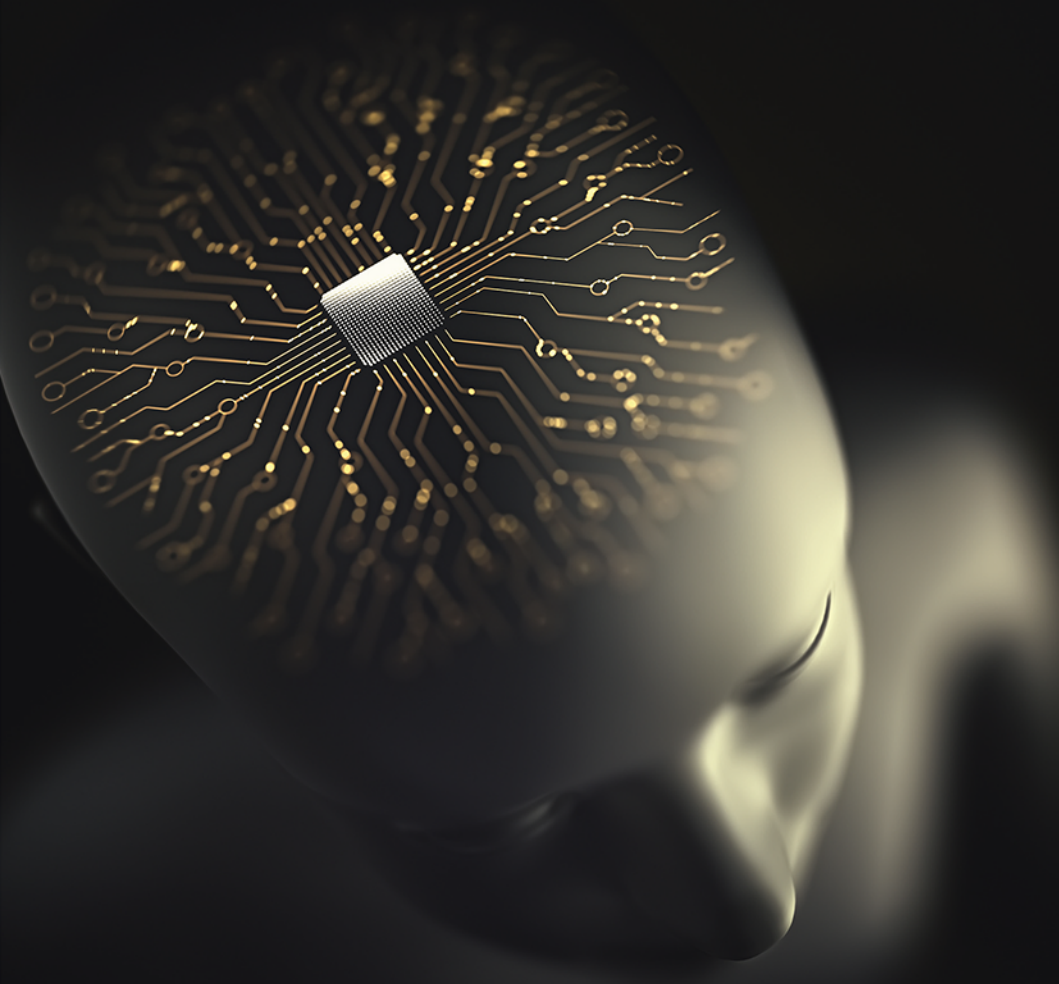




F E L I K S K U R P

SZTUCZNA INTELIGENCJA OD PODSTAW

Helion 

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiejkolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Szymon Szwaiger

Projekt okładki: Studio Gravite / Olsztyn

Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com.

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 230 98 63

e-mail: helion@helion.pl

WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/szinop>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Materiały do książki znaleźć można pod adresem:

<https://ftp.helion.pl/przyklady/szinop.zip>

ISBN: 978-83-8322-123-6

Copyright © Feliks Kurp 2023

Printed in Poland.

