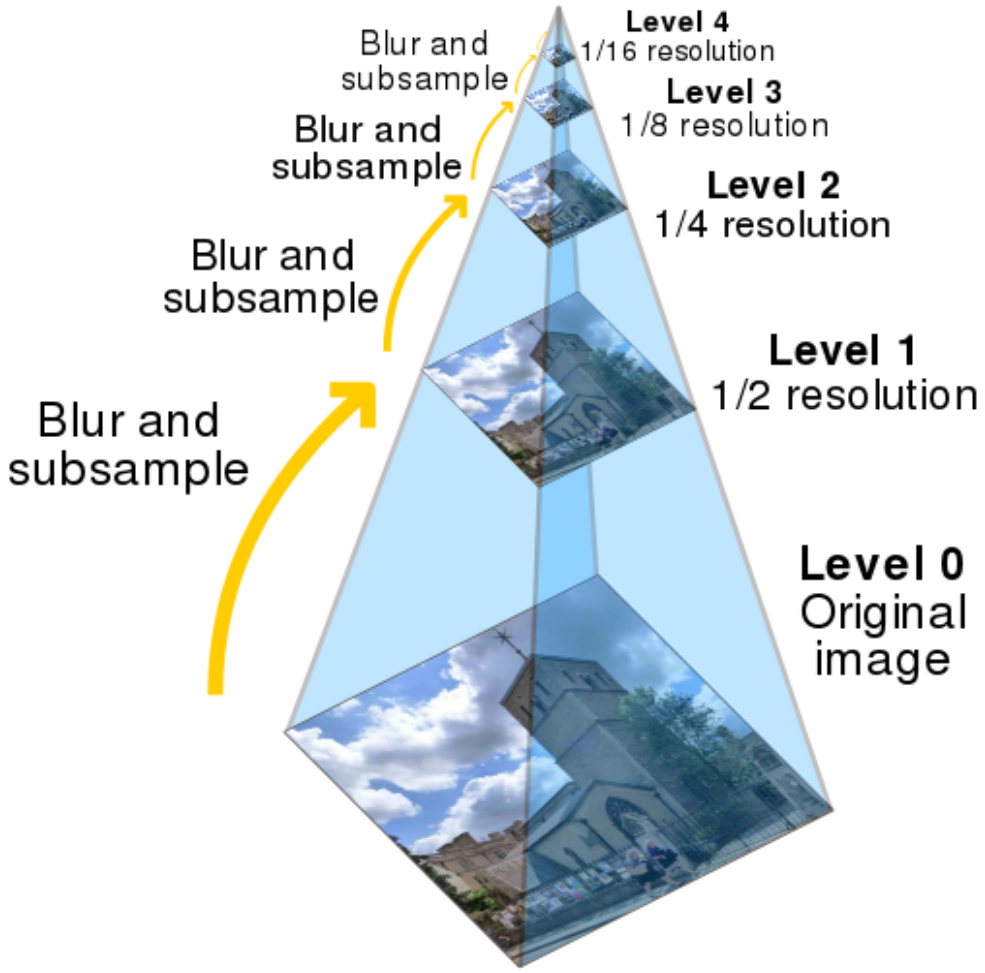
Metody Inżynierii Wiedzy
Sieci konwolucyjne - wykład 8

Adam Szmigielski
aszmigie@pjwstk.edu.pl
materiały: $ftp(public) : //aszmigie/MIW$

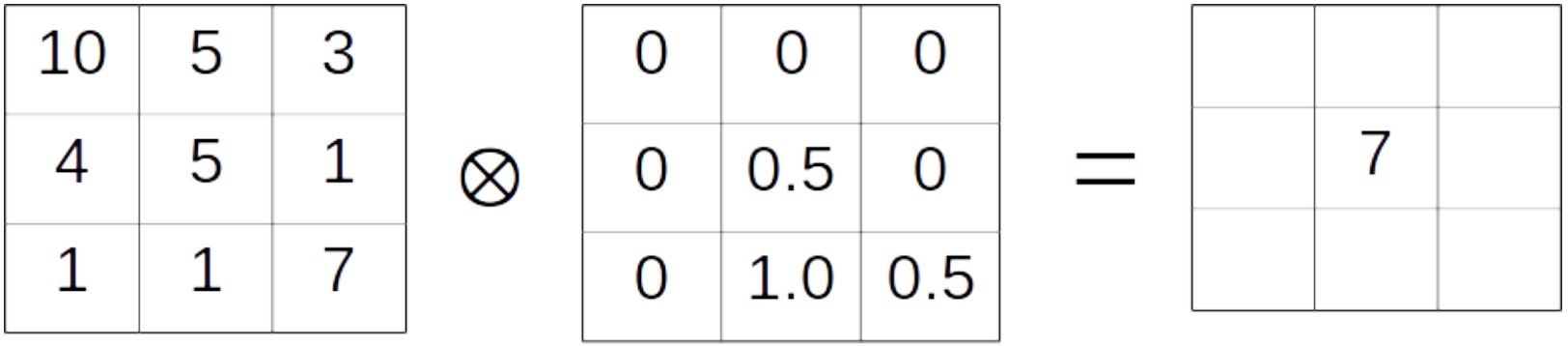
Algorytm piramid



Algorytm piramid - niezmienniki przekształceń

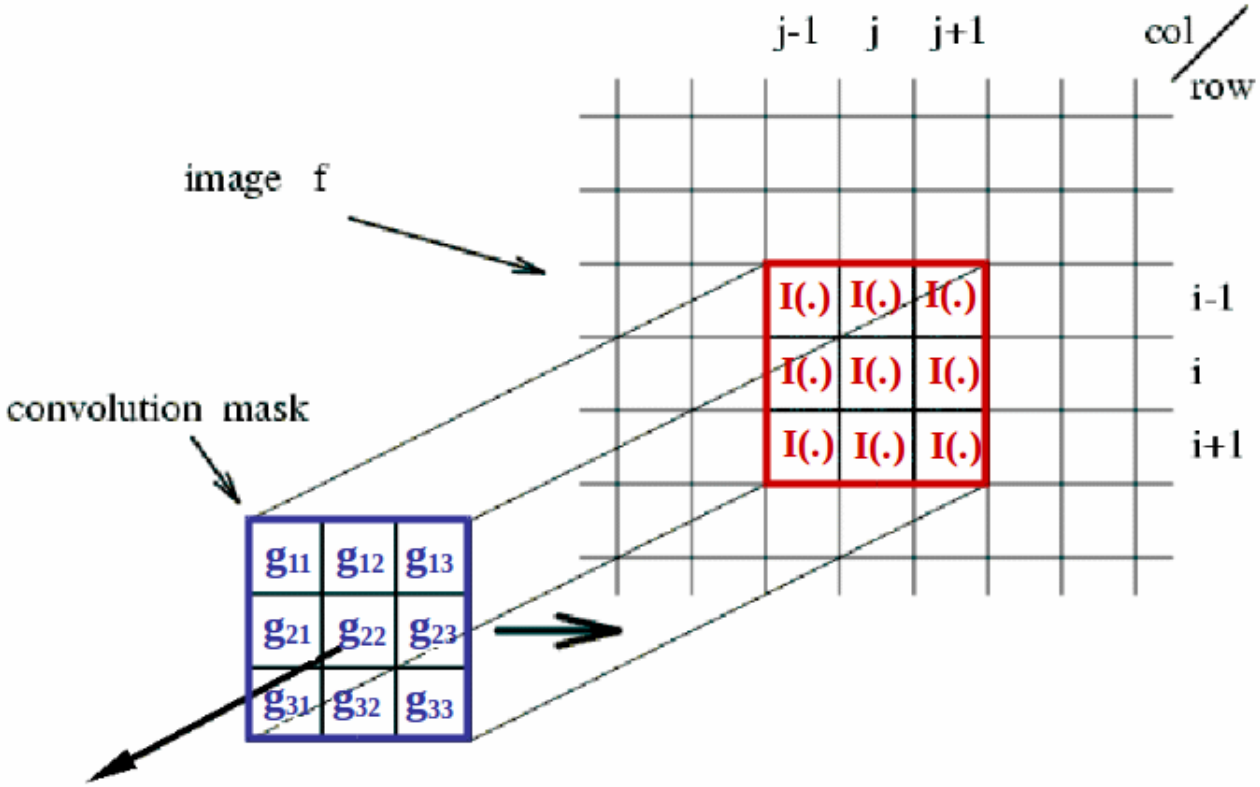
- Algorytm ten ma własność zachowywania niezmienników przekształceń liniowych - translacje, obrót, skalowanie.
- Własność ta daje możliwość opisu wzorcu jako zbioru niemienników przekształceń oraz jest powodem powstania nowego rodzaju sieci - sieci konwolucyjnych CNN

