Politechnika Wrocławska

Wydział Informatyki i Zarządzania
kierunek studiów: Informatyka
specjalność: Projektowanie Systemów Informatycznych

Praca dyplomowa - magisterska

TITLE
TITLE EN

Katatzyna Biernat

słowa kluczowe:
KEYWORDS

krótkie streszczenie:
SHORT ABSTRACT

Promotor:	dr inż. Bernadetta Maleszka	.....	.....
	imię i nazwisko	*ocena*	*podpis*

*Do celów archiwalnych pracę dyplomową zakwalifikowano do:**

a) kategorii A (akta wieczyste)

b) kategorii BE 50 (po 50 latach podlegające ekspertyzie)

** niepotrzebne skreślić*

pieczątka wydziałowa

Wrocław 2016

Niniejszy dokument został złożony w systemie LaTeX.

Spis treści

Streszczenie

ABSTRACT PL

Abstract

ABSTRACT EN

Todo list

Rozdział 1

Cel pracy

Celem pracy jest zaproponowanie i zbudowanie hybrydowego algorytmu rekomendacji. Składowymi docelowego algorytmu są metody kolaboratywnego filtrowania oraz metody filtrowania z analizą treści.

Rozdział 2

Wstęp

Wraz z rozwojem Internetu zmienił się sposób dostępu do informacji. Kiedyś to użytkownik musiał walczyć pozyskanie wiedzy; dzisiaj to informacje walczą u uwagę użytkowników. W świecie zalanym wiadomościami koniecznym wydaje się być zastosowanie filtra, który odsieje interesującą i wartościową zawartość od tej niechcianej. Tak też z pomocą przychodzą zautomatyzowane mechanizmy rekomendacji.

Jednakże sama idea rekomendacji nie jest niczym nowym. Co więcej, zjawisko to możemy zaobserwować w naturze – na przykład wśród mrówek, które podążają wyznaczoną (rekomendowaną) ścieżką feromonową w poszukiwaniu pożywienia.

Ludzie od niepamiętnych czasów posiłkowali się opiniami innych aby ułatwić sobie dokonanie wyboru, od najbliższego grona znajomych do ekspertów i autorytetów.

Wraz z rozwojem nauk informatycznych problem rekomendacji stał się problemem interesującym badaczy. Za pierwszy system rekomendacji uznaje się *Tapestry* stworzony w laboratoriach Xerox Palo Alto Research Center w 1992 roku. Motywacją było odfiltrowanie rosnącej liczby niechcianej poczty elektronicznej [16].

Wkrótce później idea ta została rozszerzona przez takich graczy jak Amazon, Google, Pandora, Netflix, Youtube, Yahoo etc. aż do formy, jaką znamy dzisiaj: systemu, który sugeruje użytkownikom produkty, filmy, muzykę, strony internetowe na podstawie ich aktywności w sieci [47].

Wielkie koncerny internetowe stale poprawiają jakość swoich algorytmów rekomendacji. Najlepszym przykładem jest tutaj Netflix, który w październiku 2006 zorganizował ogólnodostępny konkurs na najlepszy algorytm. Zadaniem uczestników było ulepszenie algorytmu Cinematch. Już po siedmiu dniach od ogłoszenia konkursu trzy zespoły zdołały przebić Cinematch o 1.06% [33][35]. 18 września 2009 Netflix ogłosił, że zespół BellKor's Pragmatic Chaos poprawił Cinematch o 10,06% osiągając wynik $RMSE = 0.8567$. Tym samym wygrał nagrodę w wysokości $1,000,000 i zakończył konkurs [34][36].

Systemy rekomendacji ulepszane są nieustannie, o czym świadczy chociażby organizowana rokrocznie konferencja *ACM International Conference on Recommender Systems*. Tematyka ta poruszana jest także na konferencjach *European Conference on Information Retrieval*, *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* i wielu innych. Mimo dużego stopnia

zaawansowania wciąż istnieje pole manewru do ulepszania algorytmów rekomendacji i co za tym idzie zwiększanie zadowolenia użytkowników, które z kolei prowadzi do osiągania korzyści biznesowych.

Rozdział 3

Przegląd istniejących rozwiązań

Tradycyjnie wyróżniamy następujące techniki rekomendacji:

- **filtrowanie w oparciu o zawartość** (eng. content-based), technika koncentrująca się na atrybutach elementów. Użytkownikowi rekomendowane są elementy, które podobne są do tych wybieranych przez niego w przeszłości;
- **filtrowanie kolaboratywne** (eng. collaborative filtering), technika polegająca na odnajdywaniu użytkowników o podobnych gustach i sugerowaniu lubianych przez nich elementów aktualnie aktywnemu użytkownikowi;
- **filtrowanie demograficzne** (eng. demographic), technika koncentrująca się na sugerowaniu aktywnemu użytkownikowi elementów popularnych pośród użytkowników z tej samej okolicy bądź w podobnym przedziale wiekowym;
- **filtrowanie z analizą domeny wiedzy** (eng. knowledge-based), technika dobierająca kolejne elementy na podstawie określonej domeny wiedzy na temat tego, jak dany element spełnia potrzeby i preferencje użytkownika;
- **filtrowanie z analizą społecznościową** (eng. community-based), technika dobierająca rekomendacje dla użytkownika w zależności od preferencji innych użytkowników z jego sieci społecznościowej. W myśl zasady "powiedz mi kim są twoi przyjaciele a powiem ci kim jesteś";
- **hybrydowe systemy rekomendacji**, to kombinacja dowolnych powyższych technik.

Każda z tych technik ma swoje wady i zalety w zależności od kontekstu, w którym ma być stosowana[40].

