Automatyczny kelner

Projekt zaliczeniowy

13 czerwca 2016

ALEKSANDER BIŃCZYK

EWELINA BUKOWSKA

KATARZYNA KAŁEK

KAMILA MALANOWICZ

1 CEL PROJEKTU

Zadaniem zespołu była implementacja agenta - kelnera, poruszającego się w dwuwymiarowym dyskretnym środowisku (restauracja) i realizującego w sposób automatyczny przyjmowanie oraz wydawanie zamówień.

2 REPOZYTORIUM KODU

`http:\\github.com\Wewe12\pro-waiter`

3 Wykorzystane technologie

- Język programowania: Python
- Biblioteki:
  - pygame (implementacja środowiska dla agenta)
  - pyfuzzy (logika rozmyta)
  - scilearn (uczenie maszynowe)
  - numpy (regresja liniowa - przygotowanie danych wejściowych, sieć neuronowa - operacje na macierzach)
- Inne:
  - Fuzzy Control Language - język dla sterownika rozmytego
  - ANTLR3 - parser dla języka FCL

4 Wykorzystane techniki

- Reprezentacja wiedzy: zbiory rozmyte
- Metody uczenia i rozwiązywania problemów:
  - drzewo decyzyjne
  - uczenie maszynowe - regresja liniowa
  - przeszukiwanie przestrzeni stanów - A*
  - sieć neuronowa

5 Role członków zespołu

- Aleksander Bińczyk: elementy graficzne, obsługa obiektów na planszy (umiejscowanie, rozwiązywanie kolizji), podprojekt indywidualny: sieć neuronowa
- Ewelina Bukowska: stworzenie planszy, podprojekt indywidualny: uczenie maszynowe
- Katarzyna Kałek: obsługa zdarzeń (przyjmowanie/wydawanie zamówień, kontrola stanów klientów, przygotowywanie danych dla technik uczenia i rozwiązywania problemów), integracja sterownika rozmytego, podprojekt indywidualny: drzewo decyzyjne
- Kamila Malanowicz: przygotowanie sterownika rozmytego, podprojekt indywidualny: przeszukiwanie przestrzeni stanów

6 Opis działania programu

Restauracja mieści 8 stolików. Zakładamy, że każdy stolik jest zajęty przez klienta i w momencie uruchomienia programu wszyscy klienci oczekują na złożenie zamówienia. Zakończenie działania programu następuje po przyjęciu i dostarczeniu przez kelnera wszystkich zamówień. Każdy klient składa zamówienie jeden raz.

Zamówienie klienta składa się z posiłku, którego czas przygotowania jest losową liczbą sekund z przedziału [20,40], oraz - opcjonalnie - napoju. Prawdopodobieństwo zamówienia napoju przez klienta wynosi $\frac{1}{2}$. Zamówiony napój jest od razu dostępny w kuchni.

Kelner może odebrać z kuchni zamówienie tylko dla jednego klienta (posiłek, napój lub posiłek oraz napój). Po opuszczeniu kuchni obsłużony może być wyłącznie klient, którego dotyczy zamówieni.

Każdy stolik posiada unikalny numer. Numer może być wyświetlany w różnych kolorach, w zależności od tego, w jakim stanie znajduje się klient:

- kolor czarny: klient oczekuje na złożenie zamówienia,
- kolor pomarańczowy: klient złożył zamówienie i oczekuje na napój i posiłek,
- kolor czerwony: klient złożył zamówienie i oczekuje na posiłek (nie zamówił napoju bądź napój został już podany),
- kolor zielony: klient został obsłużony.

Na ekranie wyświetlane są informacje dotyczące ostatniej wykonanej akcji oraz wskazówki dla użytkownika. Jeśli w danym momencie zakończono przygotowywanie posiłku dla któregoś z klientów, informacja na ten temat wyświetlana jest zamiast ostatniej wykonanej akcji.

Naciśnięcie spacji powoduje wyświetlenie nowej wskazówki dla użytkownika. Wskazówka zawiera komunikat informujący, że kelner powinien udać się do kuchni lub konkretnego stolika.