- Kup książkę
- Poleć książkę
- Oceń książkę

- Księgarnia internetowa
- Lubię to! » Nasza społeczność

Spis treści

ROZDZIAŁ 1. Definicja pojęcia „sztuczna inteligencja”	11
ROZDZIAŁ 2. Silna i słaba sztuczna inteligencja	13
ROZDZIAŁ 3. Przegląd klasycznych metod sztucznej inteligencji	16
3.1. Metody heurystyczne i metaheurystyczne	16
3.2. Sztuczne sieci neuronowe	17
3.3. Uczenie maszynowe	18
3.4. Przetwarzanie języka naturalnego	19
3.5. Algorytmy genetyczne i ewolucyjne	20
3.6. Algorytmy mrówkowe i inteligencja roju	20
3.7. Sztuczne życie	22
3.8. Sztuczna inteligencja w procesach wydobywania wiedzy z danych	23
3.9. Metody hybrydowe	24
3.10. Metody na pograniczu sztucznej inteligencji	24
3.11. Co dalej ze sztuczną inteligencją? Możliwości i zagrożenia	25
ROZDZIAŁ 4. Algorytmy genetyczne i ewolucyjne	27
4.1. Idea algorytmów genetycznych i ewolucyjnych	27
4.2. Klasyczny algorytm genetyczny	29
4.3. Operatory genetyczne	30
4.4. Przykłady operacji krzyżowania i mutacji	30
4.5. Przykłady wykorzystania algorytmu genetycznego	32
4.5.1. Szukanie ekstremum funkcji jednej zmiennej	32
4.5.2. Rozwiązanie problemu plecakowego	36
4.6. Strategie ewolucyjne	39
4.7. Eksploracja i eksploatacja	40
4.8. Porównanie metod selekcji	41
4.9. Metody skalowania funkcji dostosowania	42
4.10. Specjalne procedury reprodukcji	44

4.11. Programowanie genetyczne	46
4.12. Poszukiwanie ekstremum funkcji wielu zmiennych z dużą dokładnością	47

ROZDZIAŁ 5. Algorytm mrówkowy 50

5.1. Główne różnice w zachowaniu „sztucznych mrówek” w porównaniu z rzeczywistymi	50
5.2. Podstawowe parametry wejściowe algorytmu mrówkowego	51
5.3. Wpływ ilości pozostawionego feromonu w punktach grafu	53
5.4. Wpływ liczby mrówek biorących udział w eksperymencie	54
5.5. Wpływ liczby punktów do wyboru przez mrówki	55
5.6. Wpływ metody wyboru kolejnego punktu grafu przez mrówkę	55
5.7. Stopień nasycenia feromonem poszczególnych punktów grafu po zakończeniu symulacji	56
Podsumowanie	56

ROZDZIAŁ 6. Sztuczne sieci neuronowe 58

6.1. Sieci neuronowe biologiczne	58
6.2. Budowa i działanie sztucznego neuronu	59
6.3. Funkcje aktywacji	61
6.4. Perceptron	62
6.5. Model neuronu sigmoidalnego	66
6.6. Dłaczego sieci neuronowe?	66
6.7. Topologie sieci neuronowych	68
6.7.1. Sieci jednokierunkowe	68
6.7.2. Algorytm wstecznej propagacji błędów	69
6.7.3. Sieci rekurencyjne	70
6.7.4. Sieci komórkowe samoorganizujące się	73
6.7.5. Sieci samoorganizujące z konkurencją	74
6.7.6. Wykorzystanie sieci samoorganizujących	76

ROZDZIAŁ 7. Uczenie maszynowe	77
7.1. Modele uczenia maszynowego	77
7.1.1. Uczenie nadzorowane (z nauczycielem)	77
7.1.2. Uczenie nienadzorowane (bez nauczyciela)	78
7.1.3. Uczenie ze wzmocnieniem	78
7.2. Głębokie uczenie się	79
7.3. Zautomatyzowane uczenie maszynowe (AutoML)	80
ROZDZIAŁ 8. Sztuczne życie	82
8.1. Definicja sztucznego życia	82
8.2. Model Lotki-Volterry	83
8.3. Autorski model pęłaczce i bakterie	85
8.4. Symulacja choroby i leczenia organizmu	93
ROZDZIAŁ 9. Metody wykorzystujące zbiory rozmyte typu 1.	96
9.1. Podstawowe pojęcia teorii zbiorów rozmytych typu 1.	96
9.2. Operacje na zbiorach rozmytych	100
9.3. Relacje rozmyte	101
9.4. Przykłady zastosowań teorii zbiorów rozmytych	103
9.4.1. Rozmyta metoda Delphi	103
9.4.2. Rozmyta metoda PERT	103
9.5. Podejmowanie decyzji w otoczeniu rozmytym	104
9.5.1. Przydział dywidendy	105
9.5.2. Polityka zatrudnienia	105
9.6. Przybliżone wnioskowanie	105
9.6.1. Wnioskowanie w logice dwuwartościowej	105
9.6.2. Wnioskowanie w logice rozmytej	106
9.7. Sterowanie rozmyte	108
ROZDZIAŁ 10. Systemy ekspertowe. Metody wnioskowania	111
10.1. Definicja systemu ekspertowego	111
10.2. Ogólna budowa systemu ekspertowego	112