3.1. Filtrowanie w oparciu o zawartość

Filtrowanie content-based opiera się na cechach elementów w systemie. Rekomendowane są obiekty, które podobne są do tych pozytywnie ocenionych wcześniej przez użytkownika[17]. W zależności od domeny pod uwagę mogą być brane słowa kluczowe, cechy takie jak rok wydania, reżyser, autor, kompozytor, gatunek itp.

3.1.1. Metody tworzenia profilu użytkownika

Profil użytkownika może być tworzony na dwa sposoby. Jeżeli użytkownik jawnie pozostawia informacje można mówić o podejściu aktywnym (explicit feedback). Do takich informacji należą: ocena konkretnych elementów, tzw. łapka w górę lub w dół, komentarz itp.

Jednakże nawet jeżeli użytkownik nie jest skory do zostawiania tego typu śladów, to i tak można wiele na jego temat wywnioskować korzystając z podejścia pasywnego (implicit feedback). System bierze wówczas pod uwagę aktywność użytkownika taką jak: historia zakupów, historia przeglądarki a nawet ruchy myszką. W przypadku serwisu z muzyką czy filmem cenną informacją będzie fakt, czy użytkownik wysłuchał lub obejrzał dany materiał do końca czy też wyłączył go po paru sekundach. [29][24]

3.1.2. Zalety podejścia content-based

Do zalet filtrowania w oparciu o zawartość należy niezależność użytkownika. Podczas budowania rekomendacji brany pod uwagę jest tylko jego profil; aktywność innych aktorów w systemie nie wpływa na wynik końcowy. Inną przewagą jest przejrzystość – każda propozycja jest w pełni uzasadniona, gdyż opiera się na działaniach użytkownika w przeszłości (podczas gdy w przypadku filtrowania kolaboratywnego mamy do czynienia z czarną skrzynką). Ponadto, tego typu algorytm ma możliwość zaproponowania elementu, który nie był nigdy wcześniej oceniany przez nikogo. Zapobiega to zjawisku długiego ogona [29].

3.1.3. Najczęściej spotykane problemy

Aby rekomendacja była skuteczna użytkownik powinien ocenić jak najwięcej elementów. Problematyczni są zatem użytkownicy, którzy dopiero co dołączyli do serwisu oraz tacy, którzy nie są aktywni i rzadko zostawiają po sobie ślad [30].

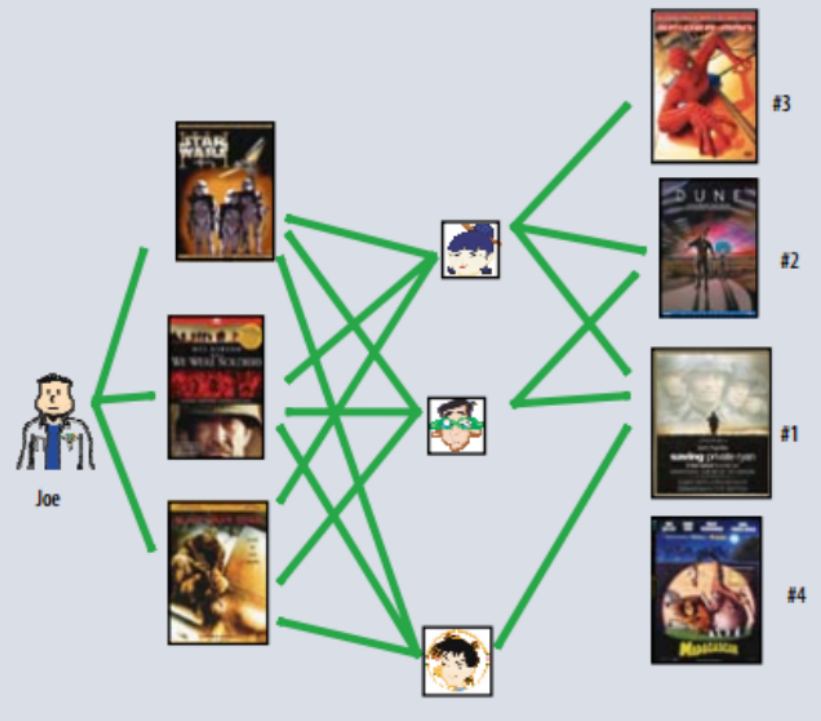
Podejście content-based jest podatne na pułapkę tzw. bańki informacyjnej. Jeżeli w systemie rekomendującym produkcje kinowe użytkownik do tej pory oceniał jedynie filmy akcji, to mało prawdopodobne jest, że algorytm zaproponuje mu ciekawy dramat obyczajowy. Nowe propozycje nie są zaskakujące[29].

3.2. Filtrowanie kolaboratywne

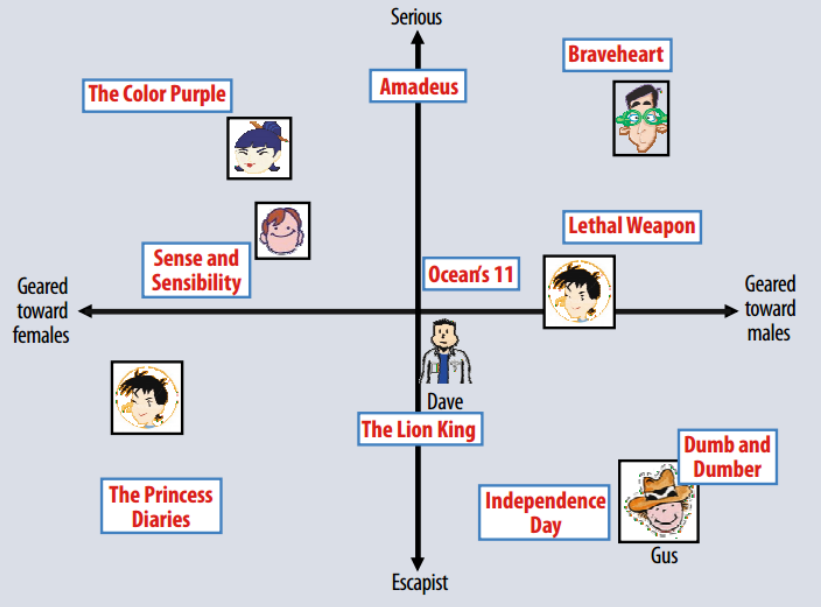
Filtrowanie kolaboratywne opiera się o założenie, że ludzie o zbliżonym guście dokonują podobnych wyborów. Użytkownicy o zbliżonym guście to osoby, które oceniły konkretne elementy w podobny sposób[40][46][17].

