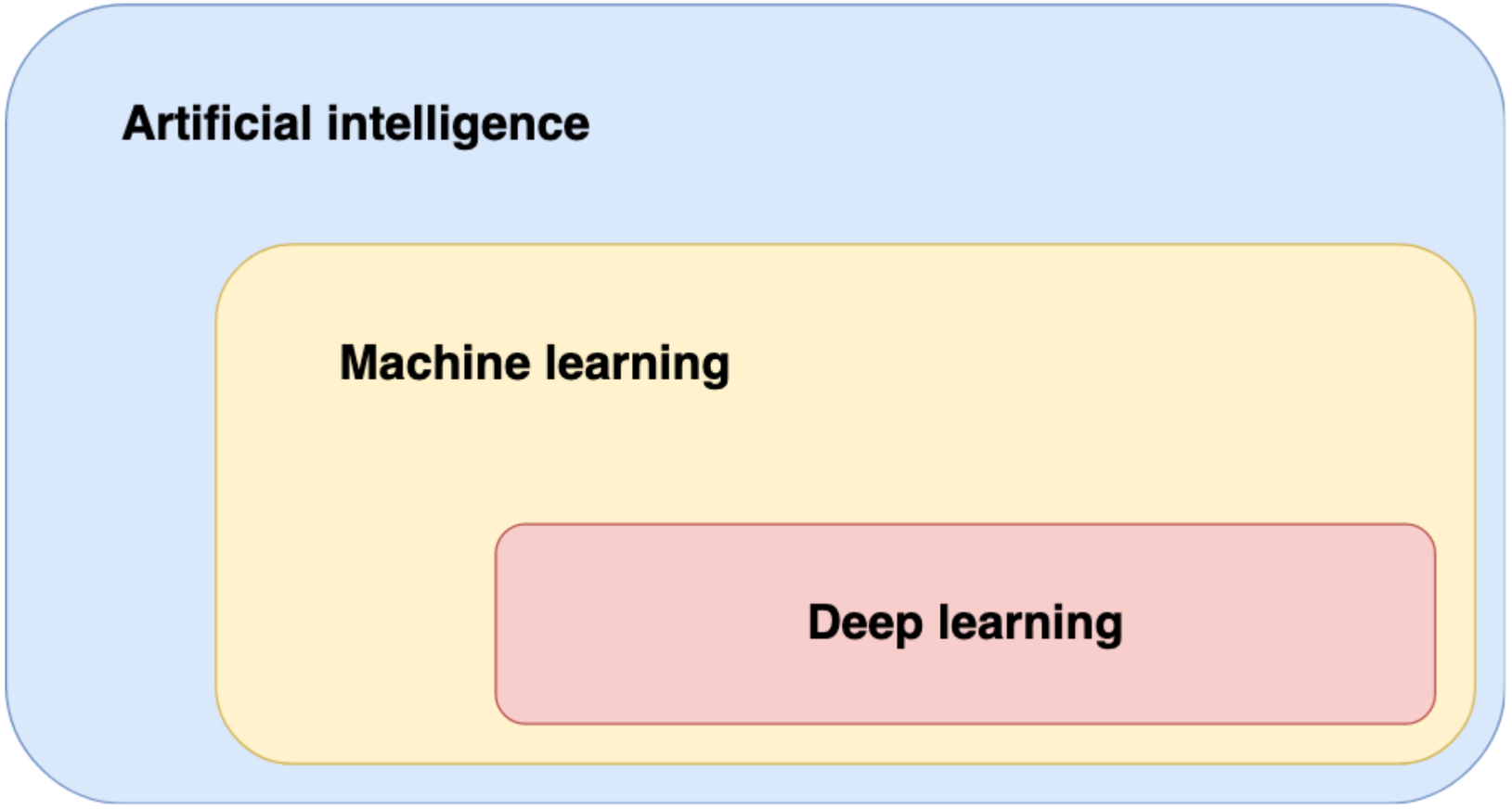
TENSORFLOW.JS

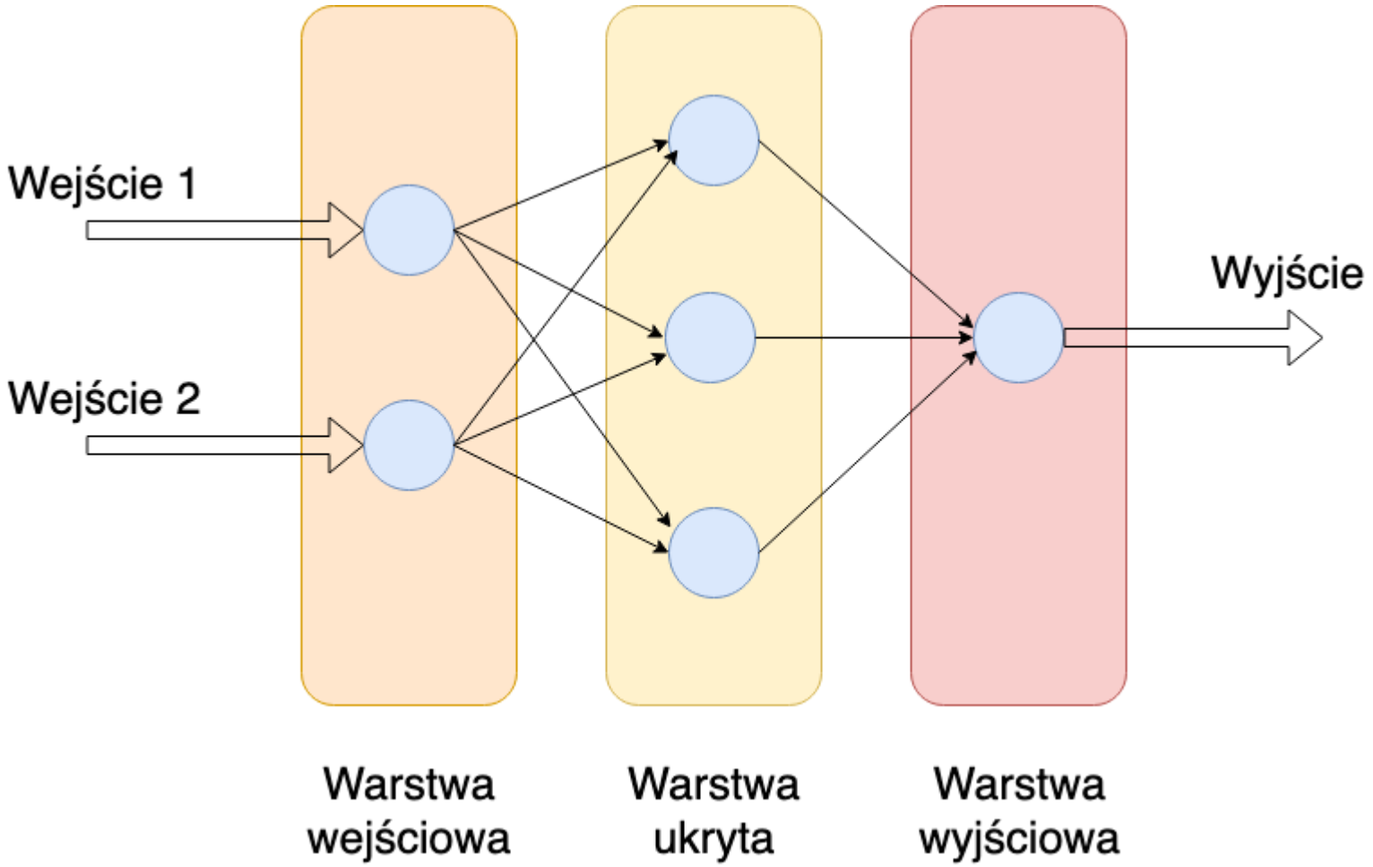
MAREK BOGATZKI

STATS PERFORM

- Wprowadzenie do AI
- Tensorflow.js
- Podstawowe pojęcia
- Przykładowy kod


Artificial intelligence
Machine learning
Deep learning

SZTUCZNA SIEĆ NEURONOWA (ANN)



NORMALIZACJA

Przeskalowanie danych, aby były w zakresie od 0 do 1 lub od -1 do 1. Dzięki temu unika się problemów w modelach treningowych, które mają wyjątkowo duże lub małe wartości (przepełnienie / niedopełnienie).

