Sieć konwolucyjna
Sieci Neuronowe 2020

Jakub Ciszek
238035

Spis treści

1 Opis badań **3**

1.1 Plan eksperymentów 3

1.2 Charakterystyka zbiorów danych 3

2 Eksperymenty **4**

2.1 Wpływ wielkości filtra na przebieg procesu uczenia 4

2.2 Porównanie z MLP 7

3 Wnioski **10**

Cały kod wykorzystany w zadaniu znajduje się pod adresem: `https://github.com/Greenpp/sieci-neuronowe-pwr-2020`

1 Opis badań

1.1 Plan eksperymentów

Wszystkie eksperymenty zostały przeprowadzone 10 razy. Losowość przy inicjalizacji wag oraz generacji danych nie została narzucona żadnym ziarnem. Podczas badań przyjęto górną granicę 5 epok, po przekroczeniu której, uczenie zostawało przerywane. Ze względu na charakter zadania (klasyfikacja) na ostatniej warstwie użyto funkcji Softmax, a za funkcję straty przyjęto Entropię krzyżową. Użyta sieć składała się z warstwy konwolucyjnej, max pool, oraz w pełni połączonej ze 128 neuronami. Wagi były inicjalizowane metodą He, a uczenie przebiegało przy pomocy metody Adam. Jako funkcję aktywacji przyjęto ReLU. TODO model MLP Z powodów wydajnościowych testowanie modelu przeprowadzano co każde 32 paczki, z których każda składała się z 32 przykładów. Zgodnie z instrukcją zostały przeprowadzone następujące badania:

- Wpływ wielkości filtra na przebieg procesu uczenia
- Porównanie z MLP

Podczas wizualizacji funkcji straty pominięto pierwsze 10 pomiarów dla lepszej czytelności.

1.2 Charakterystyka zbiorów danych

Danymi użytymi w zadaniu jest zbiór ręcznie pisanych cyfr $0 - 9$ - MNIST. Na zbiór składa się 70,000 obrazów wielkości 28x28 pikseli. Na wyjściu znajduje się 10 klasom na wyjściu. Użyta w zadaniu wersja została podzielona na 3 zbiory:

- Uczący - 50,000 przykładów.
- Walidujący - 10,000 przykładów.
- Testowy - 10,000 przykładów.

W trakcie eksperymentów wykorzystano jedynie zbiory uczący i testowy.

2 Eksperymenty

2.1 Wpływ wielkości filtra na przebieg procesu uczenia

Założenia

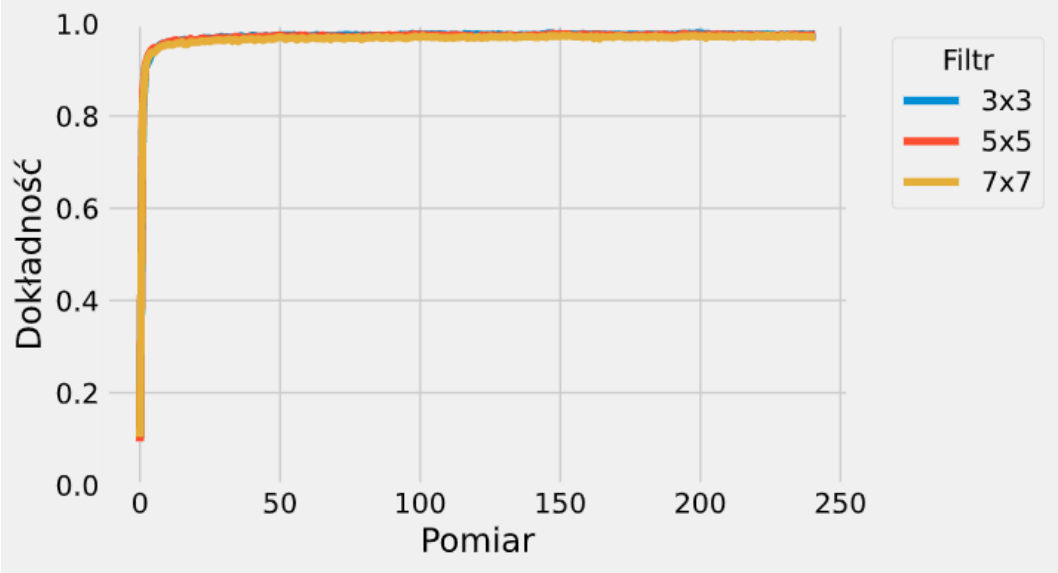
Zmienną w tym eksperymencie była wielkość filtra, przyjmowała wartości ze zbioru $\{3, 5, 7\}$