Filtracja liniowa



- liniowa jest najprostsza i najbardziej użyteczna
- Zamienia każdy piksel na liniową kombinację sąsiadów.
- Funkcję (sposób) na kombinację liniową nazywa się jądrem splotu (ang. convolution kernel).

Filtr liniowy - splot

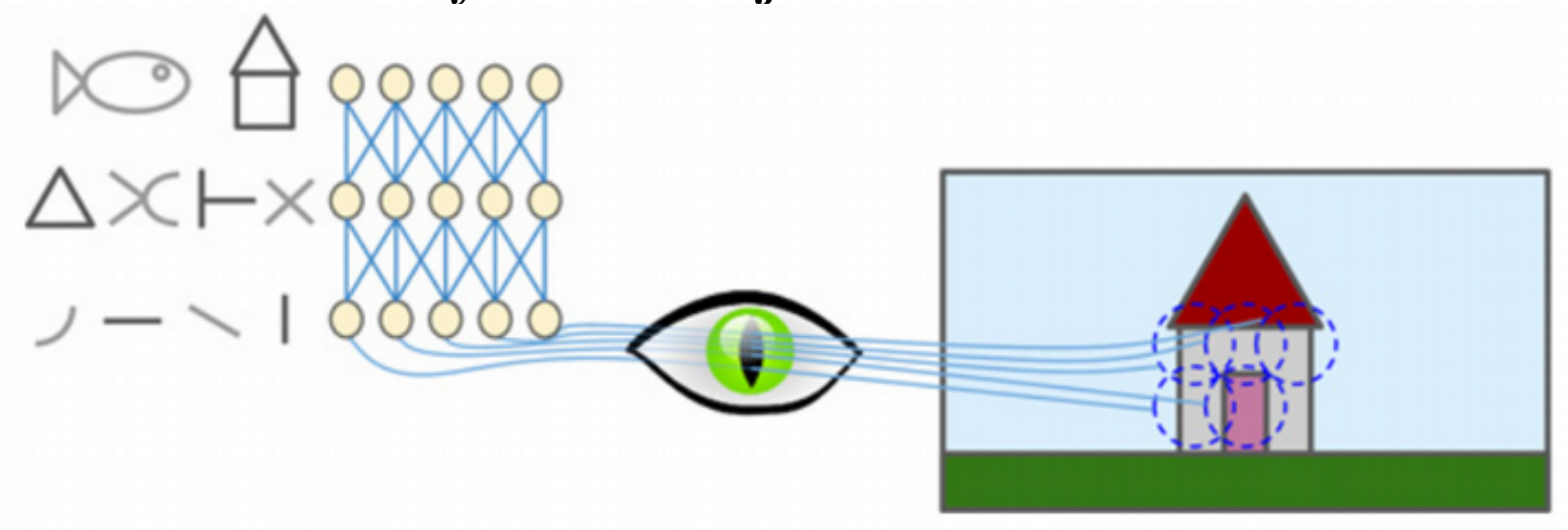


$$f(i,j) = g_{11}\, I(i-1,j-1) + g_{12}\, I(i-1,j) + g_{13}\, I(i-1,j+1) + g_{21}\, I(i,j-1) + g_{22}\, I(i,j) + g_{23}\, I(i,j+1) + g_{31}\, I(i+1,j-1) + g_{32}\, I(i+1,j) + g_{33}\, I(i+1,j+1)$$

Splotowe (konwolucyjne) sieci neuronowe

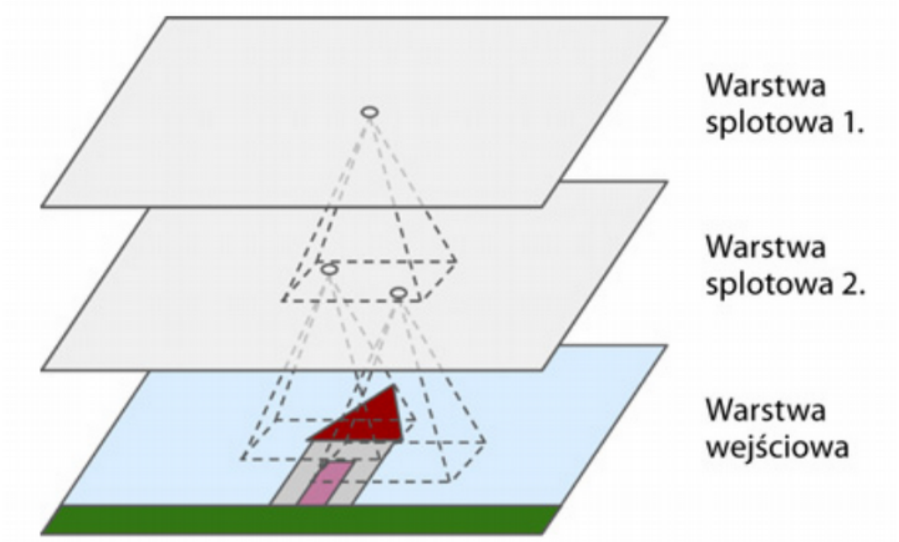
- Stanowią wynik badań nad korą wzrokową i od lat 80. ubiegłego wieku są używane do rozpoznawania obrazów.
- Stanowią one podstawę usług wyszukiwania obrazów, samochodów inteligentnych, zautomatyzowanych systemów klasyfikowania filmów itd.
- Są się również skuteczne w innych zadaniach, takich jak rozpoznawanie mowy (ang. voice recognition) czy przetwarzanie języka naturalnego.

