Inteligentna interpretacja danych wizyjnych i uczenie głębokie

Przetwarzanie sygnałów i identyfikacja w sterowaniu i monitorowaniu urządzeń mechatronicznych

Ziemowit Dworakowski
AGH, Katedra Robotyki i Mechatroniki
zdw@agh.edu.pl

1

Zastosowanie metod SI w przetwarzaniu obrazów

- **Klasyfikacja/etykietowanie**
  Przypisywanie etykiet do obrazów lub ich fragmentów
- **Inteligentne globalne przetwarzanie obrazów**
  Kalibracja filtrów, filtracja, segmentacja, poprawa jakości
- **Modelowanie – zaawansowana interpretacja obrazów**
  Rozumienie co się dzieje na obrazie (i dlaczego)
- **Sztuczne generowanie obrazów**
  Budowa obrazów na podstawie promptów, augmentacja zbiorów danych

2

Zastosowanie metod SI w przetwarzaniu obrazów

Możliwe za pomocą uczenia płytkiego i głębokiego

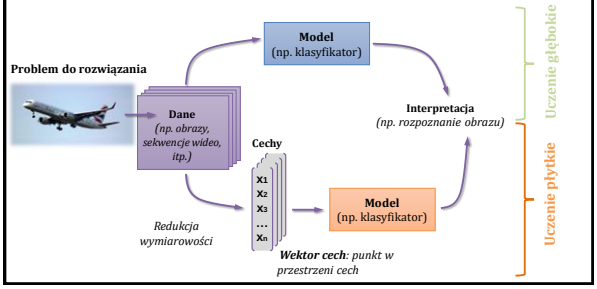
- **Klasyfikacja/etykietowanie**
  Przypisywanie etykiet do obrazów lub ich fragmentów
- **Inteligentne globalne przetwarzanie obrazów**
  Kalibracja filtrów, filtracja, segmentacja, poprawa jakości

Wyłącznie za pomocą uczenia głębokiego

- **Modelowanie – zaawansowana interpretacja obrazów**
  Rozumienie co się dzieje na obrazie (i dlaczego)
- **Sztuczne generowanie obrazów**
  Budowa obrazów na podstawie promptów, augmentacja zbiorów danych

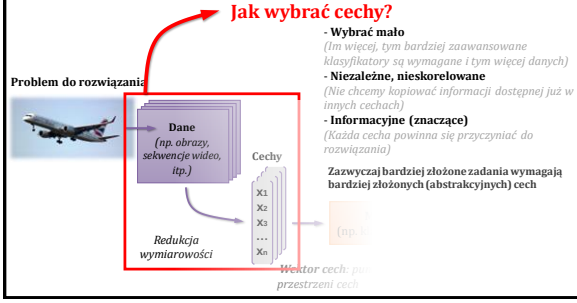
3

Podstawowa reprezentacja



4

Jak wybrać cechy?



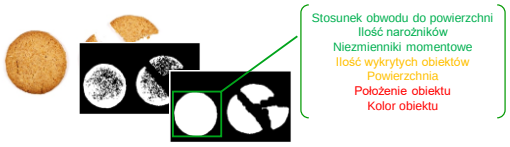
- **Wybrać mało**
  (Im więcej, tym bardziej zaawansowane klasyfikatory są wymagane i tym więcej danych)
- **Niezależne, nieskorelowane**
  (Nie chcemy kopiować informacji dostępnej już w innych cechach)
- **Informacyjne (znaczące)**
  (Każda cecha powinna się przyczyniać do rozwiązania)

Zazwyczaj bardziej złożone zadania wymagają bardziej złożonych (abstrakcyjnych) cech

5

Kroki wymagane do przygotowania typowego płytkiego klasyfikatora

1. Zebranie danych (*Dużo przykładów obiektów wszystkich klas*)
2. Przetwarzanie wstępne w celu podkreślenia różnic między klasami
3. Projekt dobrego wektora cech (*wrażliwego na różnice międzyklasowe*)
4. Projekt i trening klasyfikatora na **treningowym** podzbiorze danych
5. (opcjonalnie) Optymalizacja metaparametrów klasyfikatora na **walidacyjnym** podzbiorze danych
6. Test używając **testowego** pozbioru danych



6

Kroki wymagane do przygotowania typowego płytkiego klasyfikatora

1. Zebranie danych (*Dużo przykładów obiektów wszystkich klas*)
2. Przetwarzanie wstępne w celu podkreślenia różnic między klasami
3. Projekt dobrego wektora cech (*wrażliwego na różnice międzyklasowe*)
4. Projekt i trening klasyfikatora na **treningowym** podzbiorze danych
5. (opcjonalnie) Optymalizacja metaparametrów klasyfikatora na **walidacyjnym** podzbiorze danych
6. Test używając **testowego** pozbioru danych

