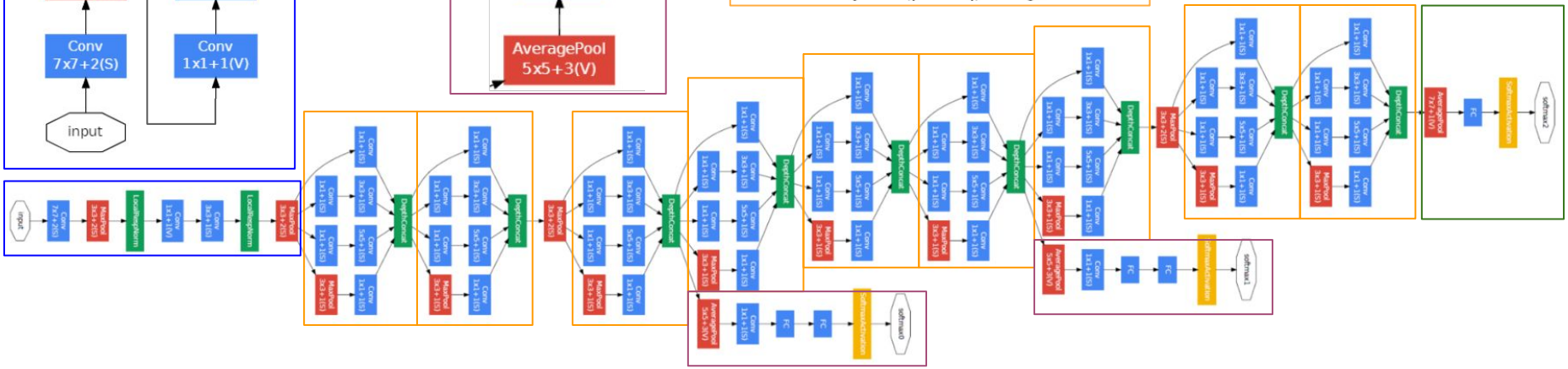






(dropout tutaj)

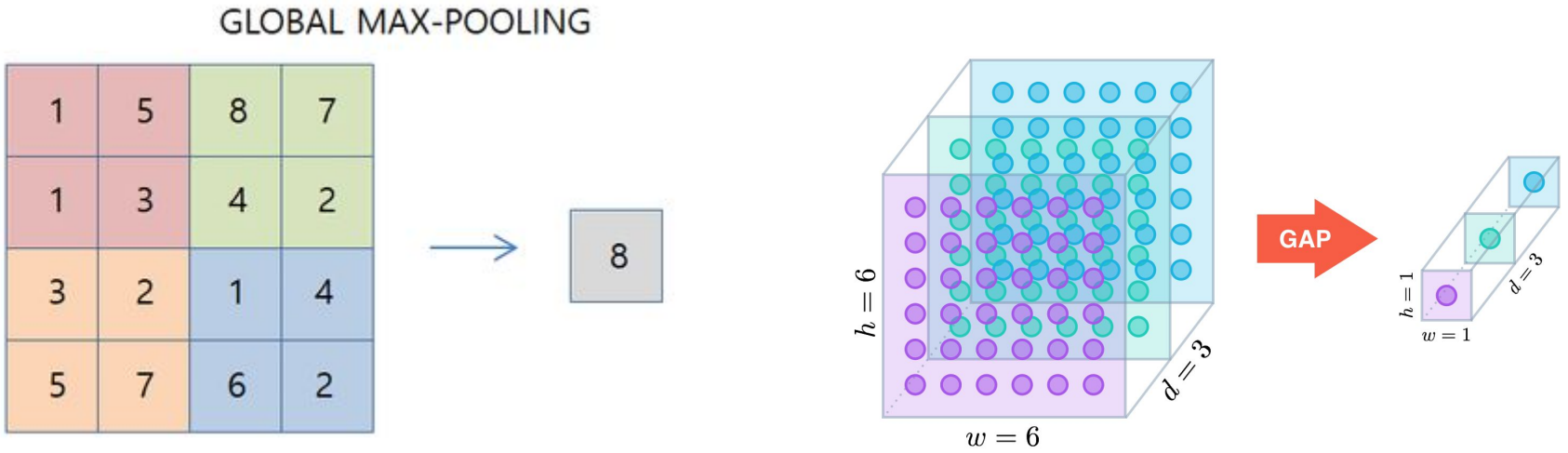




Going Deeper with Convolutions, Szegedy et al. (2014)

Global pooling

- warstwa łącząca, która redukuje każdy kanał do pojedynczej wartości
- używana przed warstwami FC podczas zamiany reprezentacji z 3D na 1D
- użyta w GoogleNet poprawiła top-1 accuracy o 0.6 punktów procentowych