Architektura kory wzrokowej



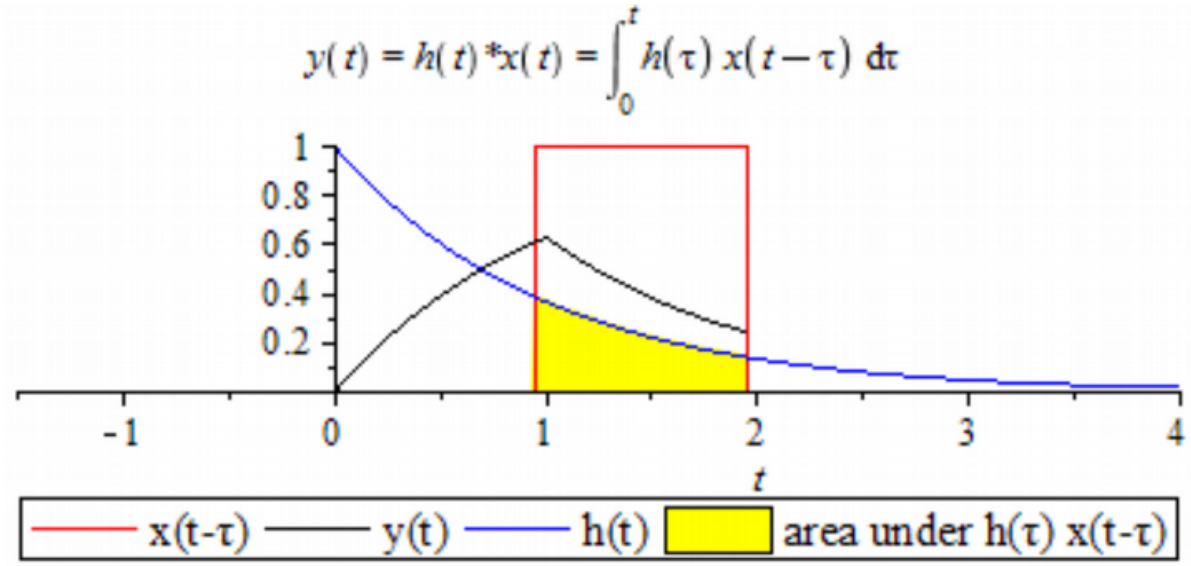
- Neurony tworzą **lokalne pola recepcyjne** - reagują jedynie na bodźce wzrokowe mieszczące się w określonym rejonie,
- Niektóre neurony mogą mieć to samo pole recepcyjne, ale reagują wyłącznie na obrazy składające się z linii poziomych, a inne pionowe,
- Neurony wykrywające bardziej skomplikowane kształty (stanowiące połączenie bardziej ogólnych wzorów) znajdują się na wyjściu sąsiadujących neuronów.

Warstwa splotowa



- Neurony w pierwszej warstwie splotowej nie są połączone z każdym pikselem obrazu wejściowego, lecz wyłącznie z pikselami znajdującymi się w ich polu recepcyjnym
- Z kolei każdy neuron w drugiej warstwie splotowej łączy się wyłącznie z neuronami znajdującymi się w niewielkim obszarze pierwszej warstwy.

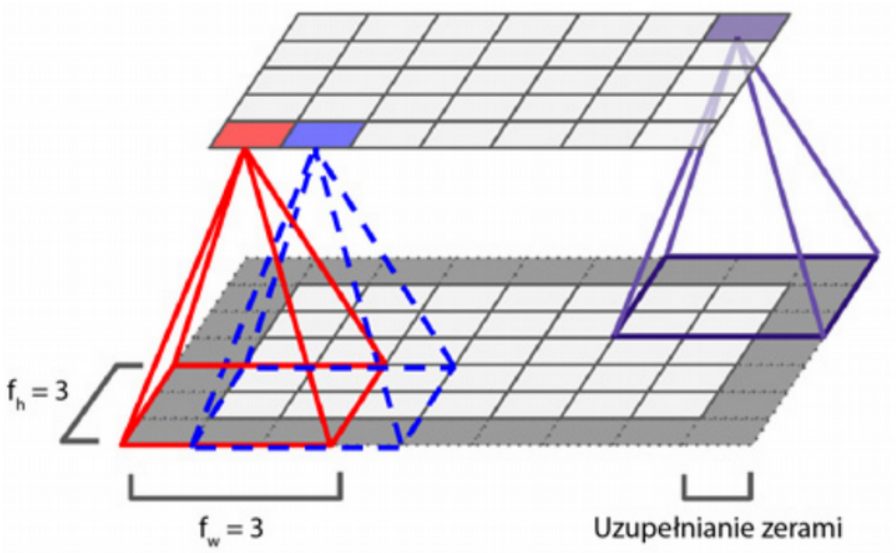
Splot



Splot (konwolucja) jest operacją matematyczną pomiędzy dwiema funkcjami, mierzącą całkę z ich iloczynu punktowego,

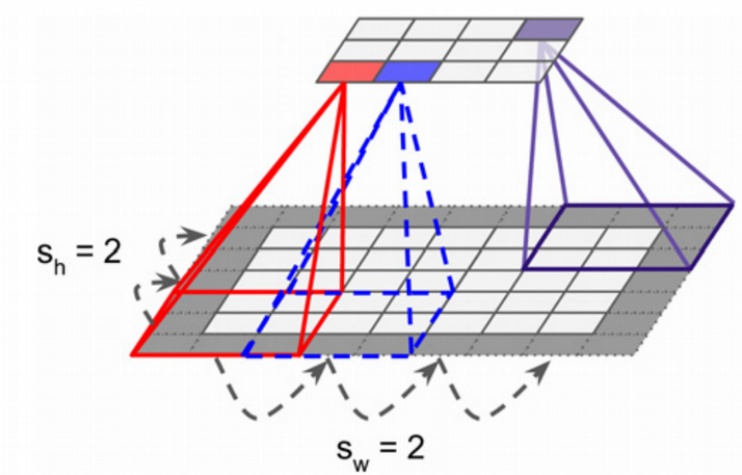
- Jest on ściśle powiązany z transformacjami Fouriera i Laplace'a oraz powszechnie używany w przetwarzaniu sygnałów,
- W sieciach CNN stosowane są operacje korelacji krzyżowej, bardzo przypominające konwolucję.