Jeśli problem jest wystarczająco prosty i cechy są wystarczająco dobre, nawet prosty, niezoptymalizowany klasyfikator poradzi sobie z problemem. W przeciwnym wypadku konieczna może być jego dokładniejsza konfiguracja

Stosunek obwodu do powierzchni
Ilość narożników
Niezmienniki momentowe
Ilość wykrytych obiektów
Powierzchnia
Położenie obiektu
Kolor obiektu

7

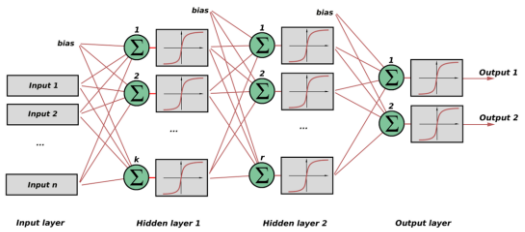
Założenia:

- Zadanie jest na tyle trudne, że prosty klasyfikator „nie daje rady"
- Zdecydowaliśmy się na zastosowanie sieci MLP

Decyzje do podjęcia:

- Jak podzielić dane na podzbiory?
- Jakiego algorytmu treningowego użyć?
- **Jaką strukturę powinna mieć nasza sieć?**
- **Jak ocenić poprawność podjętych decyzji?**

8

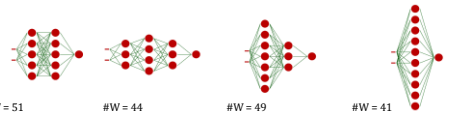


9

Jak wybrać strukturę sieci neuronowej?

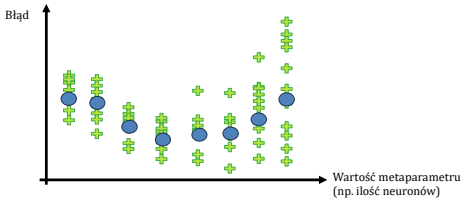
- Ile neuronów?
- Ile warstw?
- Jak przypisać neurony do warstw?

Najlepiej zacząć od jakiegoś „rozsądnego" punktu wybranego na podstawie doświadczenia. W przeciwnym wypadku narażamy się na rozwiązywanie bardzo złożonego problemu optymalizacyjnego



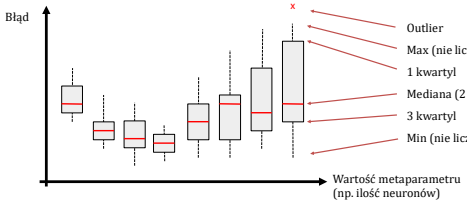
#W = 51	#W = 44	#W = 49	#W = 41
Uniwersalna	Wysoki poziom abstrakcji, trudny trening	Uniwersalna, z naciskiem na reprezentację	Niski poziom abstrakcji, łatwy trening

10



Zastosowanie uśrednienia Uśrednij wyniki kilku testów dla każdej wartości metaparametru. Zazwyczaj potrzeba tylko kilku przykładów by zauważyć ogólne tendencje (tzn. gdzie spodziewać się minimum)

11



Zastosowanie uśrednienia Uśrednij wyniki kilku testów dla każdej wartości metaparametru. Zazwyczaj potrzeba tylko kilku przykładów by zauważyć ogólne tendencje (tzn. gdzie spodziewać się minimum)

Zastosowanie wykresu pudełkowego Często wartości metaparametrów nie wpływają tylko na wartość średnią ale też na rozrzut wyników. Wielokrotne powtórzenie testów i przedstawienie statystyki na wykresie pudełkowym może pomóc w zrozumieniu rezultatów.

12

Uwagi ogólne:

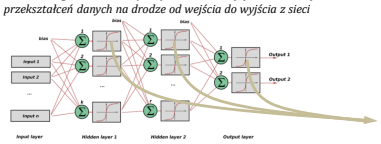
1. **Klasyfikator powinien być wybierany na podstawie naszej wiedzy o przestrzeni cech**
   Ile wymiarów? Ile cech? Czy widzimy wyraźne klastry? „Rodzaj zadania", (np. klasyfikacja warzyw ciasteczek czy zwierząt) jest dużo mniej istotny.
2. **Podzbiory treningowy, walidacyjny i testowy muszą być rozłączne!**
   Zaczynamy od losowego podziału danych na podzbiory lub (lepiej!) zbieramy każdy podzbiór w osobnym eksperymencie.
3. **Liczba stopni swobody modelu (tj. ustawialnych parametrów) zależy od ilości danych**
   Dobrą praktyką jest trzymanie się min. 10 punktów danych na każdy trenowalny parametr modelu
4. **Jakość cech > Konfiguracja klasyfikatora**
   Dobre cechy pozwalają na dobrą separację danych nawet z zastosowaniem słabego klasyfikatora. Słabe cechy nie pozwolą na uzyskanie dobrych wyników pomimo spędzenia setek godzin na optymalizacji modelu. Lepiej w pierwszej kolejności zainwestować czas i zasoby we wstępne przetwarzanie danych

13

	Różnica strukturalna	Różnica funkcjonalna
Shallow learning:	Głębokość CAP <= 3	Cechy jako wejście
Deep learning:	Głębokość CAP > 3	Surowe dane jako wejście

To rozróżnienie jest dużo ważniejsze...

