Zadanie 2

Algorytmy ewolucyjne

Kacper Kania

12.03.2024

Treść zadania

Algorytm ewolucyjny jest jednym z podstawowych podejść do optymalizacji bezgradientowej (z ang. *gradient-free optimization*). Dzięki temu, algorytmy ewolucyjne znajdują szczególnie zastosowanie w zadaniach, gdzie potrzebny jest zbiór wystarczająco dobrych, ale różnych od siebie rozwiązań[1]. Wbrew pozorom, takie algorytmy są stosowane często w tandemie z podejściami gradientowymi, np. w niedawnej publikacji dotyczącej generowania obrazów [1] lub w tym tweecie. Twoim zadaniem jest implementacja bazowej formy algorytmu ewolucyjnego, który jest podstawą dla bardziej zaawansowanych podejść ewolucyjny i innych metod optymalizacji bezgradientowej. Przetestujesz implementację na tym samych problemach co w zadaniu 1, biorąc pod uwagę, że to podejście daje podobne, lub często nawet lepsze wyniki niż wcześniej.

Standardowy algorytm ewolucyjny składa się z elementów, których spodziewam się znaleźć później w Państwa implementacjach:

- funkcji dopasowania (z ang. *fitness function*): $f(\mathbf{x}) : \mathbb{R}^D \mapsto \mathbb{R}$, która dla osobnika $\mathbf{x}$ zwraca wartość dopasowania (dobranie miary zostawiam w Państwa rękach),
- operatora mutacji $\mathbf{m}(\mathbf{x}, \mathbf{p}) : \mathbb{R} \mapsto \mathbb{R}$, które z prawdopodobieństwem $p \in \mathbf{p}$ zmienia wartość odpowiadającej współrzędnej $x \in \mathbf{x}$ (prawdopodobieństwa mogą kodować wiedzę *a priori* na temat, które wartości są mniej lub bardziej prawdopodobne i można to wyznaczać metodami statystycznymi),
- operatora krzyżowania $\mathbf{c}(\mathbf{x}_1, \mathbf{x}_2, \mathbf{p}) : \mathbb{R}^{D\times 2} \mapsto \mathbb{R}^{D\times 2}$, który z prawdopodobieństwami $\mathbf{p}$ wyznacza chromosomy do krzyżowania, oraz prawdopodobieństwa krzyżowania[2].

Proszę zwrócić uwage, że tutaj zakładam zbiór liczb rzeczywistych $\mathbb{R}$, natomiast w wielu problemach wartości mogą mieć dowolny typ: całkowite, binarne, tekstowe itp.

[1] Podejścia gradientowe zawsze dążą globalnego optimum i zwracają jedno konkretne rozwiązanie dla tego samego punktu początkowego.

[2] Ze względu na potenjalną złożoność $O(n^2)$, gdzie n to liczba osobników w populacji, można założyć losowanie ze zwracaniem danemu osobnikowi innego osobnika do krzyżowania—wtedy złożoność wynosi $2n$.

Po każdej iteracji następuje selekcja, która wybiera nową populację zgodnie z funkcją dopasowania. Ponownie, można założyć, że selekcja odbywa się z prawdopodobieństwem do wartości funkcji dopasowania, jednak można w tym zadaniu założyć odcięcie z góry określonej liczby najgorszych osobników. Po każdej iteracji, rozmiar populacji powinien wynosić stałą, określoną z góry liczność.

W skrócie algorytm ewolucyjny wygląda następująco:

1. Inicjalizacja populacji,
2. Obliczenie wartości funkcji dopasowania dla każdego osobnika,
3. Powtarzaj aż do spełnienia kryterium stopu (np. określonej liczby kroków):
   (a) Wybór osobników do krzyżowania,
   (b) Krzyżowanie wybranych osobników,
   (c) Mutacja wybranych osobników,
   (d) Obliczenie wartości funkcji dopasowania dla każdego osobnika,
   (e) Wybór nowej populacji.