10.3. Drzewa decyzyjne	117
10.4. Metodologia wnioskowania	122
10.4.1. Wnioskowanie dedukcyjne a indukcyjne	122
10.4.2. Wnioskowanie dedukcyjne (udowodnienie celu)	125
10.4.3. Wnioskowanie indukcyjne (od danych do celu)	132
10.4.4. Wnioskowanie mieszane	137

ROZDZIAŁ 11. Inteligentna analiza danych 138

11.1. Eksploracja danych	138
11.2. Analityczne przetwarzanie danych	139
11.3. Klasyczne metody eksploracji danych	140
11.4. Inteligentne metody eksploracji danych	141
11.5. Podstawowe własności analizy skupień	144
11.6. Metoda k-średnich (k-means)	145
11.7. Przykładowa aplikacja analizy danych metodą k-means	149

ROZDZIAŁ 12. Metody hybrydowe i koewolucyjne 151

12.1. Metody hybrydowe	151
12.1.1. Algorytmy ewolucyjne w projektowaniu sieci neuronowych	151
12.1.2. Algorytmy ewolucyjne w uczeniu wag sieci neuronowych	152
12.1.3. Algorytmy ewolucyjne do uczenia wag i określania architektury sieci neuronowych jednocześnie	153
12.1.4. Adaptacyjne rozmyte algorytmy ewolucyjne	154
12.1.5. Algorytmy ewolucyjne w projektowaniu systemów rozmytych	156
12.1.6. Dopasowanie funkcji przynależności za pomocą algorytmu genetycznego	157
12.2. Algorytmy koewolucyjne	157
12.3. Algorytmy koewolucyjne — podsumowanie	160
12.4. Podsumowanie rozdziału 12.	161

ROZDZIAŁ 13. Przetwarzanie języka naturalnego 162

13.1. Języki naturalne i formalne	162
13.2. Historia rozwoju NLP	163
13.3. Poziomy analizy języka naturalnego	164
13.4. Analiza składniowa (syntaktyczna)	165
13.4.1. Gramatyki generatywne Chomsky'ego	165
13.5. Analiza semantyczna (znaczeniowa)	169
13.5.1. Podejście strukturalne do opisu semantyki	169
13.5.2. Symetryczna macierz współwystępowania słów	171
13.5.3. Reprezentacja wektorowa słowa (metody Word2vec i Doc2vec)	172
13.5.4. Podobieństwo cosinusowe wektorów słów	174
13.5.5. Modelowanie językowe BERT	175
13.6. Model GPT-3	177
13.7. Analiza sentymentu	182

Podsumowanie 185**Bibliografia 189**

ROZDZIAŁ 12. Metody hybrydowe i koewolucyjne

12.1. Metody hybrydowe

Jak to już zostało powiedziane, są to metody, które łączą w sobie (najczęściej dwie) różne metody sztucznej inteligencji. Tworzenie metod hybrydowych pokazane zostanie na pięciu przykładach, omówionych w punktach od 12.1.1 do 12.1.6.

12.1.1. Algorytmy ewolucyjne w projektowaniu sieci neuronowych

Algorytmy ewolucyjne znalazły zastosowanie w celu wspomagania niżej wymienionych problemów z zakresu sieci neuronowych:

- poszukiwanie optymalnej architektury sieci,
- uczenie wag sieci neuronowych,
- jednocześnie uczenie wag i określanie topologii sieci neuronowych.

Architektura sieci (liczba warstw, liczba neuronów w warstwie, sposób połączenia neuronów) jest zwykle tworzona metodą prób i błędów.

Optymalne **projektowanie architektury sieci neuronowych z wykorzystaniem algorytmów ewolucyjnych** traktować można jako poszukiwanie takiej struktury sieci, która działa najlepiej dla określonego zadania rozwiązywanego przez sieć. Oznacza to przeszukiwanie przestrzeni architektur i wybór najlepszej przestrzeni.

