Dokumentacja projektu

Opis rozwiązywanego problemu

Steam jest najpopularniejszą i największą platformą z grami zawierającą ponad 50 000 gier. Mimo to, jej system do szukania gier wciąż nie oferuje wystarczająco dobrego narzędzia do znajdowania podobnych gier na podstawie preferencji użytkownika. Celem naszego projektu było utworzenie programu, który pozwoli użytkownikowi wpisać nazwę lubianej przez niego gry i na podstawie dostępnych informacji wyszukać w naszej bazie danych podobne tytuły.

Szczegółowy opis zastosowanego algorytmu

Nasze rozwiązanie wykorzystuje dwa sterowniki rozmyte. Pierwszy sterownik jest aktywowany raz, przed uruchomieniem programu, i modyfikuje bazę danych o nowe parametry. Wylicza on dla każdej gry wartość „*fuzzy popularity*”. Parametry brane pod uwagę przy obliczaniu wartości to liczba pozytywnych recenzji, liczba osób posiadających grę, cena oraz długość gry.

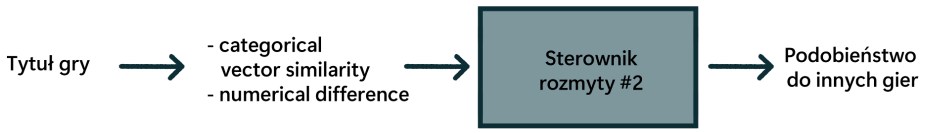
Powyższy rysunek pokazuje w uproszczony sposób jak działa algorytm obliczający wartość „*fuzzy popularity*”.

Zasady użyte w pierwszym sterowniku:

1) IF (Price IS average) OR (Game_length IS average) OR (Rating IS average) OR (Number_of_owners IS average) THEN (Popularity IS average)
2) IF (Price IS expensive) AND (Game_length IS long) AND (Rating IS positive) THEN (Popularity IS big)
3) IF (Price IS expensive) AND (Game_length IS short) THEN (Popularity IS small)
4) IF (Price IS cheap) THEN (Popularity IS big)
5) IF (Rating IS negative) THEN (Popularity IS small)
6) IF (Rating IS positive) AND (Number_of_owners IS small) THEN (Popularity IS average)

7) IF (Rating IS average) AND (Price IS cheap) THEN (Popularity IS big)

Drugi sterownik uruchamiany jest wielokrotnie podczas działania programu. Dla wpisanego przez użytkownika tytułu algorytm po kolei porównuje grę do wszystkich dostępnych tytułów w bazie danych. Podczas tej operacji obliczane są dwa parametry – „*categorical vector similarity*" oraz „*numerical difference*". „*Categorical vector similarity*" jest podobieństwem kosinusowym pomiędzy semantycznymi wektorowymi reprezentacjami wszystkich kategorii przyporządkowanych porównywanym grom w bazie danych. Wektorowa reprezentacja uzyskiwana jest poprzez sumę wektorów słów dla każdej nazwy kategorii przypisanej do danej gry, gdzie wartości pojedynczych wektorów zaczerpnięte są z pliku z pretrenowanymi wektorami uzyskanymi metodą *word2vec* udostępnianego przez firmę Google (szczegóły implementacji dostępne pod adresem: https://code.google.com/archive/p/word2vec/). Przykładowo, do pewnej gry może być przypisany następujący ciąg kategorii: "Multiplayer", "FPS", "Action", "Sci-fi". Dla każdego z tych słów odnajdywany jest odpowiadający mu wektor *word2vec* z pliku, a następnie są one wszystkie sumowane do postaci pojedynczego wektora, co daje nam ogólną semantyczną reprezentację jednocześnie charakteryzującą grę pod względem typu rozgrywki, tematyki, zawartości itp. Natomiast „*numerical difference*" porównuje wartości „*fuzzy popularity*" obliczone w pierwszym sterowniku i odejmuje je od siebie. Następnie obie te wartości wchodzą do drugiego sterownika, który skupia się na określeniu tego, czy dane gry faktycznie są do siebie podobne.



Powyższy rysunek pokazuje w uproszczony sposób jak obliczane jest podobieństwo dwóch gier (gry wybranej przez użytkownika i tytułu z bazy danych).

Zasady użyte w drugim sterowniku:

1) IF (Categorical_similarity IS average) AND (Numerical_difference IS average) THEN (Similarity IS average)

2) IF (Categorical_similarity IS small) AND (Numerical_difference IS big) THEN (Similarity IS small)

3) IF (Categorical_similarity IS big) AND (Numerical_difference IS small) THEN (Similarity IS big)

Opis wykonanej implementacji

Program został napisany w języku Python, przy użyciu biblioteki Simpful. Reprezentacja wektorowa danych kategorialnych została utworzona na podstawie pretrenowanych wektorów *word2vec*. Kod jest dostępny w repozytorium: https://git.wmi.amu.edu.pl/s449288/fuzzy-game-recommender.