Assignment

Zadania już prawie się nie zmienią

October 2022

1. Projekty zaliczeniowe.

   Poniższe zadania zaliczeniowe można wykonać samodzielnie lub w grupach do 3 osób. Każda z osób/grup powinna zapisać w Pythonie rozwiązanie jednego wybranego przez siebie problemu. Projekty mają 3 stopnie trudności.

   Łatwy Trening na zbiorze MNIST: Popraw model rozwiązujący problem identyfikacji liczb napisanych ręcznie. Popraw oznacza zwiekszenie skuteczńści nie doprowadzając do overfittingu. Możliwe są wszelkie modyfikacje: zmiana hiperparametrów, struktury sieci, rodzaje warstw, modyfikacje zbioru danych itd. W rozwiązaniu powinien znajdować się raport z opisami sieci i otrzymanymi wynikami, plusy za replikowalność. W naszej stuacji samodzielne/ręczne eksperymenty są wartościowe natomiast warto je porównać z wynikami użycia narzędzi do automatycznego poszukiwania parametrów np. Optuna.

   Średni Kontynuowanie rozwiązania. Zadanie: Zaprojektuj i wytrenuj model, który będzie przedłużał fragmenty rozwiązania równania Lotki–Volterry. Przykładwe wejście: n punktów reprezentujących próbkę rozwiązania w równych odstępach czasu. Sieć powinna zwracać k kolejnych punktów przedłurzenia rozwiązania danego na wejście.

   ✻ N.N. ✻ Rozpoznanie q w q-closest Cucker-Smale. Zakładamy, że znamy wagę w równaniu Cucker-Smalea. Zadanie polega na zaprojektorwaniu i wytrenowaniu sieci, która na wejsciu otrzymuje fragment rozwiązania równania q-closest i zwraca liczbę q, doprecyzowanie pytania jest częścią zadania.

   N.N. - non notus, trudność zadania jest nieznana, z tego względu $\mathbb{P}$-stwo tego, że sie "nie uda" jest wysokie. Sugeruje się, pracę nad nim w sytuacji poczucia pewności, że pozostałe umie się zrobić w okreslonym, któtkim czasie.

2. Zadania, które umiemy rozwiązywać.

   (a) matplotlib + numpy

      i. Narysuj niebieski, przerywany wykres funkcji $0 \vee \sin(x)$ na przedzale $(-2.64, 4.67)$.

      ii. Narysuj rodzinę losowych kół

   (b) Rysowanie Cuckera-Smale'a rozbite na kilka zadań + q-closest.

      i. Napisz funkcję, która generuje losowy warunek początkowy dla równania CS dla n ptaków w m wymiarach, o parametrach:
      n - liczba ptaków,
      m - wymiar,
      pos_distr - rozkład zmiennej losowej pozycji "ptaków",
      vel_distr - rozkład zmiennej losowej prędkości "ptaków".

ii. Napisz funkcję, która dla tablicy o wymiarach $(n, 4)$

$$\left[\begin{array}{cc} \overrightarrow{x} & \overrightarrow{v} \end{array}\right] = \begin{bmatrix} {x^1}_0 & {x^2}_0 & {v^1}_0 & {v^2}_0 \\ {x^1}_1 & {x^2}_1 & {v^1}_1 & {v^2}_1 \\ \vdots & & & \vdots \\ {x^1}_{n-1} & {x^2}_{n-1} & {v^1}_{n-1} & {v^2}_{n-1} \end{bmatrix}$$

zwraca tablicę o wymiarach $(n, 2)$, w której i-ty wiersz jest równy:

$$\sum_{j=0}^{n-1} \eta(\|x_j - x_i\|)\,(v_j - v_i)\,,$$

gdzie

$$x_i = \left[{x^1}_i\,,\ {x^2}_i\right] \quad \text{oraz} \quad v_i = \left[{v^1}_i\,,\ {v^2}_i\right].$$

iii. Napisz funkcję, która dla tablicy jak w punkcie 2(b)ii i $q \in \mathbb{N}$ zwraca tablicę wymiarach $(n, 2)$, w której i-ty wiersz jest równy:

$$\sum_{j \in \mathcal{N}_i} \eta(\|x_j - x_i\|)\cdot(v_j - v_i) = \sum_{j=0}^{n-1} \mathbb{1}\left(\mathcal{N}_i\right)(j)\cdot\eta(\|x_j - x_i\|)\cdot(v_j - v_i)\,,$$

gdzie

$$\mathcal{N}_i = \mathcal{N}_i(q, \overrightarrow{x}) = \{j \in I_n \setminus \{i\} \mid \ \#\{x_k : dist(x_i, x_k) < dist(x_i, x_j)\} < q\}\,.$$

Innymi słowy $\mathcal{N}_i$ to indeksy q-najbliższych sąsiądów ptaka znajdującego się w punkcie x_i. Wytłumacz jak rozwiązujesz problem gdy powyższa definicja nie jest jednoznaczna.

iv. (dodatek) Uzupełnij prawą stronę równania o dodadkowy parameter
r - zasięg postrzegania (perception range),
ptaki dostosowują swoje prędkosci tylko na podstawie sąsiadów w zasięgu t.j. ptak x_i patrzy tylko na ptaki z $\mathbb{B}(x_i, r)$

v. Napisz program, który rozwiązuje równanie CS dla losowego warunku początkowego o wymiarach (n,4) z $n > 10$ oraz produkuje animację zachowania się ptaków w zależności od czasu.

Oznaczenia

- $I_n = \{0, 2, 3, \ldots, n-1\}$