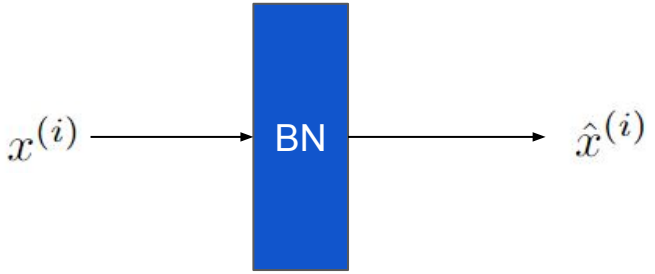


https://www.researchgate.net/figure/The-difference-of-max-pooling-and-global-max-pooling_fig4_338079465

https://www.kaggle.com/discussions/general/175896

Batch Normalization

- choć normalizacja wejścia i inicjalizacja gwarantują, że na początku uczenia wejścia wszystkich warstw mają rozkład normalny, to wraz z uczeniem rozkład ten może się zmieniać (*covariance shift*)
- *batch normalization* normalizuje wejścia używając statystyk z *mini-batch*'a
- podczas testowania używane są statystyki ze zbioru treningowego
- dodatkowe uczące się przekształcenie liniowe ma niewielkie znaczenie praktyczne

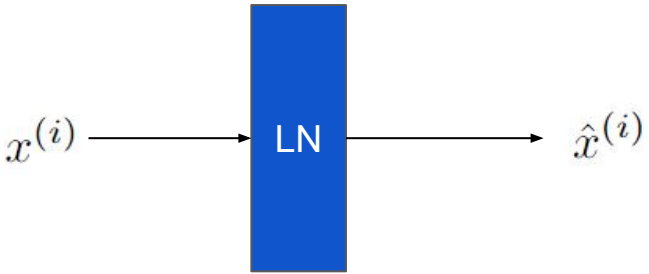


$$\hat{x}^{(i)} = \gamma \frac{x^{(i)} - \mu_B}{\sigma_B} + \beta$$

	cena kawy	dystans od centrum
Charlotte	~~15~~ 1	~~1000~~ 0.84
KFC	~~10~~ -1	~~200~~ -1.40
W Orbicie Słońca	~~12.5~~ 0	~~900~~ 0.56

Layer Normalization

- *batch normalization* wprowadza różnicę pomiędzy uczeniem a *inference* i nie działa, gdy batch jest bardzo mały
- *layer normalization* normalizuje współrzędne poszczególnych elementów zamiast normalizować elementy poszczególnych *batch*'y
- używane w Transformerach i RNN

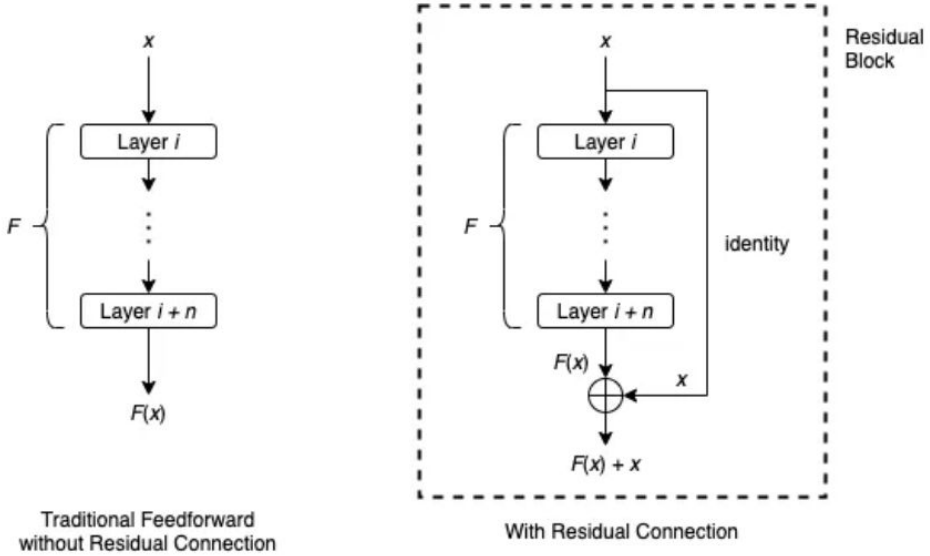


$$\hat{x}_j^{(i)} = \gamma \frac{x_j^{(i)} - \mu_{x^{(i)}}}{\sigma_{x^{(i)}}} + \beta$$

	verbs	adjectives	nouns
Pride and Prejudice	~~30k~~ 0.7	~~30k~~ 0.7	~~25k~~ -1.41
Macbeth	~~2k~~ 0.18	~~400~~ -1.3	~~3k~~ 1.12

Residual connections (in. skip connections)