Splot 2D



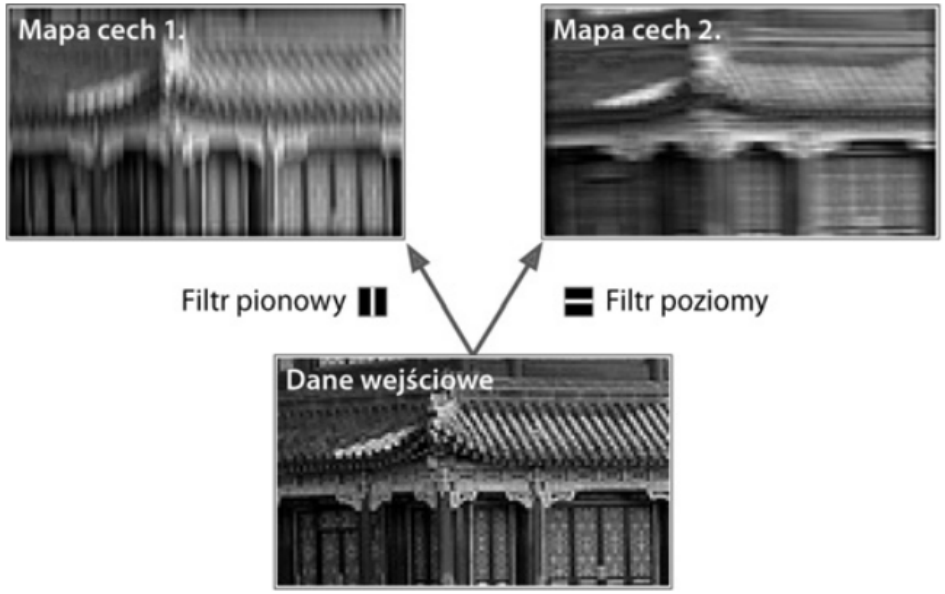
- Neuron znajdujący się w wierszu i oraz kolumnie j danej warstwy jest połączony z wyjściami neuronów poprzedniej warstwy zlokalizowanymi w rzędach od i do $i + f_h - 1$ i kolumnach od j do $j + f_w - 1$, gdzie f_h i f_w oznaczają, odpowiednio, wysokość i szerokość pola recepcyjnego
- W celu uzyskania takich samych wymiarów każdej warstwy najczęściej są dodawane zera wokół wejść - uzupełnianiem zerami (ang. zero padding).

Rozdzielanie pól recepcyjnych



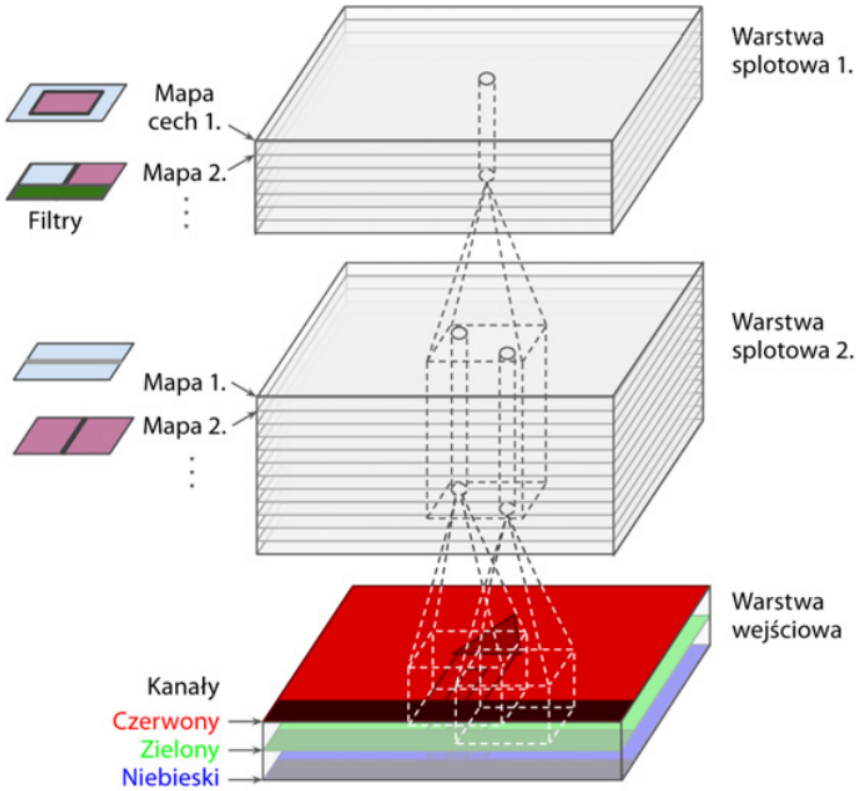
- Odległość pomiędzy dwoma kolejnymi polami jest **krokiem**
- Warstwa wejściowa o wymiarach 5×7 (plus uzupełnianie zerami) łączy się z warstwą o rozmiarze 3×4 za pomocą pól recepcyjnych będących kwadratami 3×3 i kroku o wartości 2
- Neuron zlokalizowany w rzędzie i oraz kolumnie j górnej warstwy łączy się z wyjściami neuronów dolnej warstwy mieszczącymi się w rzędach od $i \times s_h$ do $i \times s_h + f_h - 1$ i w kolumnach od $j \times s_w$ do $j \times s_w + f_w - 1$, gdzie s_h i s_w wartości kroków w kolumnach i rzędach.

Filtry



- Wagi neuronu mogą być przedstawiane jako niewielki obraz o rozmiarze pola recepcyjnego,
- Pierwszy filtr (czarny kwadrat z białą pionową linią) przechodzącą przez jego środek (jest to macierz o wymiarach 7×7 wypełniona zerami oprócz środkowej kolumny, która zawiera jedynki). Drugi filtr (środkowa linia jest ułożona poziomo),
- Warstwa wypełniona neuronami wykorzystującymi ten sam filtr daje nam mapę cech (ang. feature map),

Stosy map cech



- Dla jednej mapy cech wszystkie neurony mają te same parametry,
- Inne mapy cech mogą mieć inne wartości parametrów,
- Warstwa splotowa równocześnie stosuje różne filtry na wejściach, dzięki czemu jest w stanie wykrywać wiele cech.

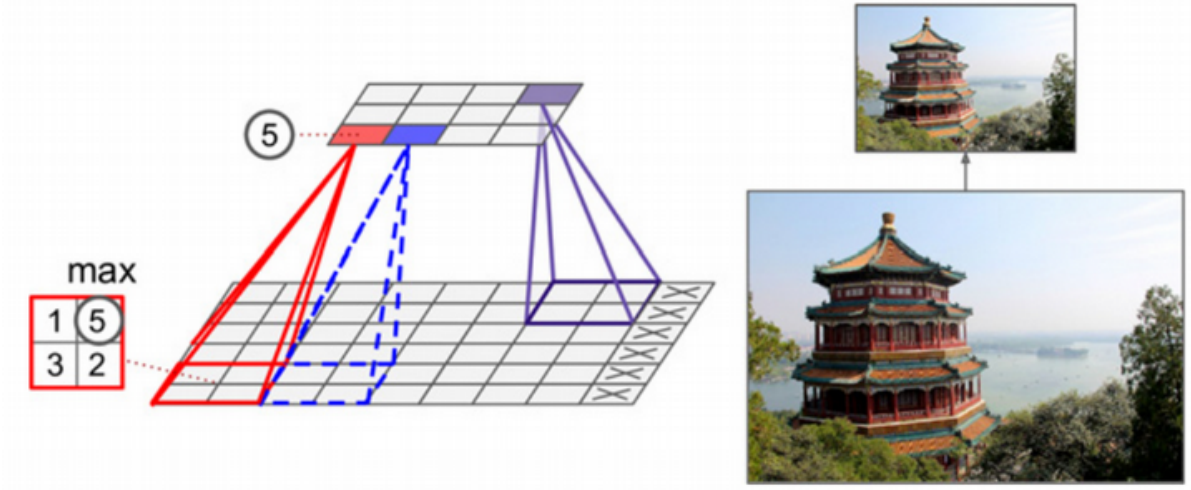
Wartości wyjściowej neuronu w warstwie splotowej