Jedną z najprostszych metod skalowania jest normalizacja min-max:

$$z = \frac{\mathrm{x} - min(x)}{[\max(x) - min(x)]}$$

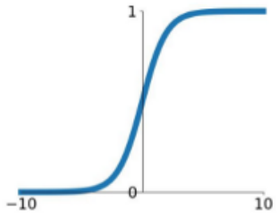
FUNKCJA AKTYWACJI

(ACTIVATION FUNCTION)

Funkcja, według której obliczana jest wartość wyjścia neuronów.

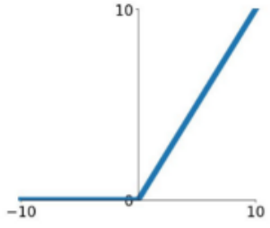
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



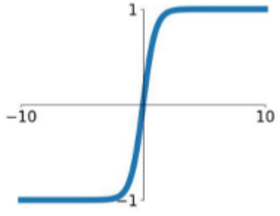
ReLU

$$\max(0, x)$$



tanh

$$\tanh(x)$$



FUNKCJA BŁĘDU

(LOSS FUNCTION, COST FUNCTION)

Funkcja mierząca jak bardzo nasz model się myli tzn. jak bardzo wynik jego predykcji jest różny od spodziewanego wyniku.

Przykładowe funkcje:

- MeanSquaredError - dla regresji
- Cross Entropy - dla klasyfikacji

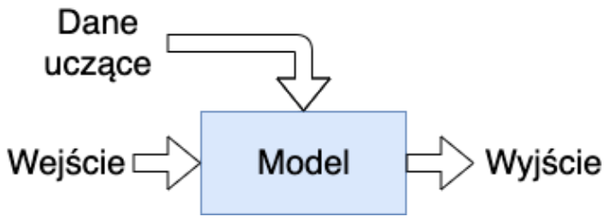
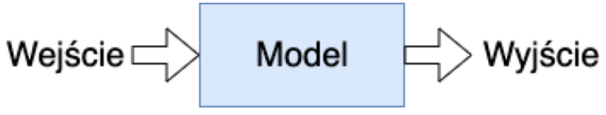
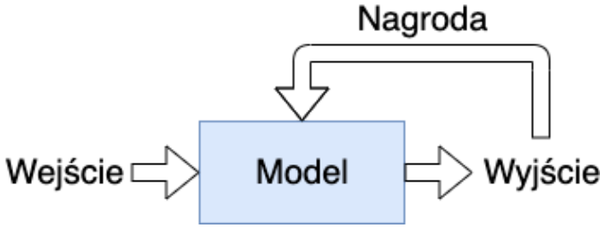
FUNKCJA OPTYMALIZUJĄCA

(OPTIMIZER)

Funkcja, która na podstawie wyniku zwróconego przez funkcję błędu (loss) modyfikuje wagi tak, aby błąd był jak najmniejszy.
Najpopularniejsze funkcje optymalizujące:

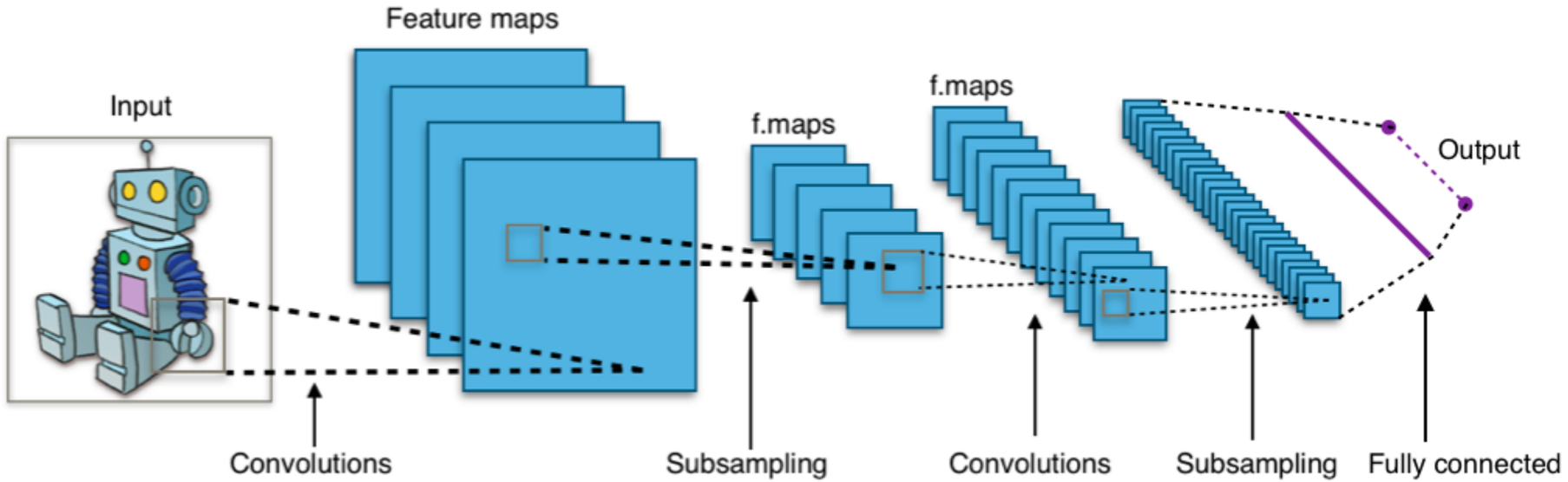
- Adam
- SGD

UCZENIE MASZYNOWE

Uczenie nadzorowane	Uczenie nienadzorowane	Uczenie przez wzmacnianie
uczenie jak klasyfikować albo przewidzieć wartość na podstawie poprzednich obserwacji	znajdowanie wzorca w istniejących danych	dostarczanie pozytywnej lub negatywnej informacji zwrotnej bazując na spodziewanym wyniku
		
klasyfikacja, regresja	grupowanie, rekomendacje	gry, robotyka
przewidywanie kiedy klient może chcieć anulować subskrypcje	znajdowanie grupy użytkowników o podobnych subskrybcjach	wygrywanie w grę Pong

GŁĘBOKIE UCZENIE

- Rekurencyjne sieci neuronowe (RNN)
- Konwolucyjne sieci neuronowe (CNN)
- Generatywne sieci współzawodniczące(GAN)



TENSORFLOW.JS

- biblioteka do uczenia maszynowego w JavaScript
- stworzona przez Google
- dwa API - Core i Layers
- Layer API wzorowane na Keras
- początkowo deeplearn.js

KTO UŻYWA TENSORFLOW?

i wiele innych…

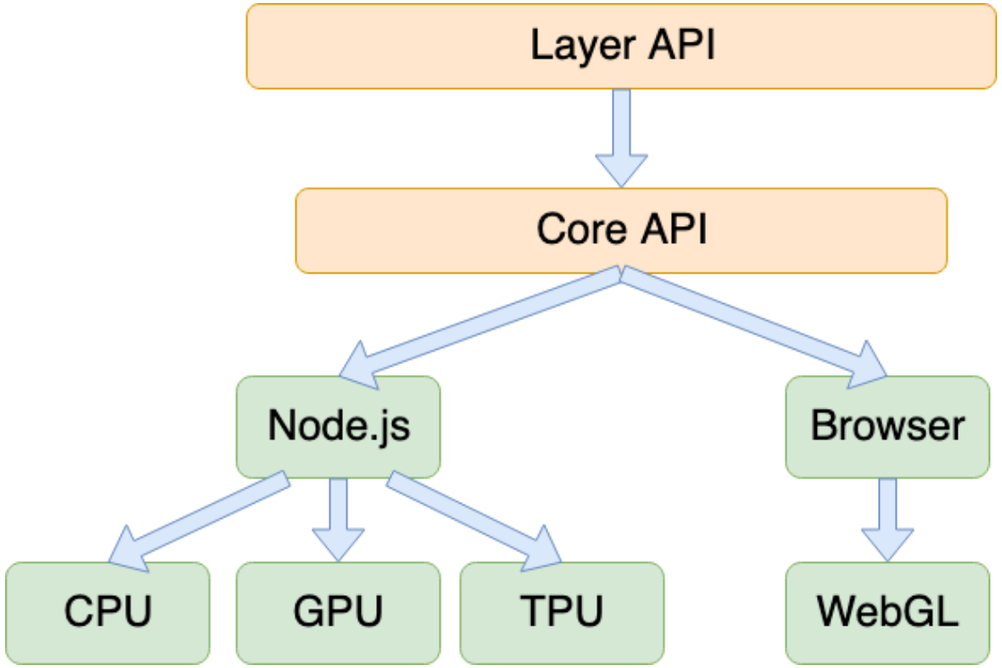
MOŻLIWOŚCI

- Rozpoznawanie mowy
- Analiza sentymentalna
- Rozpoznawanie obrazów
- Systemy rekomendacji
- Przewidywanie czasu dojazdu
- Przewidywanie wzrostu cen na giełdzie
- Detekcja wideo