Przebieg

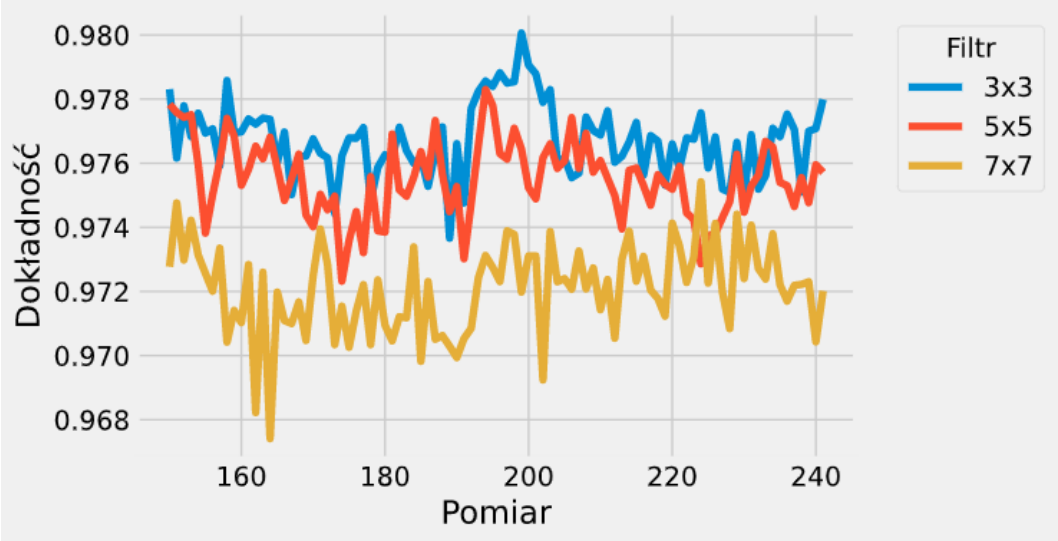
Podczas eksperymentu model został zainicjalizowany 10 razy dla każdej z badanych wartości oraz wyuczony, uzyskane wyniki zostały zapisane w postaci pliku .plk do dalszej analizy.

Wyniki

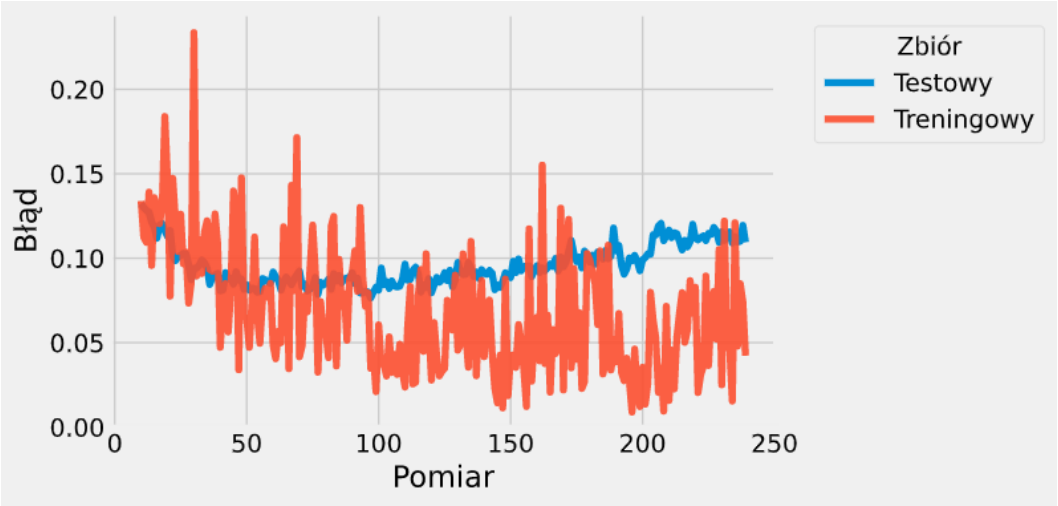
Wykres 1: Dokładność modelu w zależności od wielkości filtra