$$z_{i,j,k} = b_k \sum_{u=0}^{f_h-1} \sum_{v=0}^{f_w-1} \sum_{k'=0}^{f_{n'}-1} x_{i',j',k'} \cdot w_{u,v,k',k} \tag{1}$$

gdzie $i' = i \times s_h + u$ oraz $j' = j \times s_w + v$

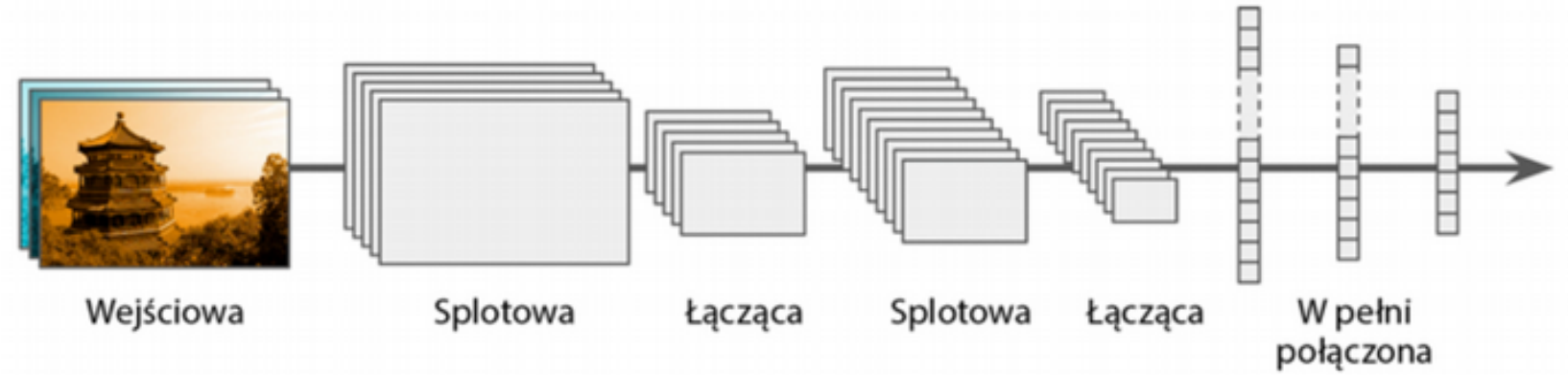
- $z_{i,j,k}$ - wyjście neuronu w rzędzie i, kolumnie j i mapie cech k warstwy splotowej l,
- s_h i s_w - kroki pionowy i poziomy,
- f_h i f_w - wysokość i szerokość pola recepcyjnego,
- f_n - liczba map cech w poprzedniej warstwie $(l-1)$,
- $x_{i,j,k}$ - wyjście neuronu w $l-1$, rzędzie i , kolumnie j , mapie cech k
- b_k - bias dla mapy cech k (w warstwie l)
- $w_{u,v,k',k}$ - waga w mapie cech k warstwy l a jego wejściem mieszczącym się w wierszu u, kolumnie v a mapą cech k

Warstwa łącząca



- Jądro łączące: 2×2, brak uzupełniania zerami
- Nie zawiera żadnych wag - gromadzi dane wejściowe za pomocą funkcji agregacyjnej (np. maksymalizującej lub uśredniającej),
- Celem warstwy splotowej jest podpróbkowanie (ang. *subsample*) obrazu wejściowego w celach optymalizacyjnym,
- Neuron z warstwy łączącej łączy się z wyjściami określonej liczby neuronów warstwy poprzedniej, w obszarze pola recepcyjnego.

Architektury splotowych sieci neuronowych



- Typowe architektury CNN składają z kilku warstw splotowych, warstwy łączącej, po niej znowu kilku warstw splotowych (+ReLU), kolejnej warstwy łączącej itd.
- Obraz stopniowo maleje, przechodząc przez kolejne warstwy sieci, ale jednocześnie zwiększa się jego głębia
- Na szczycie sieci zostaje umieszczona klasyczna sieć neuronowa, zawierająca kilka w pełni połączonych warstw (+ReLU), a ostatnia z nich wylicza prognozy (np. regresja logistyczna softmax).

```
# Przyklad sieci spolotowej w keras
from keras import layers
from keras import models
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10, activation='softmax'))
model.summary()
```

- Sieć przyjmuje tensor o kształcie określanym przez wysokość obrazu, jego szerokość i kanały obrazu np. (28, 28, 1)
- Na wyjściu każdej warstwy *Conv2D* i *MaxPooling2D* pojawia się trójwymiarowy tensor o kształcie (wysokość, szerokość, kanały) .
- Wraz z zagłębianiem się sieć wysokość i szerokość mają tendencję do przyjmowania mniejszych wartości.
- Ostatnim krokiem jest sieć gęsta z klasyfikatorem (tensor (3, 3, 64)).

Wynik tworzenia modelu

```
Instructions for updating:
Colocations handled automatically by placer.
```

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d_1 (MaxPooling2	(None, 13, 13, 32)	0
conv2d_2 (Conv2D)	(None, 11, 11, 64)	18496
max_pooling2d_2 (MaxPooling2	(None, 5, 5, 64)	0
conv2d_3 (Conv2D)	(None, 3, 3, 64)	36928
flatten_1 (Flatten)	(None, 576)	0
dense_1 (Dense)	(None, 64)	36928
dense_2 (Dense)	(None, 10)	650

```
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
```