Naciśnięcie klawisza ENTER powoduje wykonanie akcji (np. odbiór zamówienia w kuchni, obsługa stolika), jeżeli jakaś jest możliwa oraz kelner stoi w miejscu wyznaczonym przez wskazówkę.

Naciśnięcie któregoś z klawiszy 1-8 bądź K powoduje przejście kelnera do stolika o wybranym numerze bądź kuchni (odpowiednio).

7 Kryteria wyboru stolika. Reprezentacja wiedzy

Podczas podejmowania decyzji, który stolik kelner powinien obsłużyć w danym momencie, pod uwagę brane są następujące czynniki:

- czas oczekiwania:
  - jeżeli klient oczekuje na złożenie zamówienia, jest to liczba sekund, które upłynęły od początku działania programu,

  - jeżeli klient oczekuje na podanie napoju, jest to liczba sekund, które upłynęły od złożenia zamówienia.
  - 0 w przeciwnym wypadku.

- ciepłota jedzenia:
  - jeżeli klient zamówił posiłek i został on przygotowany w kuchni, ale nie jest jeszcze podany, jest to liczba sekund, które upłynęły od przygotowania posiłku,
  - 0 w przeciwnym wypadku.

- odległość:
  - jeżeli w kuchni znajduje się napój i/lub posiłek zamówiony przez klienta, jest to suma odległości (w "kafelkach") kelnera od kuchni i kuchni od stolika,
  - odległość (w "kafelkach") kelnera od stolika w przeciwnym przypadku.

Decyzje podejmowane są przy pomocy technik uczenia (więcej w kolejnym rozdziale). Zbiory treningowe dla tych technik zostały przygotowane z wykorzystaniem sterownika rozmytego. Po podaniu liczbowych danych wejściowych następuje ich rozmycie, a następnie stosowane są reguły logiczne (skrypt *fuzzycalc.fcl*), które wskazują, w jakim stopniu opłacalne jest podejście do danego stolika. Defuzyfikacja jest realizowana metodą środka ciężkości. Zwrócony zostaje wynik z przedziału (1,10), gdzie wartości bliskie 1 oznaczają największy zysk, natomiast bliskie 10 - najmniejszy.

8 METODY UCZENIA I ROZWIĄZYWANIA PROBLEMÓW

8.1 DRZEWO DECYZYJNE

Drzewo decyzyjne zostało wykorzystane w projekcie jako narzędzie wstępnej klasyfikacji. Oznacza to, że każdemu stolikowi zostaje przypisana wartość logiczna, wskazująca na to, czy obsługa tego stolika w danym momencie jest ogólnie opłacalna, czy nie. Lista stolików sklasyfikowanych jako opłacalne (True) jest przekazana do dalszej klasyfikacji przez inne techniki.

W celu zapewnienia małego rozmiaru drzewa, jego budowa odbywa się przy użyciu danych kategorycznych (dane liczbowe są rozmywane).

Atrybuty drzewa:

- waiting - odpowiada wspomnianemu w rozdziale poprzednim czasowi oczekiwania, wartości atrybutu: short, medium, long,
- meal - odpowiada ciepłocie posiłku, wartości atrybutu: absent, hot, warm,
- distance - odpowiada odległości, wartości atrybutu: close, medium, far.

Zbiór treningowy dla drzewa wygenerowany został przy użyciu sterownika rozmytego, co zostało opisane w rozdziale poprzednim. Jeżeli po defuzyfikacji dla danego zbioru wartości atrybutów została zwrócona wartość większa od 5, to zbiorowi temu przypisana jest wartość False, jeżeli mniejsza - True.

Drzewo budowane jest za pomocą algorytmu ID3. Na wybór algorytmu wpłynęły następujące czynniki:

- prostota implementacji,
- nieskomplikowana budowa zestawu treningowego,
- mała liczba atrybutów,
- dyskretne wartości atrybutów (dzięki fuzyfikacji).