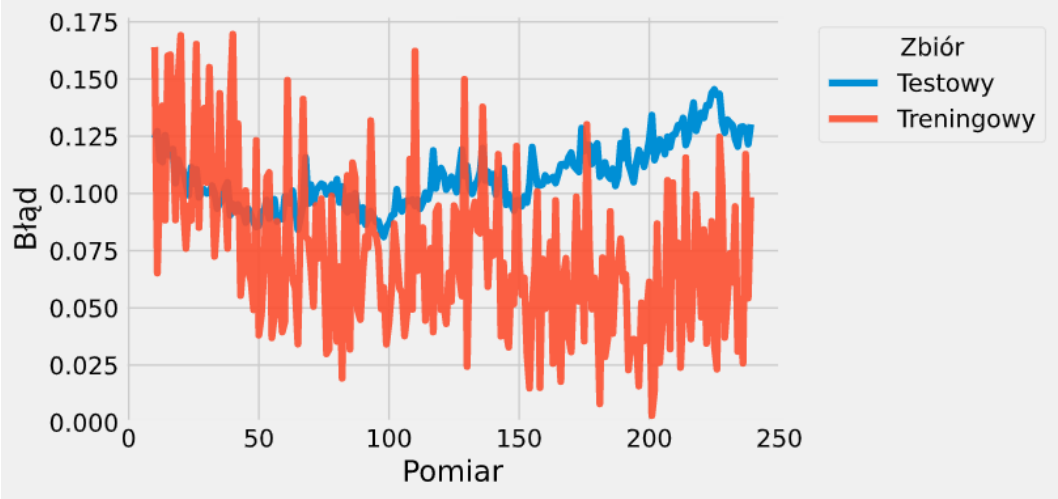
Wykres 2: Dokładność modelu w końcowym etapie uczenia w zależności od wielkości filtra