Uruchomienie i trenowanie modelu

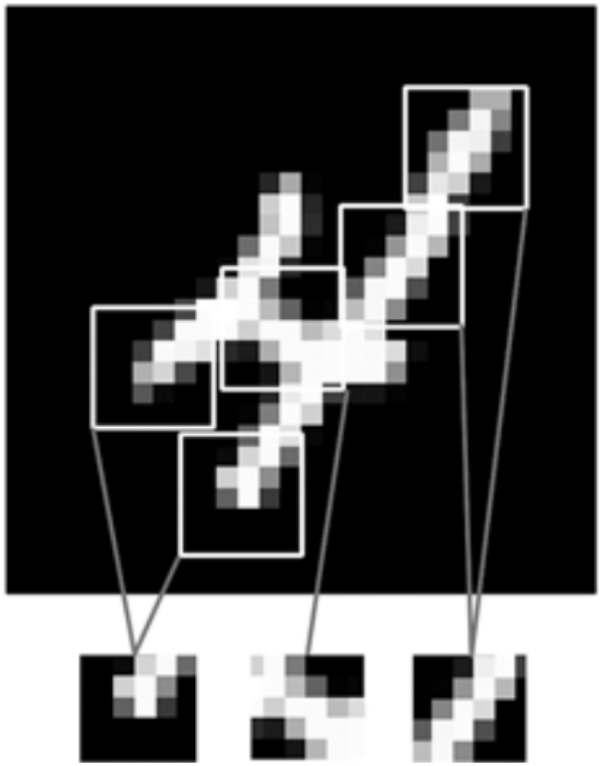
```
from keras.datasets import mnist
from keras.utils import to_categorical
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
train_images = train_images.reshape((60000, 28, 28, 1))
train_images = train_images.astype('float32') / 255
test_images = test_images.reshape((10000, 28, 28, 1))
test_images = test_images.astype('float32') / 255
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)
model.compile(optimizer='rmsprop', loss='categorical_crossentropy',
metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5, batch_size=64)

test_loss, test_acc = model.evaluate(test_images, test_labels)
print('test_acc =', test_acc)
```

Wynik działania programu:

```
.................................
10000/10000 [==============================] - 1s 72us/step
test_acc =  0.9919
```

Działanie sieci konwolucyjnej

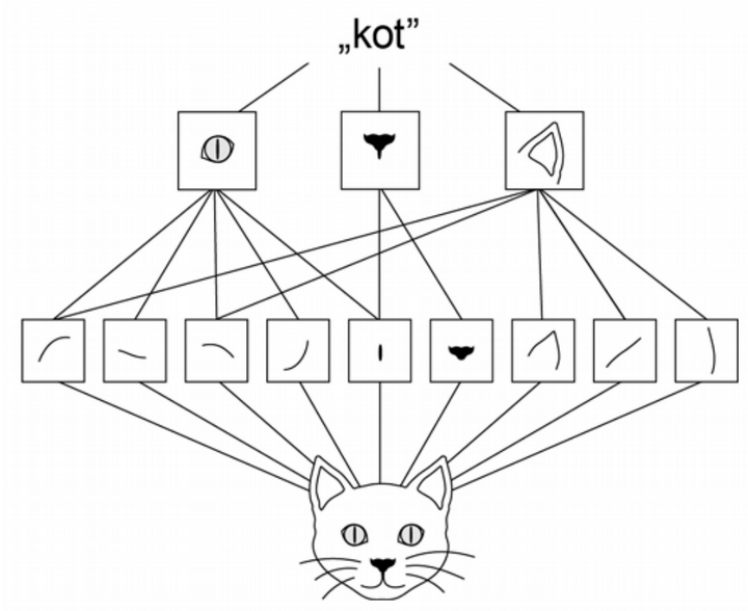


- Obrazy mogą zostać rozbite na lokalne wzorce, np. krawędzie i tekstury,
- Podstawową różnicą między warstwą gęstych połączeń a siecią konwolucyjną jest to, że warstwy Dense (gęste) uczą się cech parametrów globalnych w swoich wejściowych przestrzeniach,

Własności sieci konwolucyjnych

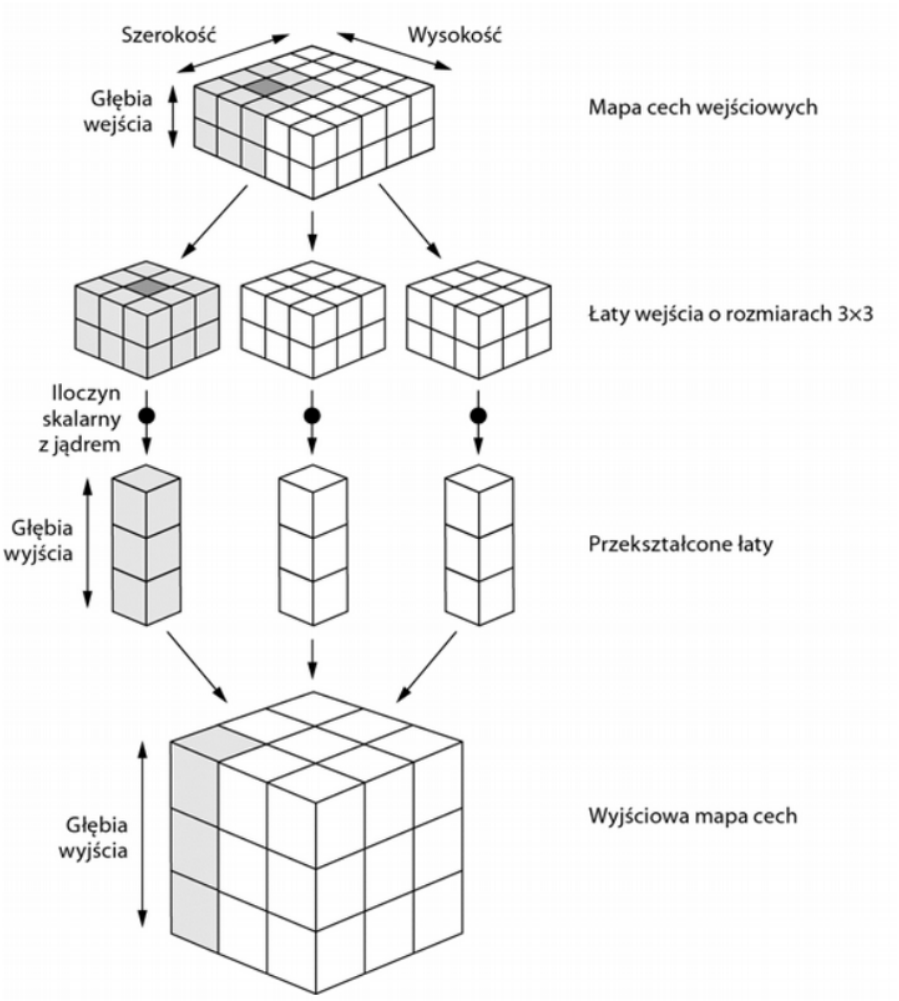
- **Wzorce rozpoznawane przez sieć są niezależne od przesunięcia.** - algorytm przetwarzający obrazy nie może wiązać kształtu z jego położeniem w kadrze.
- **Sieci konwolucyjne mogą uczyć się przestrzennej hierarchii wzorców**. Pierwsza warstwa sieci konwolucyjnej uczy się małych lokalnych wzorców (np. krawędzie), druga warstwa tej sieci będzie uczyła się większych struktur składających się z elementów rozpoznanych przez pierwszą warstwę itd.

Przestrzenna hierarchia elementów



- Działają na trójwymiarowych tensorach określanych mianem map cech, zawierających dwie przestrzenne osie definiujące wysokość i szerokość.
- Trzecią osią jest oś głębi, nazywana również osią kanałów.
- Dokonują ekstrakcji łat z cech wejściowych i przeprowadzają tę samą operację na wszystkich łatach w celu utworzenia wyjściowej mapy cech.