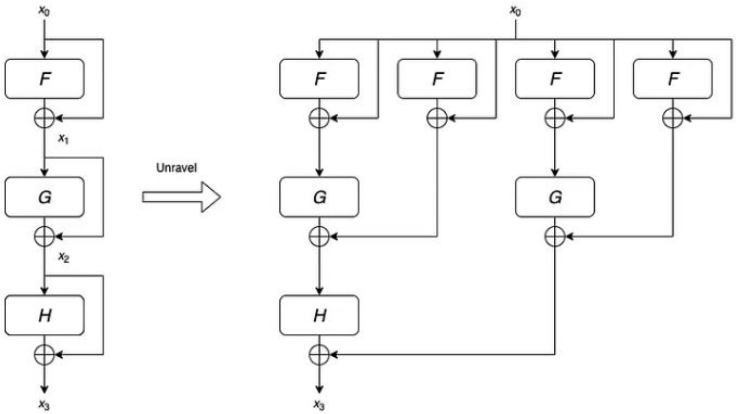
- zaobserwowano, że warstwy w sieciach z trudem uczą się funkcji f(x)=x, co sugeruje, że mogą mieć problemy z przekazywaniem informacji z poprzednich warstw
- *skip connection* dodaje wejście warstwy do jej wyjścia; f(x)+x jest modelowane zamiast f(x)
- jeśli wymiary f(x) i x są różne, na x stosowane jest proste przekształcenie (np. filtr 1x1)

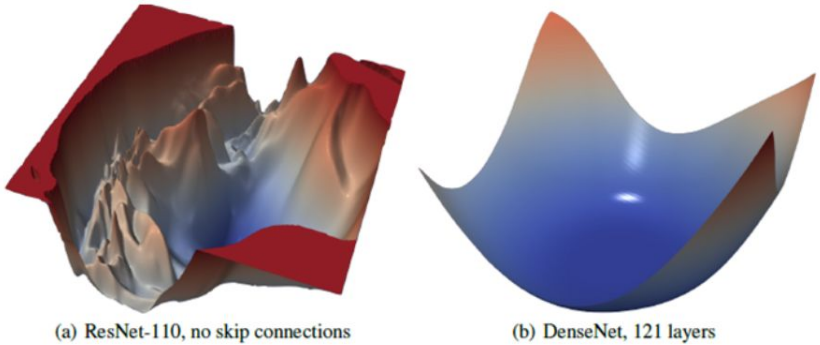


https://towardsdatascience.com/what-is-residual-connection-efb07cab0d55

Skip connections propagują wyniki poprzednich warstw

- interpretacja: *skip connections* zapobiegają utracie informacji; każda warstwa ma dostęp do wyjść wszystkich warstw poprzednich (co prawda dodanych do siebie, ale ważne cechy zaczną dominować w miarę uczenia)
- raz otrzymane cechy są poddawane mniejszej liczbie przekształceń, co łagodzi funkcję straty





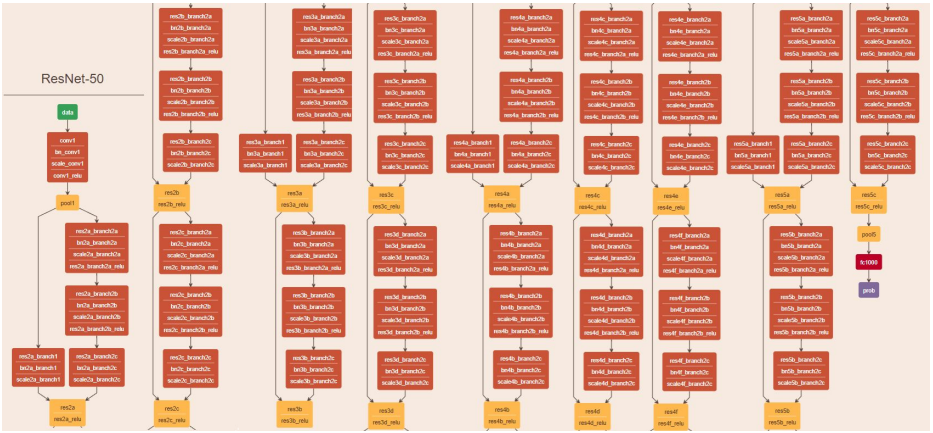
(a) ResNet-110, no skip connections (b) DenseNet, 121 layers

Figure 4: The loss surfaces of ResNet-110-noshort and DenseNet for CIFAR-10.