W przypadku filtrowania kolaboratywnego można wyróżnić dwa główne podejścia: oparte o regułę sąsiedztwa (ang. *neighborhood*) oraz oparte o model (ang. *model-based*), wykorzystujące modele ukrytych parametrów[25][24].

Filtrowanie w oparciu o regułę sąsiedztwa koncentruje się na związkach element-element bądź użytkownik-użytkownik[24]. Rysunek 3.1 pokazuje regułę sąsiedztwa skoncentrowaną na relacji użytkownik-użytkownik. Joe ocenił trzy filmy. System odnajduje innych użytkowników, którzy ocenili te trzy pozycje podobnie jak Joe. Każdy z nich pozytywnie ocenił film „Saving Private Ryan", zatem jest to pierwsza rekomendacja dla Joe.



Rys. 3.1: Filtrowanie kolaboratywne metodą sąsiedztwa, zorientowane na użytkownika[25].



Rys. 3.2: Filtrowanie kolaboratywne z wykorzystaniem modelu ukrytych parametrów[25].

Ideą podejścia model-based jest zbadanie i modelowanie zależności element-użytkownik wraz z czynnikami reprezentującymi ukryte własności elementów i użytkowników. Taki

model jest następnie uczony przy użyciu dostępnych danych. W rezultacie można z niego odczytać przewidywaną ocenę elementu dla konkretnego użytkownika[8][24].

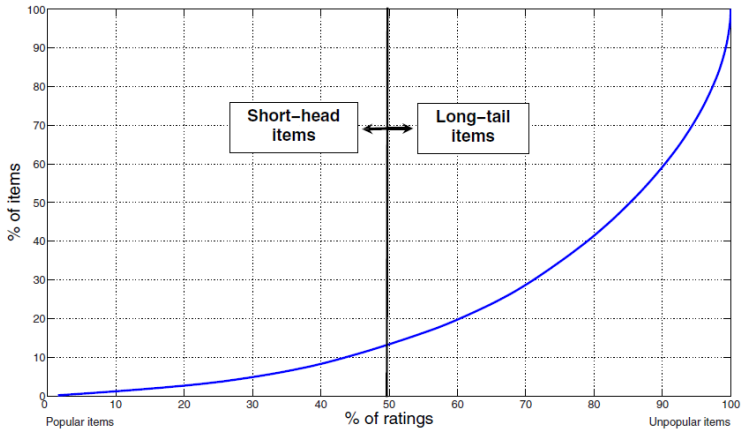
Rysunek 3.2 pokazuje w sposób uproszczony podejście oparte o model. W układzie współrzędnym oznaczeni są użytkownicy wedle swoich preferencji oraz konkretnych cech (np. płeć) a także filmy, które stanowią odpowiedź na dany zestaw preferencji/cech [25].

3.2.1. Najczęściej spotykane problemy

Jednym z problemów klasycznego podejścia do kolaboratywnego filtrowania jest brak uwzględnienia dynamiki zmian w gustach użytkowników. Ten sam użytkownik na przestrzeni kilku lat lub miesięcy może zupełnie inaczej ocenić ten sam film bądź piosenkę. Rozwiązaniem jest dodanie czynnika czasu podczas obliczania wag kolejnych ocen. [6][20][25].

Innym problemem jest tzw. zimny start (eng. cold start). Polega on na tym, że użytkownicy nowi w systemie ocenili zbyt mało elementów, aby można było zbudować dla nich dobre rekomendacje[50][44].

Powszechnym zjawiskiem jest tzw. efekt długiego ogona. Rysunek 3.3 przedstawia jak rozkłada się procentowa ilość ocen danych elementów w zależności od ich popularności. Jeżeli algorytm rekomendacji nie wspiera mniej popularnych elementów, to istnieje ryzyko, że użytkownicy nie otrzymają możliwości eksplorowania nowych, niszowych materiałów[44][5].



Rys. 3.3: Problem długiego ogona: 50% ocen dotyczy 10-12% najpopularniejszych elementów w systemie[44].

Systemy rekomendacji wykorzystujące filtrowanie kolaboratywne nie są skalowalne. Złożoność rośnie proporcjonalnie do ilości użytkowników i elementów. Wielkie koncerny internetowe takie jak Twitter wykorzystają klastry i maszyny z bardzo dużą ilością pamięci aby zachować płynność działania serwisu [12].

3.3. Popularne serwisy wykorzystujące algorytmy rekomendacji

W przeciągu ostatnich lat algorytmy rekomendacji zagościły na bardzo wielu popularnych serwisach internetowych z różnych domen. Poniższa lista prezentuje garstkę wybranych stron.