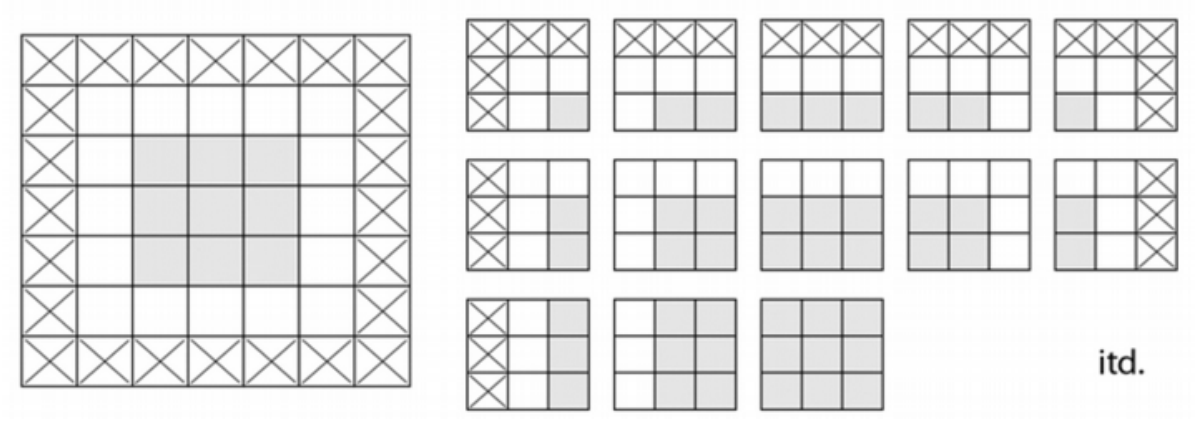
Działanie konwolucji



Działanie konwolucji

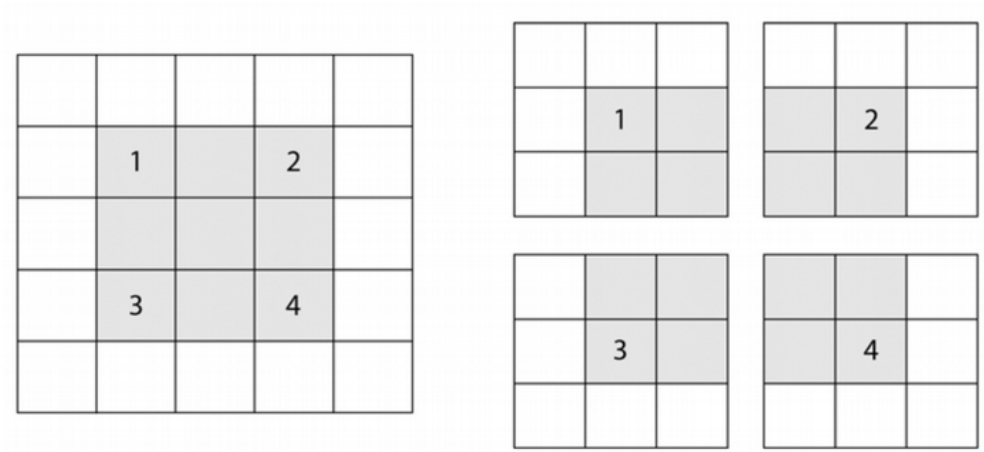
- Polega na przesuwania okien o wymiarach 3×3 lub 5×5 po trójwymiarowej mapie cech.
- Dokonywana jest ekstrakcja trójwymiarowej łaty otaczających je cech *(wysokość okna, szerokość okna, głębia wejścia)*).
- Każda taka trójwymiarowa łata jest następnie przekształcana (poprzez iloczyn tensorowy z jądrem konwolucji) w celu uzyskania jednowymiarowego wektora o kształcie *(głębia wyjścia)*.
- Wszystkie te wektory są następnie przestrzennie przebudowywane w celu utworzenia trójwymiarowej mapy wyjściowej o kształcie *(wysokość, szerokość, głębia wyjścia)*.

Efekty graniczne i dopełnienie



- Przykładowa mapą cech o wymiarach 5×5 - zawiera ona tylko 9 pól, na których można ustawić środek okna o wymiarach 3×3
- Mapa cech będzie miała wymiary 3×3.
- Gdybyśmy chcieli uzyskać wyjściową mapę cech o takich samych wymiarach jak mapa wejściowa, to musielibyśmy skorzystać z techniki dopełniania.
- Polega ona na dodawaniu odpowiedniej liczby wierszy i kolumn po każdej stronie wejściowej mapy cech

Kroki procesu konwolucji



Łaty konwolucji 3×3 przy kroku 2×2 (bez dopełniania).

- Odległość między dwoma kolejnymi oknami jest parametrem konwolucji — **krokiem** (ang. *stride*).
- Domyślnie przyjmuje on wartość 1, ale można przypisać mu większą wartość, co doprowadzi do uzyskania tzw. konwolucji kroczących.
- Krok o wartości 2 oznacza, że szerokość i wysokość mapy cech są poddawane skalowaniu (są zmniejszane dwukrotnie).
- Zamiast kroków zwykle stosuje się operację skalowania **max-pooling**.

Operacja max-pooling

- Operacja ta ma w sposób agresywny zmniejszać rozdzielczość map cech,
- Operacja skalowania *max-pooling* jest przeprowadzana przy użyciu okien ekstrakcji przetwarzających mapy cech.
- Okna te zwracają maksymalne wartości każdego z kanałów - operacja tensorowa max (maksimum),
- Operacja *max-pooling* jest przeprowadzana zwykle przy użyciu okien o wymiarach 2×2 przy kroku równym 2 (ma to na celu dwukrotne zredukowanie mapy cech).
- Konwolucje są przeprowadzane przy użyciu okien o wymiarach 3×3 przy parametrze kroku równym 1

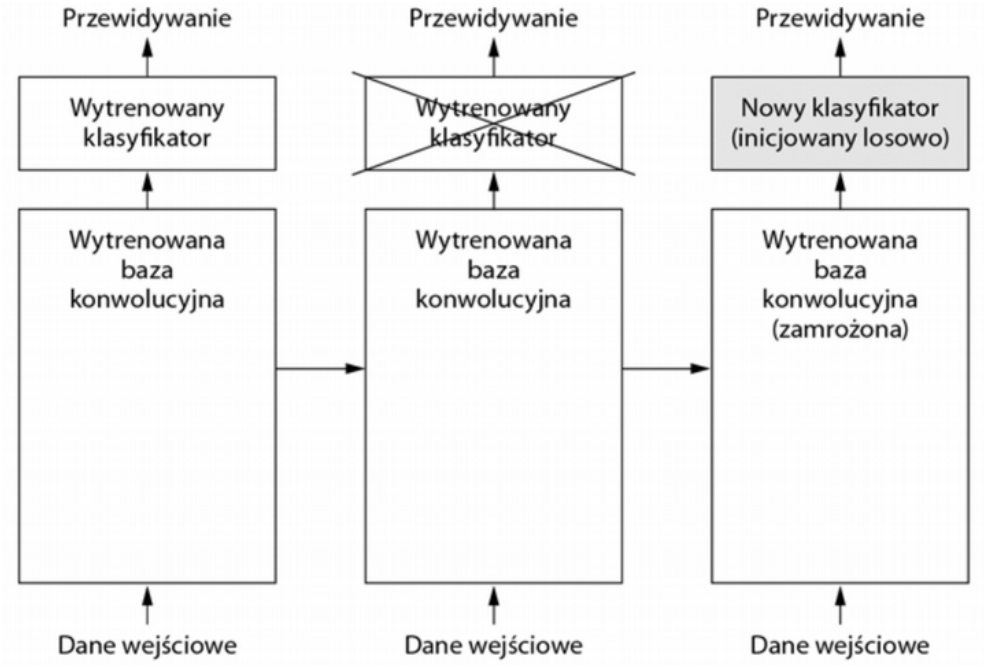
Trenowanie konwolucyjnej sieci neuronowej na małym zbiorze danych