Credit Assignment Path (CAP) *-> liczba kolejnych nieliniowych przekształceń danych na drodze od wejścia do wyjścia z sieci*



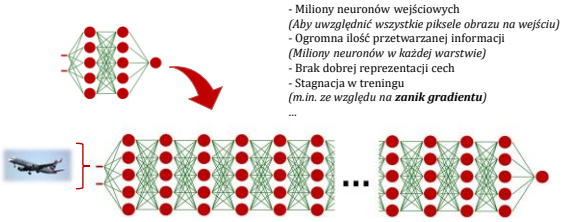
CAP = 3

14

Jak **nie** budować głębokiego modelu?

Efekt:

- Miliony neuronów wejściowych
  (Aby uwzględnić wszystkie piksele obrazu na wejściu)
- Ogromna ilość przetwarzanej informacji
  (Miliony neuronów w każdej warstwie)
- Brak dobrej reprezentacji cech
- Stagnacja w treningu
  *(m.in. ze względu na **zanik gradientu**)*
- ...



15

Więc... Co właściwie pozwala na automatyczne nauczenie się przetwarzania surowych danych?

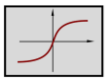
- **Aktywacje ReLU i Leaky ReLU**
- **Kernele konwolucyjne**
- **Pooling**
- **Regularyzacja**
- **Fine-tuning**

To podstawowe koncepty. Większość typowych modeli uczenia głębokiego używa ich w jakiejś formie

- Adversarial training
- Atencja
- Autoenkodery
- Latent space
- Embeddings
- Uczenie ze wzmocnieniem
- Transfer learning
- Transformery
- Sieci rekurencyjne (np. Long-Short-Term-Memory)

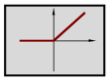
Te koncepty są bardziej zaawansowane, ich wybór pozwala na zrozumienie zaawansowanych modeli typu **Midjourney** czy **ChatGPT**

16



Funkcja sigmoidalna

+ Dobrze działa w przypadku małych sieci
- Może powodować zanik gradientu (powodując stagnację treningu)

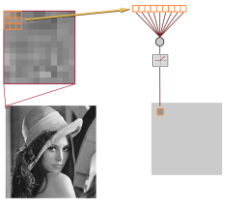


Funkcja aktywacji typu *Rectified Linear Unit* (ReLU)

+ Przeciwdziała zanikowi gradientu
+ Przyspiesza trening
-/+ Może powodować „umieranie" neuronów

17

Kernel konwolucyjny



Zapewnia:

*** Filtrację konwolucyjną**

*** Filtracje nieliniowe**

*** Filtracje morfologiczne**

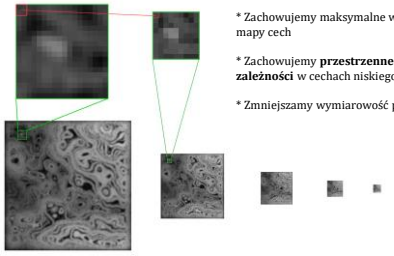
*** Wydobywa cechy niskiego poziomu**
(np. rozpoznawanie narożników)

* Dzięki ***dzieleniu wag*** jedna warstwa może realizować wiele zadań w tym samym czasie

* Wyjściem są **mapy cech**

18

(Max) pooling



* Zachowujemy maksymalne wartości mapy cech

* Zachowujemy **przestrzenne zależności** w cechach niskiego poziomu

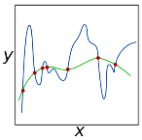
* Zmniejszamy wymiarowość problemu

19

Regularyzacja

Aby uniemożliwić zapamiętywanie danych (przeuczenie) wprowadzamy dodatkowe ograniczenia w treningu sieci

W praktyce zazwyczaj *„iteracyjnie psujemy model"*. W momencie osiągnięcia generalizacji (gdy klasyfikator zaczyna się dopasowywać do szumów w danych), czynnik regularyzacyjny zaczyna dominować uniemożliwiając dalsze dopasowanie.