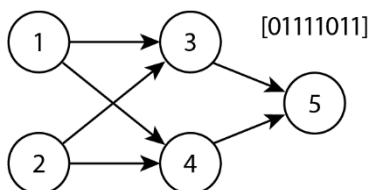
Etapy przykładowego projektowania architektury sieci neuronowej z wykorzystaniem algorytmów genetycznych:

1. Wybór sposobu reprezentowania architektury sieci w chromosomie. Jedną z wielu metod, najprostszą, jest tu **kodowanie bazujące na połączeniach**.

Rysunek 12.1 prezentuje sposób kodowania połączeń dla sieci jednokierunkowej. Kolejno numerowane neurony każdej warstwy mogą przekazywać pobudzenie do neuronów sąsiednich i kolejno numerowanych neuronów warstwy następnej. 0 oznacza brak połączenia, 1 istnienie połączenia.

RYСУNEK 12.1.

Przykład
reprezentacji
chromosomowej
dla sieci
z połączeniami
„do przodu”

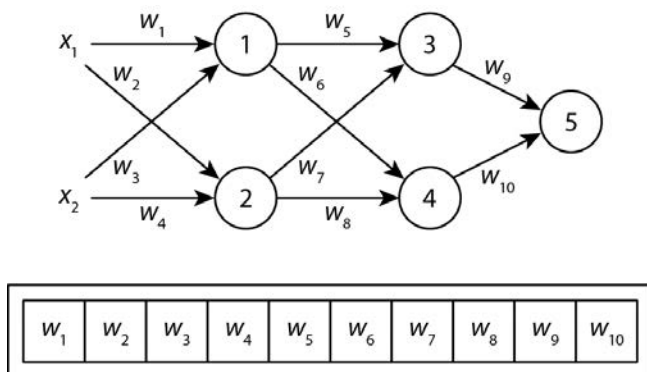


2. Dekodowanie każdego osobnika bieżącej generacji (chromosomu) do architektury wynikającej z przyjętego schematu kodowania.
3. Uczenie każdej sieci neuronowej.
4. Ocena dostosowania każdego osobnika (zakodowanej architektury) na podstawie rezultatów uczenia, np. najmniejszego średniego kwadratowego błędu uczenia.
5. Reprodukacja osobników z prawdopodobieństwem odpowiednim do ich dostosowania.
6. Zastosowanie operatorów genetycznych i otrzymanie nowej generacji.

12.1.2. Algorytmy ewolucyjne w uczeniu wag sieci neuronowych

Rysunek 12.2 pokazuje, w jaki sposób można zakodować wagi sieci neuronowej jednokierunkowej w chromosomie.

Uczenie sieci neuronowej polega na wyznaczeniu wartości wag sieci, której architektura została wcześniej ustalona. Wagi danej sieci mogą być kodowane w chromosomie za pomocą wektora liczb rzeczywistych. Każdy osobnik populacji jest określony przez całkowity wektor wag. Kolejność umieszczenia wag w chromosomie jest dowolna, ale nie może być zmieniana w trakcie uczenia.



RYСУNEK 12.2. Kodowanie wag sieci neuronowej w chromosomie dla przykładowej sieci

Ocena przystosowania osobników dokonywana jest na podstawie wartości funkcji przystosowania zdefiniowanej jako suma kwadratów błędów, będących różnicami między sygnałem wzorcowym a sygnałem wyjściowym sieci dla różnych danych wejściowych.

Po wybraniu schematu chromosomowej reprezentacji, np. tak jak pokazano na rysunku 13.2, algorytm działa wg typowego cyklu ewolucji na populacji osobników, to jest chromosomów reprezentujących sieci neuronowe o tej samej architekturze i metodzie uczenia, ale różnych wartościach początkowych wag.

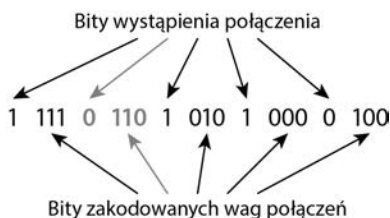
12.1.3. Algorytmy ewolucyjne do uczenia wag i określania architektury sieci neuronowych jednocześnie

Opisany w punkcie 12.1.2 proces ewolucji architektur ma wiele wad. Uczenie dla dużej populacji chromosomów wymaga bardzo dużo czasu, a jego wynik zależy od początkowego zainicjowania wag połączeń. Podobnie dobieranie wag za pomocą algorytmów genetycznych może odbywać się tylko dla jednej, określonej architektury. Wady te można wyeliminować, łącząc metody kodowania wag i struktur (opartych na kodowaniu połączeń) jednocześnie.