ZALETY

- do rozpoczęcia przygody wystarczy edytor tekstowy i przeglądarka
- bezpieczeństwo danych
- możliwość użycia gotowych modeli
- możliwość importowania modeli Keras stworzonych w Pythonie
- obsługa WebAssembly w trakcie implementacji
- akceleracja przy pomocy GPU

STRUKTURA



- Layers API
  - wysokopoziomowe definiowanie modeli
  - warstwa abstrakcji dla sztucznej sieci neuronowej
  - zbudowane na Core API
  - inspirowane Keras
- Core API
  - niskopoziomowe definiowanie modeli
  - narzędzia do transformacji macierzy i wektorów
  - przykładowe metody: tf.add, tf.sub, tf.mul, tf.div, tf.maximum

FRONTED	VS	BACKEND
• tf.browser • akceleracja tylko za pomocą WebGL		• tf.node • akceleracja z użyciem Nvidia CUDA • dostęp do systemu plików • Tensorboard • Dekodowanie różnych typów obrazów (BMP, GIF, PNG…)

AKCELERACJA Z UŻYCIEM WEBGL

1. Transformacja danych wejściowych (tensora) na teksturę
2. Kopiowanie tekstury do pamięci
3. Przeprowadzenie obliczeń jak na Fragment Shaderze

PRZYKŁADY TENSORFLOW.JS Z WYKORZYSTANIEM KAMERY:

- Pacman sterowany gestami
- Detekcja obrazów

PRZEBIEG PRACY Z TENSORFLOW.JS

1. Przygotowanie danych
2. Zbudowanie modelu
3. Uczenie/trenowanie modelu
4. Zapisanie i używanie modelu

PODSTAWOWE POJĘCIA

TENSOR

```
tf.scalar(3.14)    // => Tensor 3.140000104904175
tf.tensor1d([1,2]) // => Tensor [1, 2]

// Wszystkie poniższe dają ten sam efekt
tf.tensor2d([[1,2], [3,4]])
tf.tensor2d([1,2,3,4], [2, 2])
tf.tensor([[1,2], [3,4]])
tf.tensor2d([1,2,3,4], [2, 2])

// Tensor  [[1, 2],
//          [3, 4]]

//tf.tensor(value, shape?, dtype?)
//dtype - float, int, boolean
```

MODEL

```
const model = tf.sequential();
const model = tf.model({inputs: input, outputs: output});
```

```
model.compile({
  loss: "meanSqueredError", // jak bardzo predykcja sieci odbiega od wyniku
  optimizer: tf.train.adam(0.05), // funkcja minimalizująca błąd
});
```

```
model.fit(trainingData, outputData, {
  callbacks: someFunction, // callback wywoływany podczas uczenia
  epochs: 3, // ilość iteracji po zbiorze treningowym
});
// zwraca promisa!
```

```
model.predict(testData).print();
/* Tensor
[[8.7450171],
[10.561533],
[12.3780499],
[14.1945667]]
*/
```

LAYER

Przykładowe typy warstw:

- `tf.layers.activation`
- `tf.layers.dense`
- `tf.layers.conv2d`

```
tf.layers.dense({
  units: 10,                // ilość wyjść
  inputshape: [28,28,1],    // kształt tensora np. 28x28 pikseli
  activation: 'sigmoid'     // funkcja aktywacji
});
```

PRAKTYKA

tiny.cc/meetup-tfjs

PRAKTYCZNE ZASTOSOWANIA

- Rozpoznawanie kategorii po opisie przedmiotu
- Przewidywanie następnej strony i ładowanie jej w tle
- Polecanie innych artykułów na podstawie tagów

PRZYDATNE LINKI

- ML5.js
- P5.js
- Tensorflow playground
- Dokumentacja Tensorflow