Visualizing the Loss Landscape of Neural Nets, Li et al. 2017
https://towardsdatascience.com/what-is-residual-connection-efb07cab0d55

ResNet

- następca VGG i GoogleNet opublikowany w 2015
- używa *Batch Normalization* i *residual connections*
- Resnet-152, pomimo bardzo wielu warstw, zawiera mniej parametrów niż AlexNet
- *pooling* przestaje być potrzebny
- pełen wykres architektury pod linkiem poniżej



Comparison					
Network	Year	Salient Feature	top5 accuracy	Parameters	FLOP
AlexNet	2012	Deeper	84.70%	62M	1.5B
VGGNet	2014	Fixed-size kernels	92.30%	138M	19.6B
Inception	2014	Wider - Parallel kernels	93.30%	6.4M	2B
ResNet-152	2015	Shortcut connections	95.51%	60.3M	11B

https://ethereon.github.io/netscope/#/gist/db945b393d40bfa26006

ImageNet jako benchmark

- współczesne sieci klasyfikują ImageNet lepiej niż ludzie
- brak trudniejszego zbioru danych do klasyfikacji o podobnej skali
- bardziej popularne: generacja obrazów, semi-supervised learning
- klasyfikatory pretrenowane na ImageNet są nadal stosowane

Are we done with ImageNet?

Lucas Beyer[1*] Olivier J. Hénaff[2*] Alexander Kolesnikov[1*] Xiaohua Zhai[1*] Aäron van den Oord[2*]
[1]Google Brain (Zürich, CH) and [2]DeepMind (London, UK)

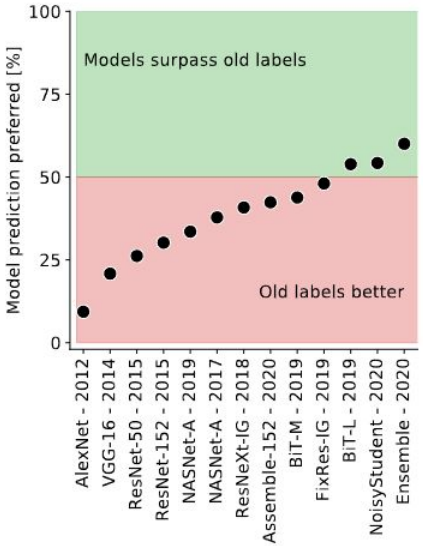


Figure 1: When presented with a model's prediction and the original ImageNet label, human annotators now prefer model predictions on average (Section 4). Nevertheless, there remains considerable progress to be made before fully capturing human preferences.

- ImageNet jest zbyt mały, aby ViT dawał lepsze wyniki

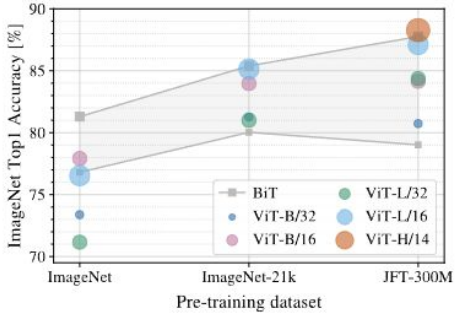


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

Are we done with ImageNet? Beyer et al. 2020
An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, Dosovitskiy et al. 2020

Adam

- najbardziej popularna metoda optymalizacyjna oprócz SGD
- “normalizuje” gradient w różnych krokach dzieląc każdą współrzędną przez pierwiastek jej oczekiwanego kwadratu (in. pierwiastek jej drugiego momentu zwykłego)
- drugi moment zwykły jest szacowany przez średnią kroczącą
- krok w dowolnym kierunku z bazy kanonicznej < learning rate

$$
\begin{aligned}
m_i &= \beta_1 m_{i-1} + (1-\beta_1)\nabla\mathcal{L}(w_i) \\
s_i &= \beta_2 s_{i-1} + (1-\beta_2)(\nabla\mathcal{L}(w_i))^2 \\
\hat{m}_i &= \frac{m_i}{1-\beta_1^i} \\
\hat{s}_i &= \frac{s_i}{1-\beta_2^i} \\
v_i &= \frac{-\eta\hat{m}_i}{\sqrt{\hat{s}_i}+\epsilon 1}
\end{aligned}
$$

- działa lepiej niż SGD gdy gradienty parametrów mają drastycznie różną skalę, np. w Transformerach

Weight decay

- zaobserwowano, że *overfitting* często towarzyszą bardzo duże pojedyncze wagi
- *weight decay* zmniejsza wszystkie wagi po każdej epoce/iteracji
- wbrew popularnemu mitowi, *weight decay* jest równoznaczny z regularyzacją L2 funkcji straty **tylko** w przypadku SGD
- aby zapobiec błędom, poprawny sposób na użycie weight decay z Adam'em został nazwany AdamW

SGD z momentum i WD:

$$v_i = \beta v_{i-1} - \eta \nabla \mathcal{L}(w_i) - \eta \lambda w_{i-1}$$

AdamW:

$$w_i = w_{i-1} - \frac{\eta \hat{m}_i}{\sqrt{\hat{s}_i} + \epsilon 1} - \eta \lambda w_{i-1}$$