Wykres 3: Zachowanie funkcji błędu dla filtra wielkości 3



Wykres 4: Zachowanie funkcji błędu dla filtra wielkości 5



Wykres 5: Zachowanie funkcji błędu dla filtra wielkości 7

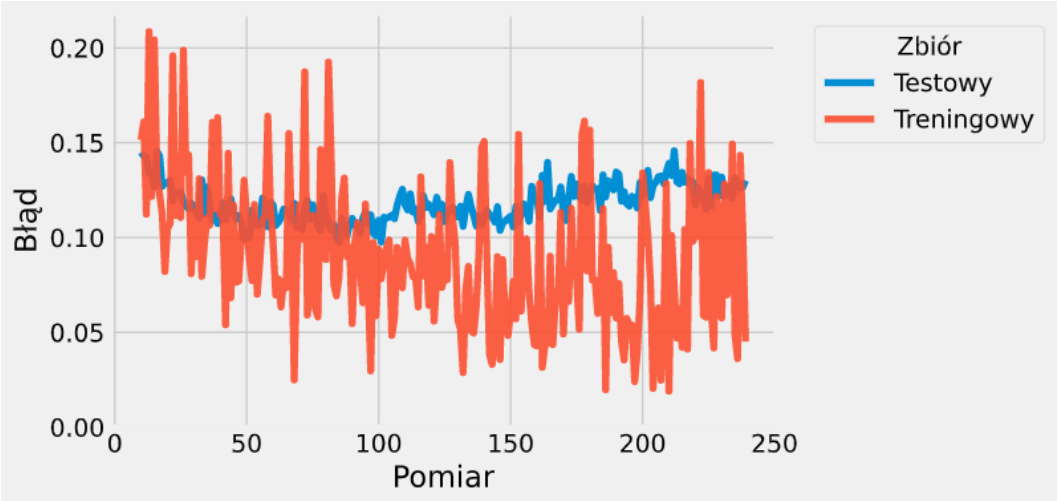


Tabela 1: Średnia maksymalna dokładność w zależności od wielkości filtra

Wielkość filtra	Dokładność [%]
3	**98.25**
5	98.16
7	97.88

Wnioski

Na przedstawionych wykresach 1, 2 oraz tabeli 1 widać, że wielkość filtra nie ma znacznego wpływu na osiągnięty wynik. Powodem takich rezultatów może być mały rozmiar istotnych cech wyuczanych przez filtry, co mogło by powodować zbieranie dodatkowych nieistotnych wartości przez większe filtry i nieznaczne obniżenie ich skuteczności. Niestety wartości wyuczonych filtrów nie zostały zapisane podczas eksperymentów, co uniemożliwia sprawdzenie tej hipotezy. Biorąc pod uwagę zachowanie funkcji błędu widoczne na wykresach 3, 4 oraz 5, można jednak przypuścić, że za gorsze wyniki modeli z większymi filtrami odpowiada ich liczba parametrów, co przekłada się na większe przeuczenie niż w przypadku filtra 3x3.