3.3.1. Rekomendacja muzyki

- **YouTube** – serwis powstały w 2005 roku, pozwalający na bezpłatne umieszczanie, odtwarzanie, ocenianie i komentowanie filmów. Od 2006 roku przejęty przez Google. YouTube buduje profil użytkownika w oparciu o jego aktywność w serwisie. Brane pod uwagę są polubienia (łapka w górę), subskrypcje, udostępnianie a także informacje czy użytkownik obejrzał film do końca czy tylko pewien jego procent. Techniki rekomendacji stosowane przez serwis to przede wszystkim asocjacyjna eksploracja danych i licznik wspólnych odwiedzin danego wideo w czasie trwania pojedynczej sesji [7].
- **LastFM** – internetowa radiostacja oferująca rozbudowany mechanizm rekomendacji piosenek "Audioscrobbler".
- **Pandora** – spersonalizowane radio internetowe wykorzystujące projekt Music Genome Project. Każda piosenka przeanalizowana jest pod kątem maksymalnie 450 cech; na tej podstawie budowane są rekomendacje[1].

3.3.2. Rekomendacja filmów

- **Netflix** – amerykańska platforma oferująca strumieniowanie filmów i seriali. Działający od 2007 roku gigant oferuje rozbudowany system rekomendacji Cinematch[39].
- **Filmweb** – polski serwis poświęcony filmom i jednocześnie druga największa baza filmowa na świecie. Oferuje system rekomendacji Gustomierz, który umożliwia poznawanie nowych filmów w guście użytkownika[9].
- **Internet Movie Database (IMDb)** – największa internetowa baza filmów. Baza zawiera $3,837,014$ pozycji, które są oceniane w skali od 1 do 10 przez użytkowników[19].

3.3.3. Platformy typu e-commerce

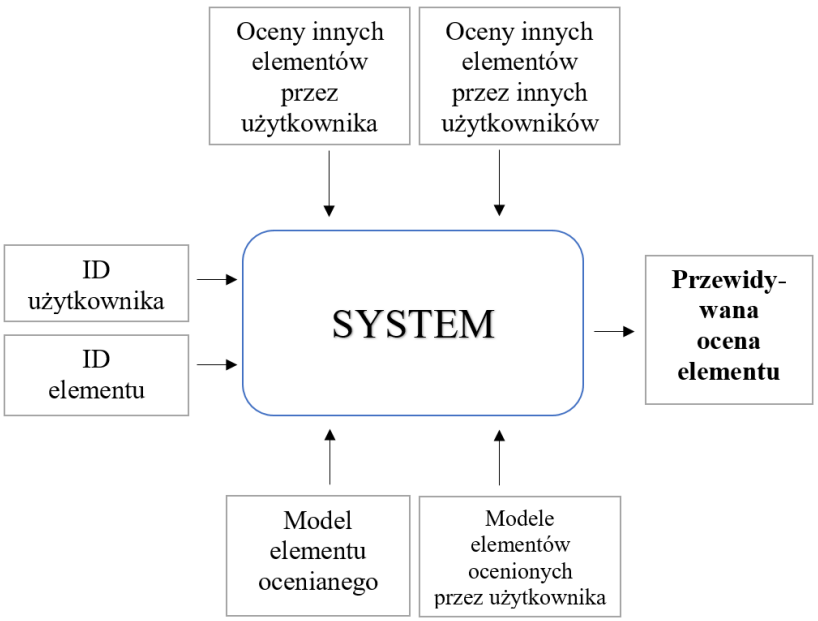
- **Allegro** – polski portal aukcyjny. Swoim użytkownikom oferuje panel rekomendacji. Prezentowane produkty wybierane są w oparciu o to co dotychczas kupował i oglądał użytkownik[2].
- **Amazon** – największy na świecie sklep internetowy typu B2C. Amazon w swoich mechanizmach rekomendacji wykorzystuje algorytmy filtrowania kolaboratywnego typu item-to-item[28].

Rozdział 4

Model systemu

Głównym założeniem systemu zaproponowanego przez autorkę jest połączenie zalet kolaboratywnego filtrowania i filtrowania w oparciu zawartość minimalizując jednocześnie wady obu podejść.

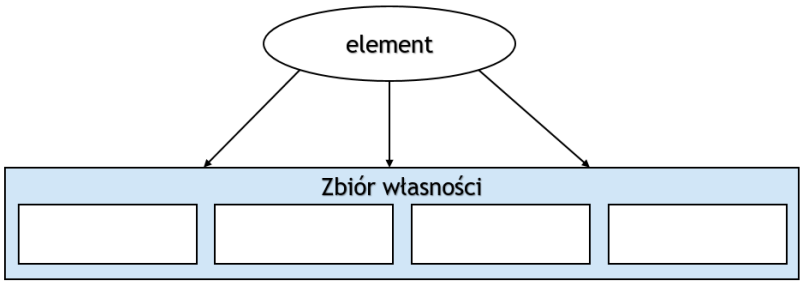
Rys. 4.1 przedstawia czarnoskrzynkowy model systemu. Danymi wejściowymi są numery identyfikacyjne użytkownika dla którego ma być zbudowana rekomendacja oraz elementu, dla którego ma być przewidziana ocena. System pobiera model elementu a także modele wszystkich innych elementów, które użytkownik ocenił w przyszłości. Jednocześnie pobierane są informacje o tym jak użytkownicy systemu ocenili inne elementy. Wynikiem wyjściowym jest predykcja – jak aktywny użytkownik oceni element.



Rys. 4.1: Model czarnoskrzynkowy

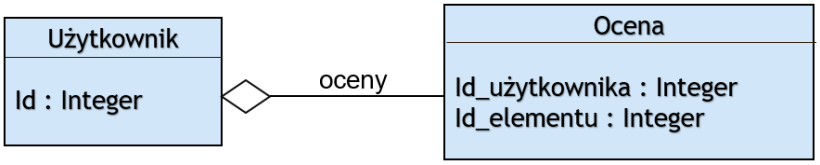
Założeniem systemu jest uniwersalność, zatem model elementu jest uogólniony i dostosowuje się w zależności do domeny, w której system jest wykorzystywany. Rys.