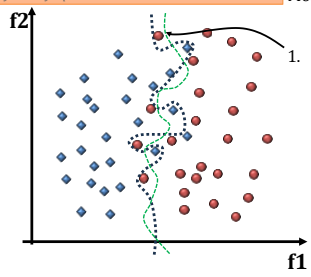
***Na przykład**:*
• W algorytmie gradientowym, poza aktualizacją wag wynikającą z kierunku gradientu dodajemy niewielką losową modyfikację do części lub całości wag

• Niektóre połączenia między neuronami są usuwane (tzw. metoda „brain damage")

20

Przykład graficzny – wyłącznie na potrzeby zrozumienia idei regularyzacji. W praktyce głębokie sieci nigdy nie działają w dwuwymiarowych przetrzeniach cech!

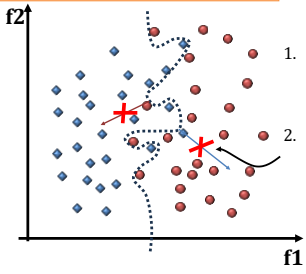
Możemy zatrzymać ten proces:



1. Dodając szum losowy do wag, nie pozwalając na idealne dopasowanie

21

Przykład graficzny – wyłącznie na potrzeby zrozumienia idei regularyzacji. W praktyce głębokie sieci nigdy nie działają w dwuwymiarowych przetrzeniach cech!



Możemy zatrzymać ten proces:

1. Dodając szum losowy do wag, nie pozwalając na idealne dopasowanie
2. Wprowadzając karę za zbyt duże wagi (uniemożliwiając dalsze zwiększanie złożoności dopasowania)

22

Fine tuning

Każdorazowo, w uczeniu głębokim potrzebujemy
ogromnych ilości danych i ogromnej mocy obliczeniowej

W przypadkach specjalistycznych tych danych nie będzie wystarczająco dużo, trening modelu może być również nieopłacalny czasowo i energetycznie

Rozwiązanie:
Trening ogólny na dużym zbiorze, **fine-tuning** (douczanie) na zbiorze specjalistycznym

Robimy to poprzez reinicjalizację wag w końcowych warstwach sieci, często jednocześnie zamrażając wagi w warstwach początkowych

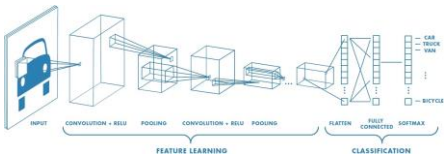
23

Fine tuning - przykłady

Nauczenie dużego modelu językowego na dużym, ogólnym zbiorze tekstów	Douczenie go na twórczości jednego autora, by umożliwić dokładne naśladowanie jego stylu
Nauczenie głębokiej sieci neuronowej rozpoznawania gatunków różnych zwierząt	Douczenie jej na przykładach kilku tysięcy zdjęć wiewiórek dwóch gatunków – do specjalistycznego zliczania wiewiórek w parku narodowym
Nauczenie głębokiego modelu do rozpoznawania mowy na zbiorze nagrań mówiących ludzi	Douczenie go na zbiorze danych ultradźwiękowych (?!) w celu odróżniania struktur uszkodzonych od zdrowych

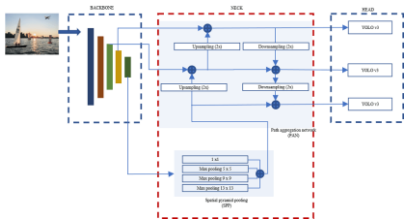
24

Przykłady: Matlab DCNN



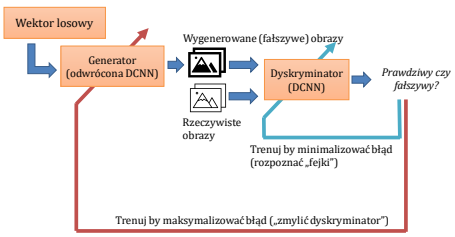
25

Przykłady: YOLO v4



26

Przykłady: Sieć generatywno-przeciwstawna



27

Podsumowanie (tematy na test):

1) Kategorie zadań w inteligentnym przetwarzaniu obrazów (z przykładami)
2) Schemat przetwarzania głębokiego i płytkiego (wymagane kroki, z przykładem)
3) Funkcjonalna i strukturalna różnica między płytkim i głębokim uczeniem
4) Różnice pomiędzy różnymi strukturami sieci MLP
5) Jak możemy zoptymalizować sieć MLP? (Uśrednianie, wykres pudełkowy)
6) 4 uwagi ogólne w treningu klasyfikatorów
7) Wyjaśnij funkcję ReLU, porównaj z sigmoidalną
8) Wyjaśnij kernel konwolucyjny
9) Wyjaśnij max pooling
10) Wyjaśnij regularyzację
11) Wyjaśnij fine tuning (douczanie) – z przykładami

(*Szczegółowych schematów głębokich sieci neuronowych nie będzie na teście*)

28