2.2 Porównanie z MLP

Założenia

Tabela 2: Stałe dla eksperymentu 2

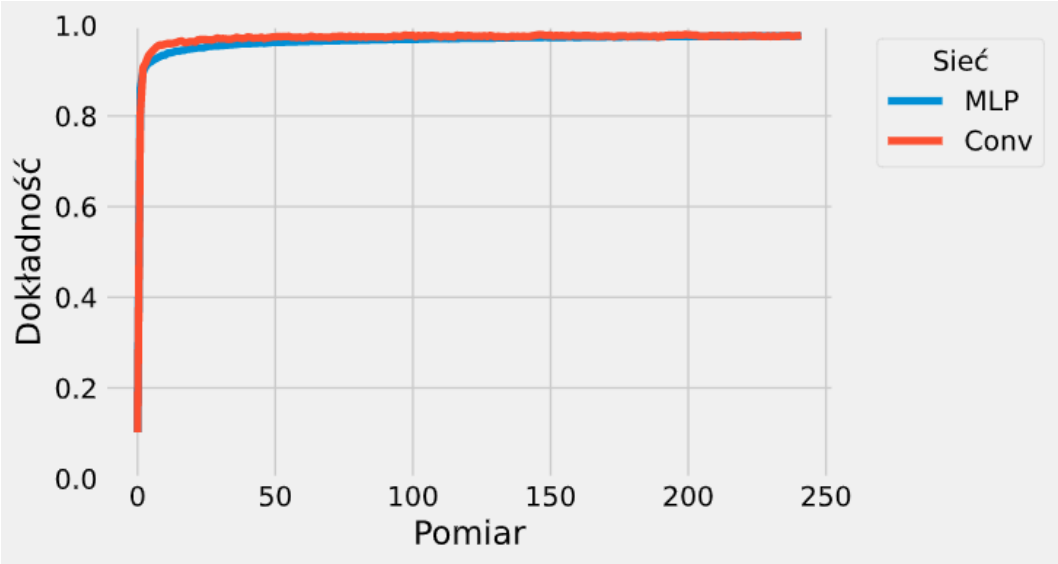
Parametr	Wartość
Wielkość filtra	3

Przebieg

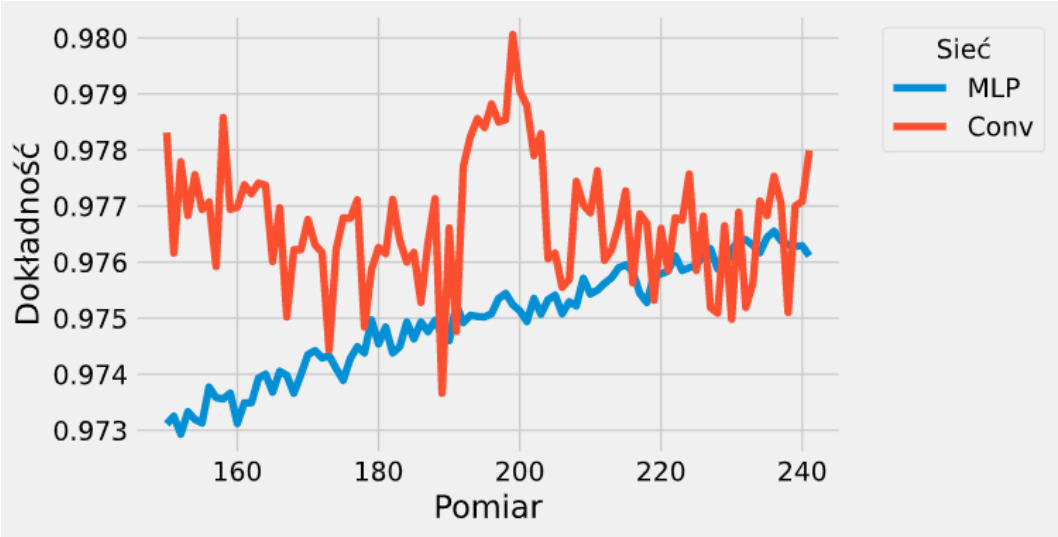
Porównano wyniki sieci konwolucyjnej z wielkością filtra dającą najlepsze wyniki (3x3) z siecią MLP która uzyskała najlepsze wyniki w poprzednim zadaniu tj. warstwa ukryta złożona z 512 neuronów, funkcja aktywacji ReLU, inicjalizacja wag He oraz optymalizator AdaGrad z początkowym współczynnikiem uczenia 0.01.