4.2 przedstawia reprezentację elementu w systemie. W zbiorze wartości mogą znaleźć się takie pozycje jak lista aktorów, reżyser (w przypadku filmów), gatunek, wykonawca (w przypadku muzyki), typ produktu lub cena (w przypadku systemów typu e-commerce).



Rys. 4.2: Uogólniony model elementu

Każdy użytkownik systemu jest anonimowy. Nie jest znana jego płeć, wiek, pochodzenie itp. System nie przechowuje także informacji właściwych mediom społecznościowym takich jak relacje między użytkownikami (przyjaźnie, śledzenie). Wiadomym jest jedynie jakie elementy zostały ocenione i jak zostały ocenione. Rys. 4.3 przedstawia reprezentację użytkownika w systemie.



Rys. 4.3: Uogólniony model elementu

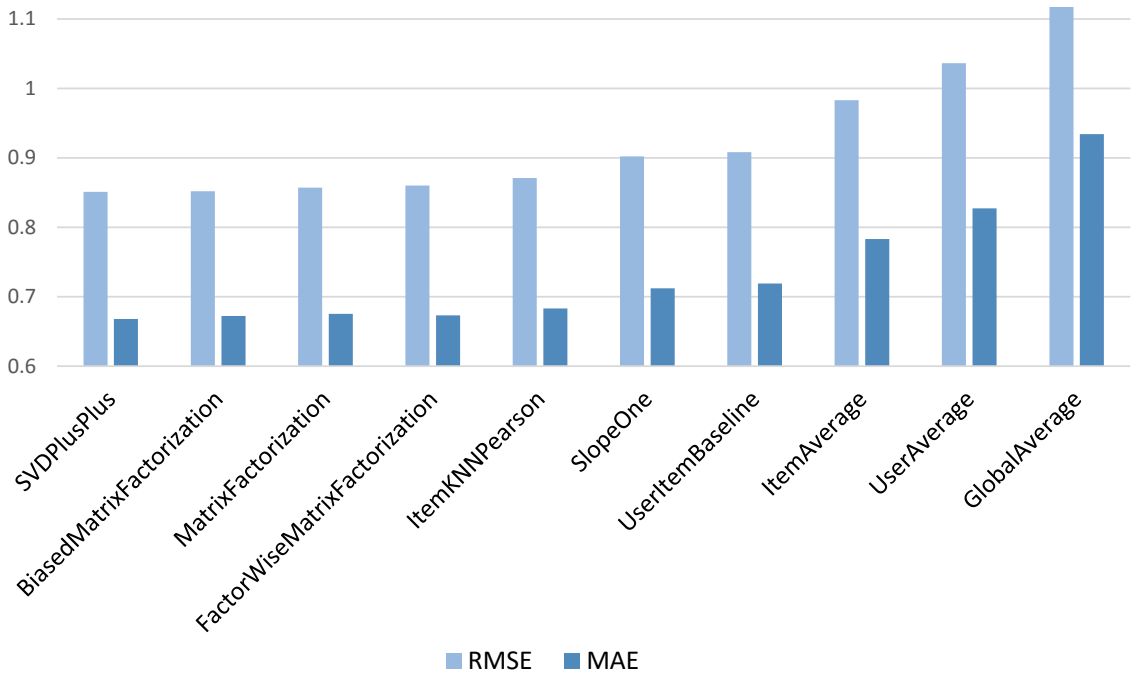
Rozdział 5

Algorytmy

5.1. Filtrowanie kolaboratywne

Implementacja algorytmów collaborative-filtering, które wykorzystane zostały w projekcie pochodzą biblioteki MyMediaLite [10][11].

Przed podjęciem decyzji dotyczącej wyboru algorytmu kolaboratywnego filtrowania zostały przeanalizowane testy na bazie MovieLens M1[13]. Testy przeprowadzone zostały z pięciokrotną walidacją krzyżową. Rys. 5.1 przedstawia wyniki (im mniejsza wartość RMSE i MAE tym lepiej).



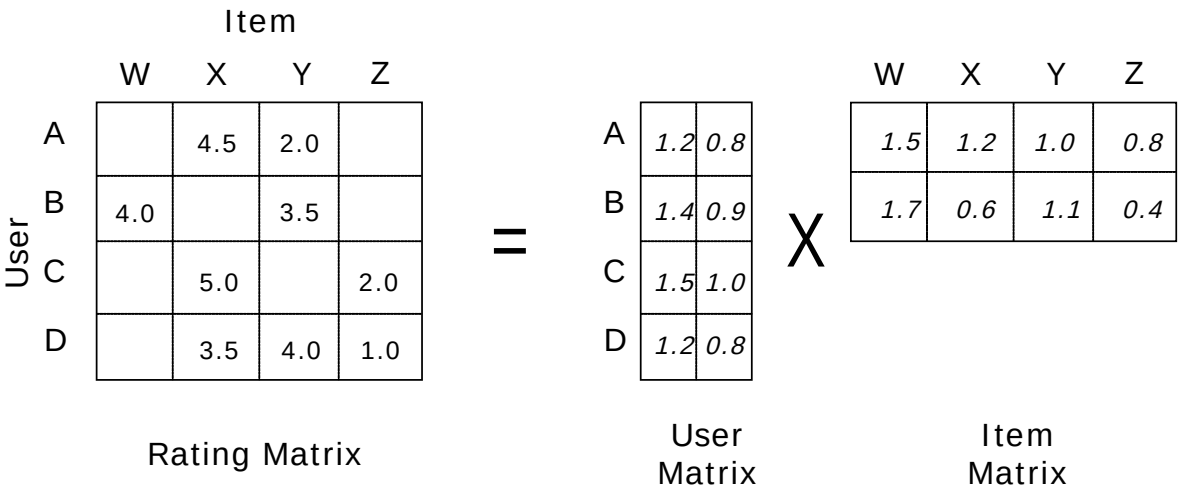
Rys. 5.1: Test algorytmów filtrowania kolaboratywnego [32]

Najefektywniejsze okazały się algorytmy SVD++, Biased Matrix Factorization i Matrix Factorization bazujące na podejściu model-based.