Poniższy rysunek 12.3 przedstawia prosty sposób realizacji tej idei. Połączenie neuronów w sieci jest określone za pomocą jednego bitu.

RYSUNEK 12.3.

Przykład
jednoczesnego
kodowania
struktury i wag
sieci neuronowej



Dodatkowo wprowadzono zakodowane binarnie wartości wag połączeń. Jeżeli jednak gen odpowiadający za połączenie jest nieaktywny (przyjmuje wartość zero), zakodowane wartości wag tego połączenia nie są brane pod uwagę przy dekodowaniu struktury.

Inny sposób kodowania informacji o wagach i połączeniach sieci jednocześnie wykorzystuje fakt, że informacja o połączeniu zawarta jest w samej wartości wagi — jeśli waga wynosi zero, to połączenie nie jest brane pod uwagę przy dekodowaniu struktury chromosomu. Metoda ta wymaga dodatkowego operatora genetycznego, który z pewnym prawdopodobieństwem usuwa bądź tworzy nowe połączenia, zerując bądź losując wartości wag połączeń.

12.1.4. Adaptacyjne rozmyte algorytmy ewolucyjne

Jednym z najpoważniejszych problemów w trakcie pracy algorytmu ewolucyjnego jest dobór właściwych proporcji między eksploatacją a eksploracją. Jak to już zostało powiedziane, równowagę osiągnąć można poprzez zwiększanie lub zmniejszanie nacisku selektywnego.

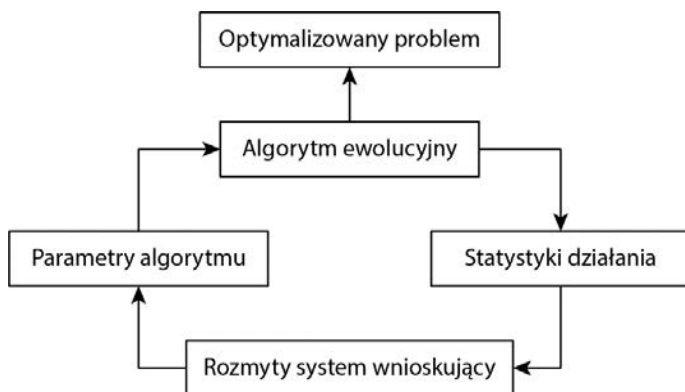
Spośród metod zmiany nacisku selektywnego wymienić należy:

- zmianę prawdopodobieństwa krzyżowania i mutacji,
- zmianę rozmiarów populacji,
- skalowanie funkcji przystosowania.

Wszystkie te metody zostały już omówione poprzednio. Potraktujmy je jako **parametry sterujące** pracą algorytmu genetycznego.

Adaptacyjne algorytmy ewolucyjne same mogą nastawiać te parametry w celu osiągnięcia jak najlepszych rezultatów. Do modyfikacji parametrów używa się **rozmytych systemów wnioskujących**.

Podczas pracy algorytmu genetycznego (patrz rysunek 12.4) generowane są różne statystyki.



RYSUNEK 12.4. Schemat ogólny adaptacyjnego algorytmu ewolucyjnego

Na podstawie ich wartości rozmyty system wnioskujący analizuje pracę algorytmu i dynamicznie dobiera wartości wybranych parametrów. Rozmyty system wnioskujący ma bazę wiedzy w formie reguł lingwistycznych, odpowiednio przygotowaną przez eksperta. Na jej podstawie dobiera parametry algorytmu. Proces ten odbywa się w sposób ciągły, sterując pracą algorytmu ewolucyjnego.

Statystyki, na podstawie których sterownik rozmyty podejmuje decyzje o zmianie parametrów, można podzielić na kilka grup:

- ocena różnorodności osobników za pomocą pewnej funkcji miary,
- badanie przystosowania fenotypów,
- stosunek liczby mutacji, które poprawiły dostosowanie, do liczby wszystkich mutacji.

Można stosować inne jeszcze statystyki.

Przykład:

Niech danymi wejściowymi do sterownika rozmytego będą trzy parametry:

- statystyka
 α = średnie przystosowanie / najlepsze przystosowanie

- statystyka
 β = *najgorsze przystosowanie / średnie przystosowanie*
- prawdopodobieństwo krzyżowania p_k ,
Wartością wyjściową będzie *wielkość populacji*.

