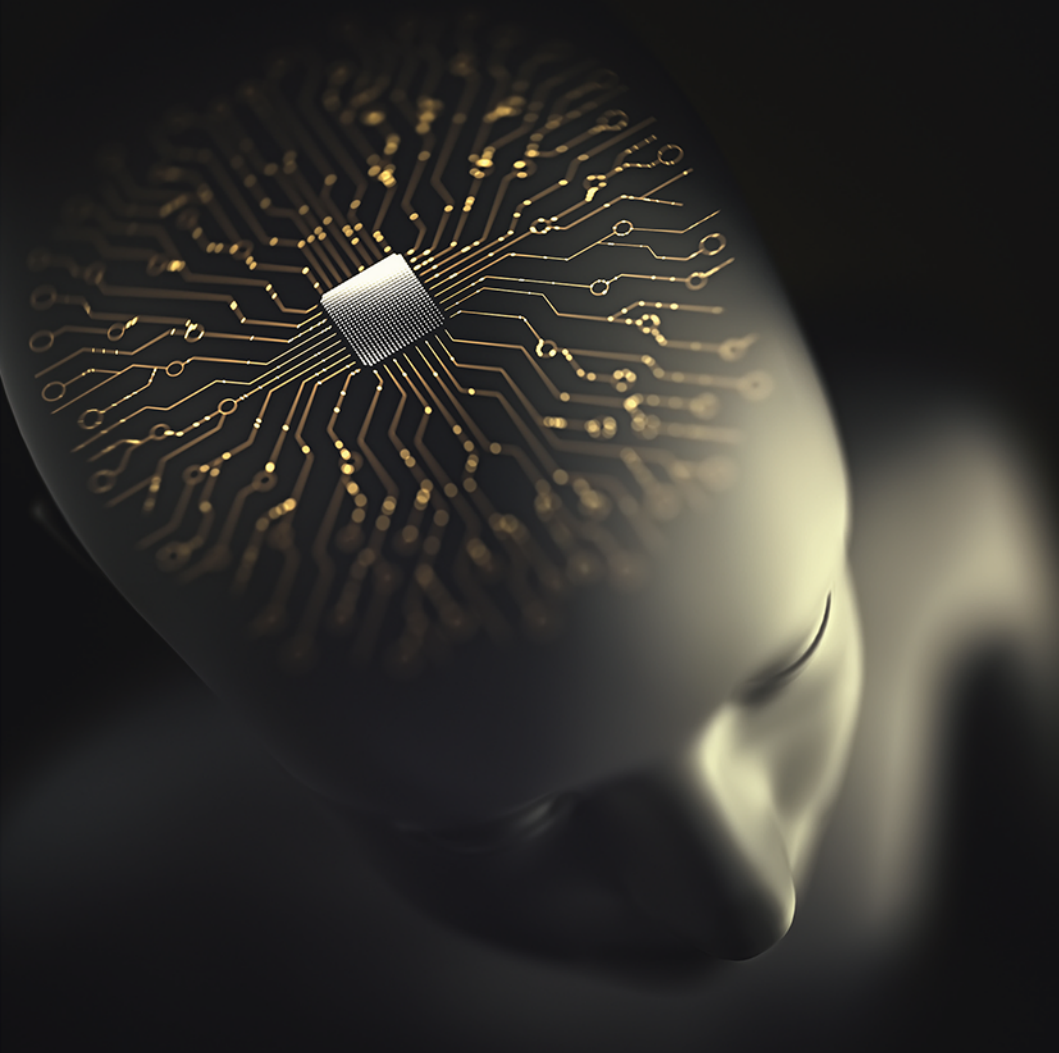


FELIKS KURP

SZTUCZNA INTELIGENCJA OD PODSTAW

Helion

Redaktor prowadzący: Szymon Szwajger

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

Printed in Poland.

Spis treści

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.

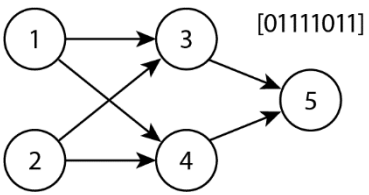
Architektura sieci (liczba warstw, liczba neuronów w warstwie, sposób połączenia neuronów) jest zwykle tworzona metodą prób i błędów.

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.



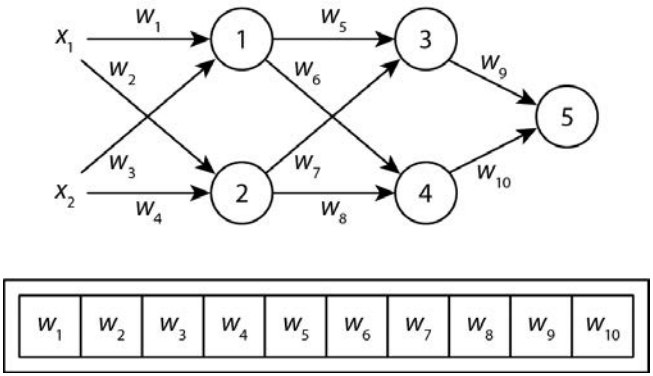
RYSUNEK 12.1. Przykład reprezentacji chromosomowej dla sieci z połączeniami „do przodu"

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. Reprodukcja 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.



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

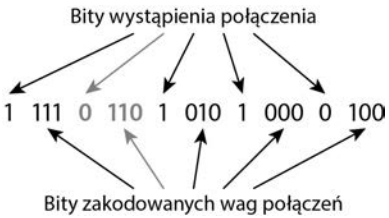
Ocena przystosowania osobników dokonywana jest na podstawie wartości funkcji przystosowania zdefiniowanej jako suma kwadratów błędów, będących różnicami między sygnałem wzorcowym a sygnałem wyjściowym sieci dla różnych danych wejściowych.

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 osiągać można poprzez zwiększanie lub zmniejszanie nacisku selektywnego.

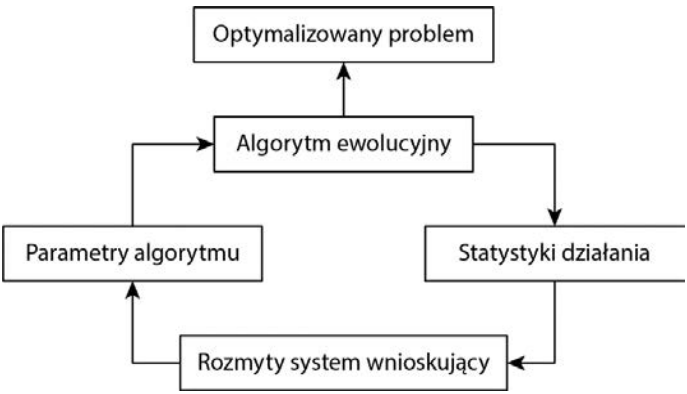
Spośród metod zmiany nacisku selektywnego wymienić należy:

- 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*.

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

Przykładowy fragment bazy reguł sterujących doborem 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:

a) **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.
b) **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 == 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