5.1.1. Matrix Factorization

Metoda matematyczna

Algorytm *Matrix Factorization* bazuje na matematycznej metodzie rozkładu macierzy na czynniki.



Rys. 5.2: Faktoryzacja macierzy [43]

Początkowo dana jest niekompletna macierz zawierająca oceny, jakie użytkownicy wystawili konkretnym elementom (*Rating Matrix*). Celem metody jest odnalezienie wartości, jakie można wstawić w puste miejsca, czyli przewidzenie jaką ocenę dany użytkownik wystawi nieocenionemu jeszcze elementowi.

W tym celu tworzone są odrębne macierze dla użytkowników i elementów zawierające ukryte własności. Każdy element powiązany jest z wektorem $q_i \in \mathbb{R}^f$ (wektory W, X, Y, Z na rys. 5.2) a każdy użytkownik z wektorem $p_u \in \mathbb{R}^f$ (wektory A, B, C, D na rys. 5.2). Wartości czynników ukrytych determinują stopień zainteresowania daną cechą (w przypadku macierzy użytkowników) bądź stopień, w jakim dany element posiada tę cechę (w przypadku macierzy elementów).

Iloczyn skalarny $q_i^T p_u$ przedstawia relację pomiędzy użytkownikiem a elementem. Na tej podstawie można wnioskować ocenę r_{ui}, jaką użytkownik może wystawić elementowi: $r_{ui} = q_i^T p_u$ i w konsekwencji estymować zainteresowanie użytkownika danym elementem [25]..

Głównym wyzwaniem jest odnalezienie wartości macierzy użytkownika i elementu, które po przemnożeniu przez siebie dadzą kompletną macierz ocen. W przypadku omawianych algorytmów stosowana jest metoda stochastycznego gradientu prostego.

Stochastyczny gradient prosty

Stochastyczny gradient prosty (ang. *stochastic gradient descent*, *SGD*) jest iteracyjnym algorytmem optymalizacyjnym mającym za zadanie odnalezienie minimum bądź maksimum zadanej funkcji. Jest on uproszczeniem popularnej metody gradientu prostego [4].

Dana jest funkcja celu w postaci

$$Q(w) = \sum_{i=1}^{n} Q_i(w)\,. \tag{5.1}$$

Zadaniem algorytmu jest odnalezienie takiej wartości parametru w, dla którego $Q(w)$ będzie minimalne. Wykorzystując klasyczną metodę gradientu prostego poszukiwania można zapisać następującym wzorem:

$$w := w - \eta \sum_{i=1}^{n} \nabla Q_i(w)\,, \tag{5.2}$$

gdzie η symbolizuje współczynnik uczenia (ang. *learning rate*). W przypadku algorytmu stochastycznego następuje uproszczenie. Zamiast obliczać dokładny gradient $Q(w)$, w każdej iteracji jest on aproksymowany na podstawie pojedynczego, losowo wybranego przypadku:

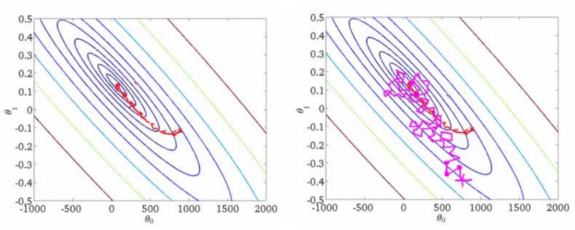
$$w := w - \eta \nabla Q_i(w)\,. \tag{5.3}$$

Algorytm przetwarza wszystkie elementy zbioru treningowego i dla każdego przypadku wykonuje aktualizację wartości w. Przebieg algorytmu przedstawiony jest poniżej (algorytm 1).

Algorytm 1 Stochastyczny gradient prosty

```
w ← początkowy wektor w
η ← współczynnik uczenia
while nie odnaleziono minimum do
  losowo mieszkaj przykłady ze zbioru testowego
  for i ∈ {1, 2, ..., n} do
    w := w − η∇Q_i(w)
  end for
end while
```

Stochastyczny gradient prosty wykonuje dużo więcej kroków niż jego klasyczny odpowiednik aby odnaleźć rozwiązanie optymalne. Mimo tego jest on szybszy, gdyż każdy pojedynczy krok jest mniej kosztowny niż w oryginale. Różnicę w działaniu przedstawia rys. 5.3.



Rys. 5.3: Różnica pomiędzy klasycznym gradientem prostym (po lewej) a jego stochastyczną wersją (po prawej) [3]

Przebieg algorytmu Matrix Factorization

Przebieg algorytmu składa się z dwóch faz. W pierwszej fazie inicjowany jest model (zob. algorytm 2). Daną wejściową jest macierz zawierająca dotychczasowe oceny elementów przez użytkowników w systemie. Na wyjściu otrzymywane są dwie nowe macierze reprezentujące ukryte własności użytkowników i elementów. Na tym etapie są one wypełnione wartościami losowymi.