Założmy, że wyżej wymienione cztery parametry opisane zostały za pomocą zbiorów rozmytych: *duże, średnie, małe*.

Przykładowy fragment bazy reguł sterujących doбором parametrów algorytmu może wyglądać, jak poniżej:

JEŻELI α jest *duże*, TO zwiększ populację.

JEŻELI β jest *małe*, TO zmniejsz populację.

JEŻELI p_k jest *małe* i *populacja jest mała*, TO zwiększ populację.

W tym przykładzie parametrem sterowanym będzie wielkość populacji, za pomocą której możemy bezpośrednio wpływać na stopień różnorodności osobników, czyli zwiększać lub zmniejszać nacisk selektywny.

12.1.5. Algorytmy ewolucyjne w projektowaniu systemów rozmytych

Projektowanie systemów rozmytych, podobnie jak w innych metodach sztucznej inteligencji, wymaga czasu, doświadczenia i wiedzy eksperta. Proces ten można znacznie wspomóc dzięki algorytmom ewolucyjnym, których elastyczność i niezależność można wykorzystać w rozwiązywaniu takich problemów, jak:

- dopasowanie (ang. *tuning*) funkcji przynależności do zbioru rozmytego poprzez zmianę położenia lub kształtu tej funkcji,
- generowanie odpowiednio dopasowanej bazy reguł lingwistycznych.

Interesujące jest zwłaszcza drugie rozwiązanie. Stosowane są trzy podejścia:

- **Podejście Michigan** — wykorzystywany jest tu pomysł reprezentacji jednej reguły w chromosomie i poszukiwania w pewnej populacji pożądanego zbioru reguł.

- **Podejście Pittsburgh** — jest metodą bardziej odpowiadającą działaniu algorytmu ewolucyjnego. Cała baza reguł zakodowana jest w jednym chromosomie. Poszukiwane rozwiązanie znajdujemy w najlepiej przystosowanym osobniku.
- **Uczenie iteracyjne** — łączy najlepsze cechy podejścia Michigan i Pittsburgh. Wykorzystano pomysł kodowania jednej reguły w chromosomie, ale utworzenie całej bazy odbywa się stopniowo. Ostateczną bazę reguł tworzą najlepsze osobniki, będące rezultatem działania kolejnych uruchomień algorytmu ewolucyjnego.

12.1.6. Dopasowanie funkcji przynależności za pomocą algorytmu genetycznego

Mając wybraną bazę reguł decyzyjnych systemu rozmytego, można jeszcze zwiększyć ich skuteczność poprzez odpowiednie dopasowanie funkcji przynależności użytych zbiorów rozmytych. Algorytm ewolucyjny może modyfikować funkcje przynależności poprzez zmianę położenia punktów charakterystycznych tych funkcji. Są to najczęściej współrzędne wierzchołków figur, które te funkcje opisują. Informacje o wierzchołkach kodowane są w chromosomach.

Podejście to pozwala modyfikować często używane kształty funkcji przynależności: trójkątne, trapezowe, funkcję gaussowską, sigmoidalną.

W zależności od zastosowanego algorytmu ewolucyjnego punkty charakterystyczne koduje się w chromosomie w postaci binarnej lub za pomocą liczb rzeczywistych.

12.2. Algorytmy koewolucyjne

W biologii **koewolucja** oznacza współzależną ewolucję dwóch lub większej liczby gatunków. Każdy z tych gatunków w pewien sposób wywiera nacisk na pozostałe, przyczyniając się do ich ewolucji.

Dla uproszczenia założymy, że koewolucja zachodzić będzie między dwoma gatunkami, na przykład między owadami a zapylanymi przez nie roślinami czy między drapieżnikiem a jego ofiarą. W naturze

z jednej strony następowały przystosowania drapieżników w czasie zdobywania ofiar, z drugiej — przystosowania ofiar do uniknięcia drapieżnika.

W układzie drapieżnik – ofiara w literaturze wyróżnia się następujące formy koewolucji:

- **mimikra** — przystosowanie ochronne występujące u zwierząt (zwłaszcza owadów), polegające na tym, że zwierzęta bezbronne upodabniają się do zwierząt zdolnych do obrony, przybierając ich kształt lub barwy, aby być trudnym do wykrycia przez naturalnych wrogów;
- **mimikra agresywna** — przybieranie przez drapieżcę formy atrakcyjnej dla ofiary;
- **mimetyzm** (naśladownictwo) — upodabnianie się do otoczenia w celu ukrycia się przed wrogiem lub przed potencjalną ofiarą.