8.2 Regresja liniowa

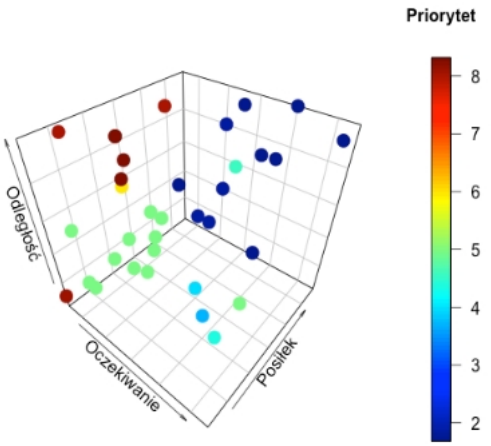
Regresja liniowa została wykorzystana do ostatecznego wyboru stolika do którego ma się udać w danym momencie kelner. Stolik jest wybierany ze zbioru tych, które drzewo decyzyjne zakwalifikowało jako opłacalne. Dane treningowe do regresji zostały wygenerowane przez wspomniane wcześniej zbiory rozmyte. Pod uwagę są brane takie sam cechy jak w przypadku drzewa decyzyjnego, tj.waiting, meal, distance. Jednak wartości tych parametrów są ciągłe i ich maksymalne wartości są ograniczone przez zbiory rozmyte. Zakres wartości dla poszczególnych cech:

- waiting : 0-45
- meal: 0-30
- distance: 0-51 Wartością przewidywaną jest priorytet z jakim dany stolik powinien zostać obsłużony. Należy on do zbiory [1,10]. Gdzie 1 to największy priorytet, a 10 najmniejszy.

Na wybór tej metody wpłynęły następujące czynniki:

- cechy, które były do dyspozycji miały charakter ciągły
- wynik regresji jest wartością ciągła - dzięki czemu można w prosty sposób ustawiać kolejność stolików
- po wstępnym wygenerowaniu zbiorów okazało się, że cechy są od siebie liniowo zależne, ostatecznie udało się osiągnąć współczynnik korelacji regresji 0.77, wynik jest dość dobry biorąc pod uwagę charakter danych.
- regresja działa również w momencie, gdy czas gry wychodzi poza zakres zbiorów rozmytych. Wynik predykcji spada wtedy poniżej zakres przewidywanego priorytetu jednak wyniki nadal są od siebie różne i można wybrać minimalną wartość. Jest to przypadek bardzo skrajny i nie pożądany w restauracji, żeby klient dostawał posiłek zimny.

Dane zostały zapisane w pliku *machinelearning_trainingset.csv* i są wczytywane do konstruktora. Poniższy wykres został wygenerowany na podstawie danych treningowych:



8.3 Algorytm A*

Algorytm heurystyczny A* został użyty w programie do przemieszczania się kelnera po planszy. Mamy możliwość udać się kelnerem na klawiaturze w dowolne dostępne pole za pomocą strzałek, ale możemy także za pomocą algorytmu przedostać się z każdego pola do odpowiedniego stolika (klawisz 1-8) bądź kuchni (klawisz k) omijając oczywiście pola na których znajdują się stoliki, goście, bądź ściany. Algorytm przyjmuje, że koszt przedostania się z jednego pola do następnego wynosi 10. Natomiast przyjęta heurystyka szacuje odległość pola od celu biorąc pod uwagę wartość bezwzględną odległości pól pomnożoną przez 10.

8.4 Sieć neuronowa

Sieć neuronowa została użyta jako alternatywny algorytm wskazywania kelnerowi następnego klienta bądź kuchni. Do nauki zastosowano regresję liniową. Sieć składa się z warstwy wejściowej, ukrytej i wyjściowej. Dane wejściowe to seria krotek długości 3, zawierających dane o tym, jak długo klient czeka na danie, jak dawno temu danie zostało przygotowane przez kuchnię (jeśli już zostało przygotowane) oraz dystans kelnera od stolika klienta. Za funkcję aktywacji posłużyła funkcja sigmoidalna. Na wyjściu sieć zwraca priorytet z jakim dany klient powinien być obsłużony. Kelner obsługuje klienta z najwyższym priorytetem w danej chwili. Dane treningowe zostały wygenerowane na podstawie przykładowych danych o posiłkach przez bibliotekę do pracy na zbiorach rozmytych.