Algorytm 2 Matrix Factorization – Inicjacja modelu

$N \leftarrow$ liczba użytkowników
$M \leftarrow$ liczba elementów
$F \leftarrow$ liczba ukrytych własności
$ratings \leftarrow$ Macierz NxM zawierająca dotychczasowe oceny wszystkich elementów przez wszystkich użytkowników
$user\_factors \leftarrow$ Macierz NxF reprezentująca ukryte własności użytkowników
$item\_factors \leftarrow$ Macierz MxF reprezentująca ukryte własności elementów
for each $uf \in user\_factors$, $if \in item\_factors$ **do**
  wstaw losową wartość za pomocą transformacji Boxa-Mullera
end for
for each $user, item \in ratings$ **do**
  if $ratings_{user,item} = NULL$ **then**
    wstaw 0 do wiersza $user\_factors_{user}$ i $item\_factors_{item}$
  end if
end for
return $user\_factors, item\_factors$

W fazie drugiej następuje uczenie metodą stochastycznego gradientu prostego (zob. algorytm 3). Wynikiem tej fazy są macierze reprezentujące ukryte własności użytkowników i elementów. Mnożąc je ze sobą uzyskiwana jest przewidywana ocena każdego z elementów przez użytkowników.

Przed rozpoczęciem uczenia ustalane są parametry: parametr regulujący (ang. *regularization*), współczynnik uczenia (ang. *learning rate*), parametr zanikania (ang. *decay*) i liczba iteracji. W trakcie trwania głównej pętli parametr regulujący pozostaje niezmienny. Służy on zapobieganiu zjawisku nadmiernego dopasowania (ang. *overfitting*). Tempo uczenia jest przy każdym przebiegu pętli mnożone przez parametr zanikania, dzięki czemu można kontrolować w jakim stopniu kolejne przebiegi pętli wpływają na finalny wynik.

Ostatnim parametrem ustalanym przed główną pętlą jest skośność globalna (ang. *global bias*), która jest średnią wszystkich znanych ocen.

W pętli uczenia wykonywane są następujące operacje: dla każdej pary użytkownik – element budowana jest przewidywana ocena poprzez obliczenie iloczynu skalarnego odpowiednich wartości z macierzy wartości ukrytych. Ocena ta jest modyfikowana poprzez dodanie globalnej skośności a następnie porównywana z faktyczną oceną elementu przez użytkownika. Tak uzyskany błąd służy do wyliczenia delty. Macierze wartości ukrytych uaktualniane są o wyliczoną deltę.

Pod koniec każdej iteracji aktualizowany jest współczynnik uczenia.

Algorytm 3 Matrix Factorization – Faza uczenia

$global\_bias \leftarrow$ średnia wszystkich ocen
$X \leftarrow$ liczba iteracji
$regularization \leftarrow$ parametr regulujący
$current\_learnrate \leftarrow$ współczynnik uczenia
$decay \leftarrow$ paramert zanikania
for each $x \in X$ **do**
  for each $user, item \in ratings$ **do**
    $predicion = global\_bias + IloczynSkalarny(user\_factors_{user}, item\_factors_{item})$;
    $error = ratings_{user,item} - prediction$
    //dopasowanie własności ukrytych:
    for each $f \in F$ **do**
      $delta_u = error * item\_factors_{item,f} - regularization * user\_factors_{user,f}$
      $delta_i = error * user\_factors_{user,f} - regularization * item\_factors_{item,f}$
      $user\_factors_{user,f}$ += $current\_learnrate * delta_u$
      $item\_factors_{item,f}$ += $current\_learnrate * delta_i$
    end for
  end for
  $current\_learnrate$ *= $decay$
end for
return $user\_factors, item\_factors$

5.1.2. Biased Matrix Factorization

Algorytm *Biased Matrix Factorization* jest modyfikacją wyżej opisanego algorytmu *Matrix Factorization*. Podobnie jak jego pierwowzór składa się z dwóch faz. W pierwszej fazie dodatkowo inicjowane są dwa dodatkowe wektory: skośność użytkowników (ang. *user bias*) i skośność elementów (ang. *item bias*).

Inaczej jest też obliczana skośność globalna:

$$global\_bias = \frac{\frac{a - r_{min}}{r_{max} - r_{min}}}{1 - \frac{a - r_{min}}{r_{max} - r_{min}}}, \tag{5.4}$$

gdzie

a to średnia wszystkich ocen
r_{min} to minimalna ocena w systemie
r_{max} to maksymalna ocena w systemie

Faza druga wygląda podobnie jak w przypadku algorytmu *Matrix Factorization*, jednak są uwzględniane dodatkowe parametry i wykonywane dodatkowe kroki.

Algorytm 4 Biased Matrix Factorization – Faza uczenia

$global\_bias \leftarrow$ średnia wszystkich ocen
$X \leftarrow$ liczba iteracji
$regU, regI, BiasReg \leftarrow$ parametry regulujące dla użytkownika, elementu i ogólny
$current\_learnrate \leftarrow$ współczynnik uczenia
$BiasLearnRate \leftarrow$ współczynnik uczenia skośności
$decay \leftarrow$ paramert zanikania
for each $x \in X$ **do**
  for each $user, item \in ratings$ **do**
    $score = global\_bias + user\_bias_{user} + item\_bias_{item} + IloczynSkalarny(user\_factors_{user}, item\_factors_{item})$
    $sig\_score = \frac{1}{1+\exp(-score)}$
    $predicion = rating_{min} + sig\_score + (rating_{max} - rating_{min})$
    $error = ratings_{user,item} - prediction$
    $gradient\_common = err * sig\_score * (1 - sig\_score) * (rating_{max} - rating_{min})$
    //dopasowanie skośności:
    $user\_bias_{user}$ += $BiasLearnRate * current\_learnrate * (gradient\_common - BiasReg * RegU * user\_bias_{user})$
    $item\_bias_{item}$ += $BiasLearnRate * current\_learnrate * (gradient\_common - BiasReg * RegI * item\_bias_{item})$
    //dopasowanie własności ukrytych:
    for each $f \in F$ **do**
      $delta_u = gradient\_common * item\_factors_{item,f} - RegU * user\_factors_{user,f}$
      $delta_i = gradient\_common * user\_factors_{user,f} - RegI * item\_factors_{item,f}$
      $user\_factors_{user,f}$ += $current\_learnrate * delta_u$
      $item\_factors_{item,f}$ += $current\_learnrate * delta_i$
    end for
  end for
  $current\_learnrate$ *= $decay$
end for
return $user\_factors, item\_factors$