W modelowaniu systemów sztucznego życia z użyciem koewolucji modelowane jest środowisko składające się z dwóch lub więcej gatunków, których ewolucja częściowo zależy od ich wzajemnych związków.

Podobnie jak w naturze, gatunki są genetycznie izolowane, tj. osobniki z dwóch różnych gatunków nie mogą się krzyżować.

Wyróżnia się dwa rodzaje algorytmów koewolucyjnych:

- Konkurujące** (ang. *competitive coevolutionary algorithms*). Prawdopodobieństwo przeżycia gatunku zależy od zachowania innych gatunków. W najprostszym scenariuszu istnieją tylko dwa gatunki, np. drapieżnik i ofiara, żywiciel i pasożyt.
- Współpracujące** (ang. *cooperative coevolutionary algorithms*). Gatunki są zachęcane do kooperacji przez nagradzanie za wspólne rozwiązywanie problemów. Karze się je natomiast za dużą samodzielność.

Przykład algorytmu konkurującego został omówiony w rozdziale 8., „Sztuczne życie”.

Ogólnie działanie algorytmów kooperacyjnych można opisać kodem przedstawionym na listingu 12.1.

LISTING 12.1. Ogólny schemat działania algorytmów koewolucyjnych

```

 $t = 0$ 
FOR EACH gatunek  $S$ 
    Zainicjalizuj  $P_t(S)$  losowymi osobnikami
FOR EACH gatunek  $S$ 
    Oblicz funkcję oceny dla każdego osobnika w  $P_t(S)$ 
WHILE warunek zakończenia  $\neq$  FALSE
BEGIN
FOR EACH gatunek  $S$ 
    BEGIN
        Wybierz osobniki do reprodukcji z populacji  $P_t(S)$ 
        Zastosuj operatory genetyczne
        Oblicz funkcję oceny
        Zastąp osobniki z  $P_t(S)$  potomkami w celu otrzymania
        populacji  $P_{t+1}(S)$ 
    END
     $t = t + 1$ 
END

```

Powyższy algorytm praktycznie nie różni się od klasycznych algorytmów ewolucyjnych. Istotnym elementem jest sposób oceniania osobników. Nie są one oceniane w odosobnieniu.

Przed dokonaniem oceny danego osobnika łączy się go z osobnikami (reprezentantami) pozostałych gatunków. Dla takiego związku ostatecznie oblicza się funkcję celu; jej wartość zostaje przypisana do osobnika, dla którego była liczona (nie do reprezentantów) — listing 12.2. Sposób, w jaki dobiera się reprezentantów, zależy od dziedziny problemu. Jedną z metod jest wybieranie najlepszych osobników w populacji. Stosowane jest również wybieranie losowe.

LISTING 12.2. Metoda oceniania osobników

```

Wybierz reprezentantów z pozostałych populacji
FOR EACH osobnik  $i$  w  $P_t(S)$ 
    BEGIN
        Stwórz związek  $i$  z reprezentantami pozostałych gatunków
        Oblicz funkcję oceny dla związku
        Przypisz wartość funkcji do  $i$ 
    END

```

12.3. Algorytmy koewolucyjne

— podsumowanie

Wydaje się, że algorytmy koewolucyjne mogą znaleźć zastosowanie w obszarach, w których klasyczne algorytmy ewolucyjne byłyby skazane na niepowodzenie.

Pierwszą grupę takich problemów stanowią problemy o dużej złożoności obliczeniowej i wyraźnej strukturalizacji. Zakładając, że podział problemu na części zostanie przeprowadzony prawidłowo, zastosowanie algorytmów współpracujących w takich przypadkach jest bardzo intuicyjne.

Poszczególne gatunki mogą koewoluować, każdy „zajmując się” własną częścią problemu. Takie podejście jest bardziej efektywne niż tradycyjne algorytmy ewolucyjne, które traktowały problem jako całość.

Ponadto dzięki stosowaniu izolacji genetycznej obliczenia wykonywane przez algorytmy współpracujące mogą być w prosty sposób rozpraszane na wiele maszyn (np. każdy gatunek na innej maszynie). Nie istnieje tu problem współdzielenia zasobów, ponieważ każdy gatunek szuka rozwiązania w lokalnej przestrzeni rozwiązań.