Wyniki

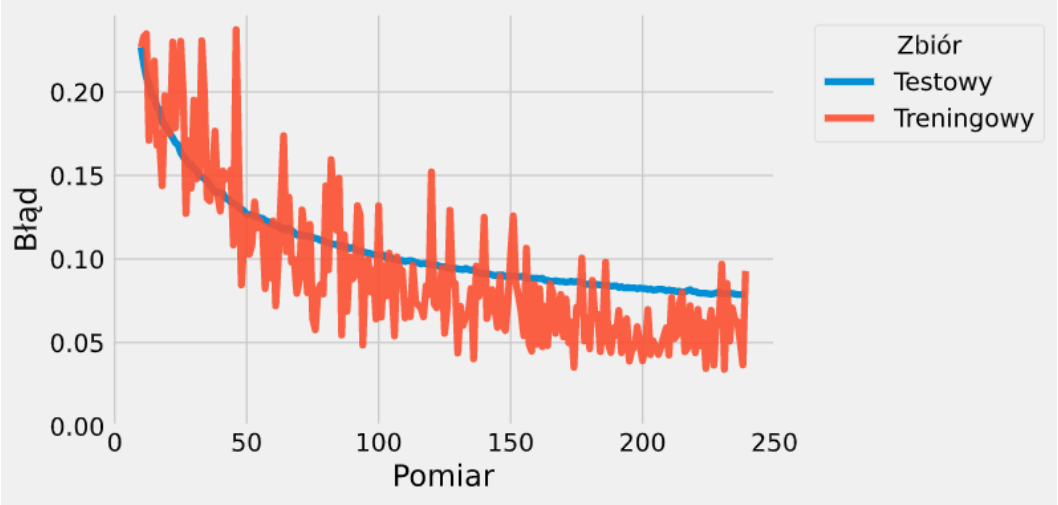
Wykres 6: Porównanie dokładności modeli



Wykres 7: Porównanie dokładności modeli w końcowym etapie uczenia



Wykres 8: Zachowanie funkcji błędu dla modelu MLP



Wykres 9: Zachowanie funkcji błędu dla modelu konwolucyjnego

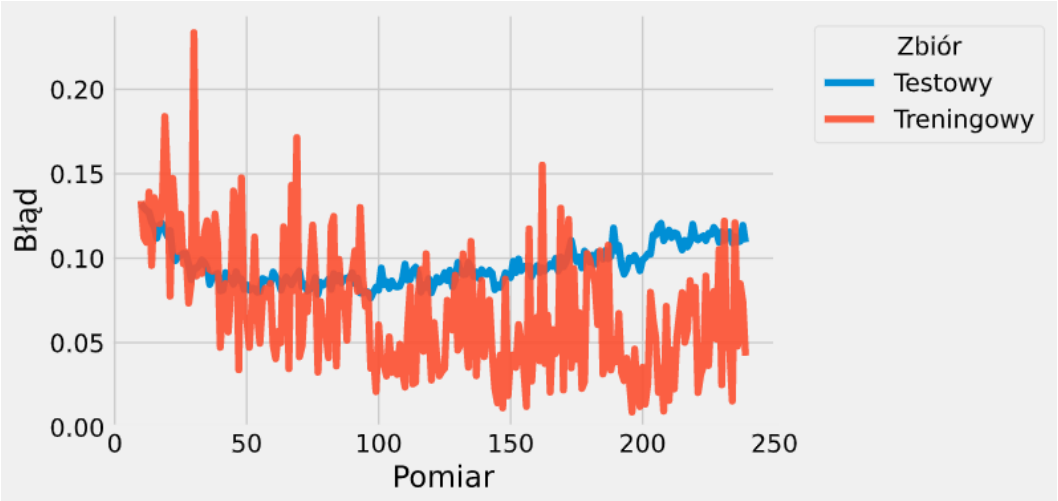


Tabela 3: Średnia maksymalna dokładność w zależności od modelu

Sieć	Dokładność [%]
MLP	97.73
Konwolucyjna	**98.25**

Wnioski

Na przedstawionych wykresach 6, 7 oraz tabeli 3 widać, że model z warstwą konwolucyjną osiąga lepsze rezultaty od zwykłej sieci MLP. Wpływ na to na pewno ma zdolność konwolucji do wydobywania lokalnych zależności między wartościami, co jest charakterystyczną cechą obrazów. Warstwa konwolucyjna służy jako wstępna ekstrakcja cech przed siecią MLP, co poprawia jej możliwości. Wyraźnie widoczna jest też przewaga sieci splotowej w szybkości osiągnięcia poziomu dokładności bliskiego do ostatecznego. Patrząc jednak na zachowanie funkcji błędu widoczne na wykresach 8 i 9 można zaobserwować wyraźną tendencję do przeuczenia w przypadku zastosowania konwolucji, w przeciwieństwie do MLP przy którym błąd testowy spada przez cały okres uczenia. Jest to zrozumiałe, jeśli model osiąga dobrą generalizację po zdecydowanie mniejszej ilości przykładów niż MLP to dalsze uczenie go powoduje zbytnie dopasowanie.

3 Wnioski

- Konwolucja idealnie nadaje się do ekstrakcji cech z obrazów, ponieważ wyłapuje lokalne zależności.
- Odpowiednia ekstrakcja cech z danych może poprawić skuteczność modelu.
- Używanie mniejszych filtrów, wydobywających drobniejsze cechy, może być skuteczniejszym podejściem niż nakładanie dużych i przy tym bardziej kosztownych konwolucji.
- Odpowiedni dobór modelu do typu danych skutkuje poprawą wyników.