- Często spotykana sytuacja podczas prywatnej pracy nad problemami analizy obrazu,
- *Mała liczba próbek* może oznaczać różną liczbę — od kilkuset do kilkudziesięciu,
- Uczenie głębokie działa głównie wtedy, gdy możliwe jest uzyskanie dostępu do dużej ilości danych,
- Algorytmy tego uczenia mogą samodzielnie wybrać przydatne cechy z treningowego zbioru danych, ale wymagają do tego licznego treningowego zbioru danych.
- Zbiór kilkuset przykładów może okazać się wystarczający, jeżeli model będzie mały i poddany regularyzacji, a zadanie będzie proste.

Technika augmentacji danych

- Nadmierne dopasowanie wynika ze zbyt małej liczby próbek, na których model może się uczyć,
- Augmentacja danych to technika generowania większej liczby elementów treningowego zbioru danych poprzez augmentację próbek na drodze losowych przekształceń zwracających obrazy,
- W Keras augmentacja wspomagana jest przez instancję *ImageDataGenerator*.

Ekstrakcja cech



- Ekstrakcja cech polega na korzystaniu z reprezentacji wyuczonej przez sieć wcześniej w celu dokonania ekstrakcji interesujących nas cech z nowych próbek.
- Cechy te są następnie przepuszczane przez nowy klasyfikator trenowany od podstaw.

Zamrażanie warstwy lub zestawu warstw

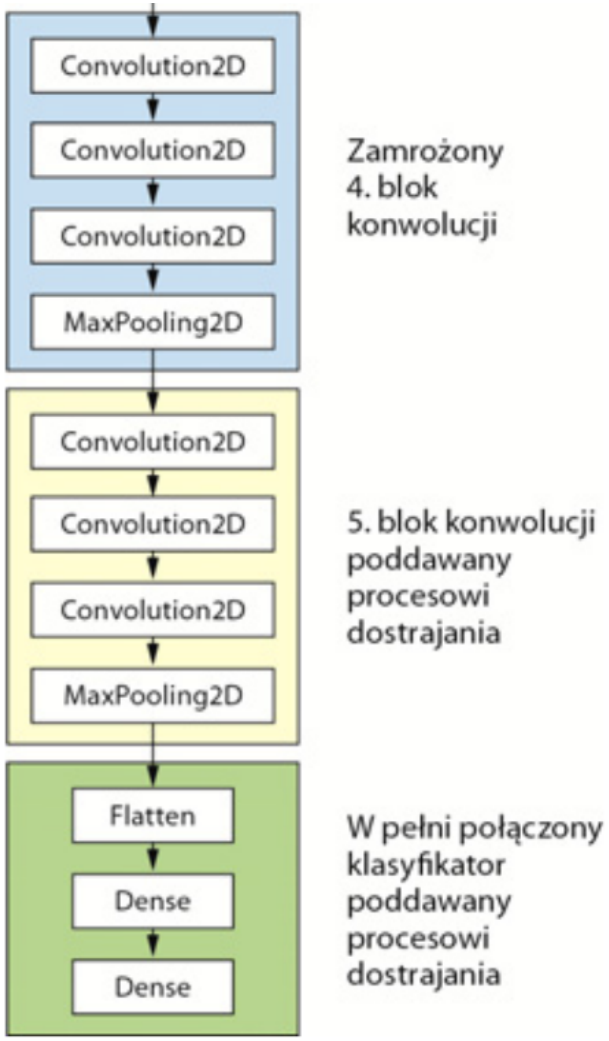
- Zamrażanie warstwy lub zestawu warstw polega na zapobieganiu aktualizacji ich wag w procesie trenowania.
- Be zamrażania reprezentacje wyuczone wcześniej przez bazę konwolucyjną zostaną zmodyfikowane podczas trenowania.
- Warstwy Dense znajdujące się u góry są inicjowane w sposób losowy, co sprawia, że dochodziłoby do dużych zmian wszystkich parametrów sieci, a to skutecznie zniszczyłoby wyuczone wcześniej reprezentacje.
- W pakiecie Keras sieć zamraża się, przypisując wartość *False* atrybutowi *trainable*

Dostrajanie

Dostrajanie jest techniką ponownego stosowania modeli uzupełniającą ekstrakcję cech.

- Polega ona na odmrażaniu kilku górnych warstw zamrożonej bazy modelu używanej do ekstrakcji cech i trenowaniu jej łącznie z nową częścią modelu,
- Najczęściej tą częścią modelu jest w pełni połączony klasyfikator,
- Proces ten określamy mianem dostrajania, ponieważ modyfikuje częściowo wcześniej wytrenowane bardziej abstrakcyjne reprezentacje modelu w celu dostosowania ich do bieżącego problemu.

Dostrajanie



Wizualizacja efektów uczenia konwolucyjnych sieci neuronowych

- **Wizualizacja pośrednich danych wyjściowych** - technika przydatna do zrozumienia tego, jak kolejne warstwy sieci konwolucyjnej przekształcają dane wejściowe, i tego, co robią poszczególne filtry sieci.
- **Wizualizacja filtrów sieci konwolucyjnej** - technika umożliwiająca dokładne zrozumienie graficznego wzorca lub graficznej koncepcji, na którą reaguje każdy z filtrów sieci konwolucyjnej.
- **Wizualizacja map ciepła aktywacji klas obrazu** — technika przydatna do określenia części obrazu zidentyfikowanych jako należące do danej klasy

Zadania na laboratoria

1. Dla danych “CIFAR10 small image classification” z biblioteki Keras zaproponuj i zrealizuj podział tych danych na dane treningowe i dane testowe w proporcji 3 do 7,
2. W oparciu o sieć konwolucyjną, zgodnie z wskazaniami prowadzącego, zaproponuj klasyfikator, klasyfikujący dwie z dziesięciu klas,
3. Zaproponuj klasyfikator z jedną, dwiema oraz trzema warstwami konwolucyjnymi,
4. Porównaj wyniki z poprzedniego punktu i zaproponuj optymalny klasyfikator dla danego problemu.