Ważny obszar zastosowań to **problemy optymalizacyjne**, których przestrzeń argumentów jest zwykle bardzo duża, a w szczególności nieskończona. W takich sytuacjach stosowanie algorytmów genetycznych wiąże się z bardzo długimi czasami obliczeń i algorytmy koewolucyjne powinny tu być bardzo przydatne.

Istnieją wreszcie dziedziny, dla których koewolucja jest wręcz stworzona. Są to problemy, które są z natury interaktywne, przede wszystkim gry.

Załóżmy dla przykładu, że naszym zadaniem jest znalezienie strategii gry w warcaby.

W jaki sposób dokonać pomiaru jakości strategii?

Nasuwa się tutaj zastosowanie koewolucji w postaci testu adaptacyjnego, zakładając, że strategia testująca i testowana rozwijać się będą w dwóch różnych populacjach, a celem algorytmu będzie doskonalenie każdej z nich.

12.4. Podsumowanie rozdziału 12.

Pomimo wielu zalet metod hybrydowych i algorytmów koewolucyjnych, a także ich częstej przewagi nad algorytmami sztucznej inteligencji w postaci podstawowej, są one wciąż za rzadko stosowane. Wciąż uboga jest też wiedza teoretyczna na ich temat. To, co doskonale sprawdza się w przyrodzie, nie zawsze jeszcze działa w informatyce.

Chociaż wiele zagadnień można stosunkowo łatwo zidentyfikować, np. jest oczywiste, że sposób dekompozycji problemu jest kluczowy dla działania algorytmu kooperacji, poznanie i udokumentowanie poprawnych praktyk dekomponowania zadań jest w dalszym ciągu trudne.

PROGRAM PARTNERSKI

— GRUPY HELION —

- 
1. ZAREJESTRUJ SIĘ
 2. PREZENTUJ KSIĄŻKI
 3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

SZTUCZNA INTELIGENCJA OD PODSTAW

Nie ma wątpliwości, że sztuczna inteligencja (AI) zrewolucjonizuje w najbliższych dekadach nasze życie. Wśród największych autorytetów świata nauki panuje przekonanie, że stoimy w obliczu przełomu porównywalnego z wynalezieniem i zastosowaniami elektryczności.

Sztuczna inteligencja od podstaw to pozycja, która począwszy od opisu klasycznych metod AI, takich jak algorytm genetyczny, algorytm mrówkowy, systemy ekspertowe czy sztuczne życie, zapoznaje Czytelnika z najbardziej zaawansowanymi modelami opartymi na sztucznych sieciach neuronowych. Autor skrupulatnie objaśnia złożone zagadnienia dotyczące zarówno podstaw teoretycznych, jak i budowy i zastosowań takich systemów, nie unika przy tym odwołania do historii ich rozwoju. Książka stanowi kompendium wiedzy na temat tej niesłychanie szybko rozwijającej się i dynamicznie wkraczającej w nasze życie dziedziny. Została napisana tak, aby była przystępna dla osób posiadających podstawowe umiejętności matematyczne. Może stanowić podręcznik dla studentów takich kierunków jak informatyka, mechatronika, a także automatyka i robotyka.

Dzięki książce:

- poznasz historię rozwoju sztucznej inteligencji
- zdobędziesz wiedzę na temat aktualnych metod AI, takich jak uczenie maszynowe (ML), głębokie uczenie maszynowe (DL) czy przetwarzanie języka naturalnego (NLP)
- na podstawie udostępnionych kodów źródłowych kilku autorskich aplikacji nabędziesz umiejętności w zakresie tworzenia i optymalizacji systemów sztucznej inteligencji

FELIKS KURP

Z wykształcenia jest fizykiem. Jako pracownik naukowo-badawczy uczelni medycznej zajmował się badaniem czynności bioelektrycznej mózgu; uzyskał stopień doktora nauk przyrodniczych. Aktualnie jego działalność skupia się na zagadnieniach dydaktyki i popularyzacji nauki. Jest pracownikiem naukowo-dydaktycznym Akademii Ekonomiczno-Humanistycznej w Warszawie.

Helion 



helion.pl



HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

KOD KORZYŚCI

Sięgnij po więcej! ►



ISBN 978-83-8322-123-6



9 788383 221236

Cena: 49,00 zł