W czasie zajęć będę sprawdzał dokładnie czy dokładnie te elementy są zaimplementowane. Ze względu na wykorzystanie funkcji ciągłych, możecie Państwo wykorzystać bibliotekę `numpy` do operacji na macierzach.

Funkcje do przetestowania

Te same co w zadaniu 1. (poza zakresami), czyli:

- funkcja Rastrigina w zakresie $\mathbf{x} \in [-5.12, 5.12]^2, \mathbf{x} \in \mathbb{R}^2$,

$$g(\mathbf{x}) = 10d + \sum_{i=1}^{d} \left(x_i^2 - 10\cos(2\pi x_i)\right),$$

- funkcja Griewanka w zakresie $\mathbf{x} \in [-50, 50]^2, \mathbf{x} \in \mathbb{R}^2$,

$$g(\mathbf{x}) = \sum_{i=1}^{d} \frac{x_i^2}{4000} - \prod_{i=1}^{d} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$

dla d=2. W czasie zajęć będę sprawdzał, jak implementacja działa na funkcji typu *Drop-Wave*:

$$g(\mathbf{x}) = \frac{1 + \cos\left(12\sqrt{x_1^2 + x_2^2}\right)}{0.5(x_1^2 + x_2^2) + 2}, \mathbf{x} \in [-5.12, 5.12]^2.$$

Więcej informacji na jej temat można znaleźć tu.

Uwaga: Nie ograniczam Państwa do tych funkcji. Jeśli ktoś z Was ma ochotę przetestować implementację na innych funkcjach, lub bardziej skomplikowanych problemach (np. z wartościami binarnymi), to jestem otwarty na to. Sam algorytm ewolucyjny jest bardzo elastyczny i jeżeli operatory działają na wyżej wymienionych problemach, to powinny też działać na innych (napisanie generalnej implementacji dla dowolnych wartości sprzyja ćwiczeniu dobrych praktyk implementacyjnych oraz wzorców projektowych.).

W sprawozdaniu

Należy przetestować:

- Wpływ prawdopodobieństwa mutacji danego parametru na wyniki. Można założyć, że mutacja odbywa się poprzez dodanie losowej perturbacji do wartości chromosomu (współrzędnej) z rozkładu normalnego, tzn. $x^{(t+1)} = x^{(t)} + \mathcal{N}(0, \sigma^2)$ dla wybranej wartości σ,
- Wpływ prawdopodobieństwa krzyżowania dwóch osobników populacji na wyniki. Można założyć że krzyżywanie dla dwóch osobników $\{\{x_1, y_1\}, \{x_2, y_2\}\}$ może wystąpić z jednostajnym prawdopodobieństwem jako $\{\{x_2, y_1\}, \{x_1, y_2\}\} \vee \{\{y_2, y_1\}, \{x_1, x_2\}\} \ldots$ itd.,
- Wpływ rozmiaru populacji na wyniki,
- Wpływ liczby iteracji na wyniki.

Sprawozdanie powinno zawierać:

- Wykres zależności wartości funkcji dopasowania od liczby iteracji, prawdopodobieństwa mutacji, prawdopodobieństwa krzyżywania oraz rozmiaru populacji,
- Tabelę z wynikami dla tych parametrów. Proszę **nie** przeprowadzać walidacji krzyżowej— można założyć określone wartości parametrów przy badaniu pozostałego, odizolowanego parametru,
- Podsumowanie wyników.
- Kilka wybranych wizualizacji porównujących jak np. w zależności od dobranego parametru wygląda populacja końcowa (np. dla prawdopodobieństwa mutacji 0.1, 0.5, 0.9, zestawione jedna obok drugiej). Proszę nie bać się zmniejszać margines w Wordzie/Latexie w celu poprawy czytelności (raporty/publikacje zazwyczaj mają małe marginesy),
- Wnioski.

Do pomiaru dokładności algorytmu można użyć miary odległości absolutnej lub błąd średniokwadratowy.

Literatura

[1] Zhengcong Fei, Mingyuan Fan, and Junshi Huang. Gradient-free textual inversion, 2023. URL: https://dl.acm.org/doi/10.1145/3581783.3612599.