5.1.3. SVD++

opisać SVD++

SVD++ jest rozszerzeniem metody SVD (dekompozycja głównych składowych, ang. *singular value decomposition*). Od poprzednich omawianych algorytmów różni go to, że korzysta nie tylko z podejścia aktywnego do tworzenia profilu użytkownika ale także z pasywnego (zob. 3.1.1).

Model SVD++ opisywany jest równaniem:

$$r_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + \frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} y_j\right), \tag{5.5}$$

Użytkownik powiązany jest z wektorem $p_u \in \mathbb{R}^f$ reprezentującym zainteresowanie konkretnymi cechami. Taki model uzupełniany jest sumą $\frac{1}{\sqrt{|N(u)|}} \sum_{j \in N(u)} y_j$, która re-

prezentuje informacje niejawne (*implicit feedback*). Zmienne b_u i b_i reprezentują obserwowane odchylenie od średniej dla użytkowników i elementów, natomiast μ to średnia wszystkich ocen[23].

5.2. Filtrowanie z analizą zawartości

Algorytmy content-based budują rekomendację na podstawie ocen, jakie zostały dotychczas wystawione przez użytkownika. Analizowane są cechy elementów i ich wartości oraz określana jest ich siła wpływu na finalną ocenę.

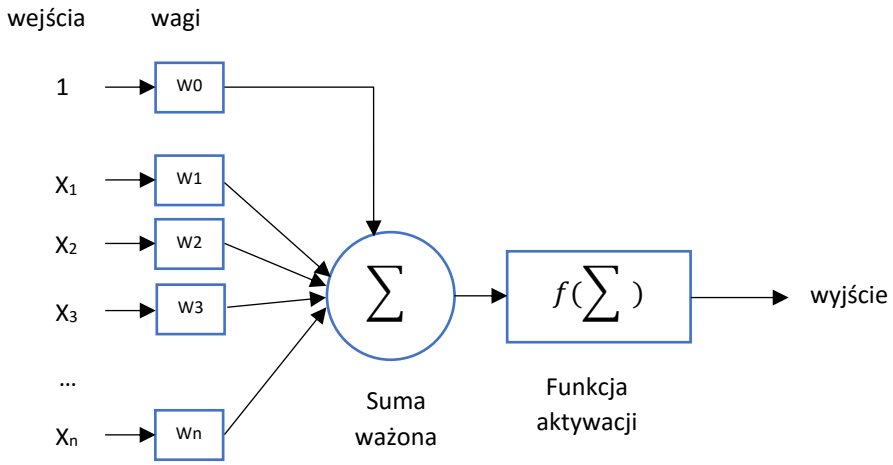
W tym celu dla każdego użytkownika tworzona jest sieć neuronowa, która uczy się jego preferencji.

W projekcie wykorzystana została implementacja sieci neuronowych z biblioteki AForge.NET Framework [21].

5.2.1. Konstrukcja sieci neuronowej

Struktura perceptronów

Sieć neuronowa składa się z trzech warstw neuronów (perceptronów). W każdej warstwie wszystkie neurony mają konstrukcję na jak rys. 5.4



Rys. 5.4: Schemat perceptronu

Do neuronu przekazywany jest zestaw wartości w postaci wektora x. Następnie obliczana jest suma ważona tych wartości w zależności od nadanych wag w. Proces dobierania odpowiednich wag jest nazywany uczeniem (zob. 5.2.2). W następnym kroku suma ważona przekazywana jest do funkcji aktywacji neuronu. Jeżeli funkcja przyj-

mie wartość wyższą lub równą niż określony próg aktywacji, to perceptron zostanie pobudzony (zwróci wartość 1). Proces ten obrazuje równanie 5.6.

$$N(x_1, x_2, x_3, \ldots, x_n) = \begin{cases} 1 & \text{jeśli } f(w_0 + \sum_{i=1}^{n} w_i x_i) \geqslant \eta \\ 0 & \text{jeśli } f(w_0 + \sum_{i=1}^{n} w_i x_i) < \eta \end{cases}, \tag{5.6}$$

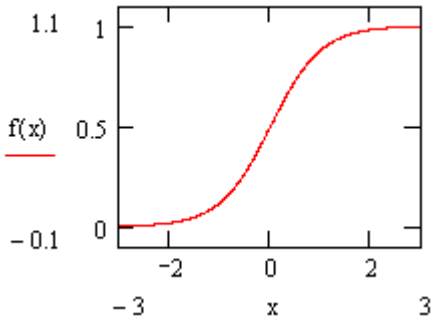
gdzie

w to wagi kolejnych wejść
x to wartości przekazywane do wejść
$f(u)$ to funkcja aktywacji neuronu
η to próg aktywacji neuronu

Na potrzeby algorytmu rekomendacji zdecydowano się przyjąć sigmoidalną unipolarną funkcję aktywacji neuronu (równanie 5.7). Funkcja przyjmuje wartości z zakresu $[0, 1]$.

$$f(x) = \frac{1}{1 + \exp(-\alpha x)}. \tag{5.7}$$

Wykres funkcji wygląda jak na rys. 5.5.



Rys. 5.5: Wykres sigmoidalnej funkcji aktywacji perceptronu [21]

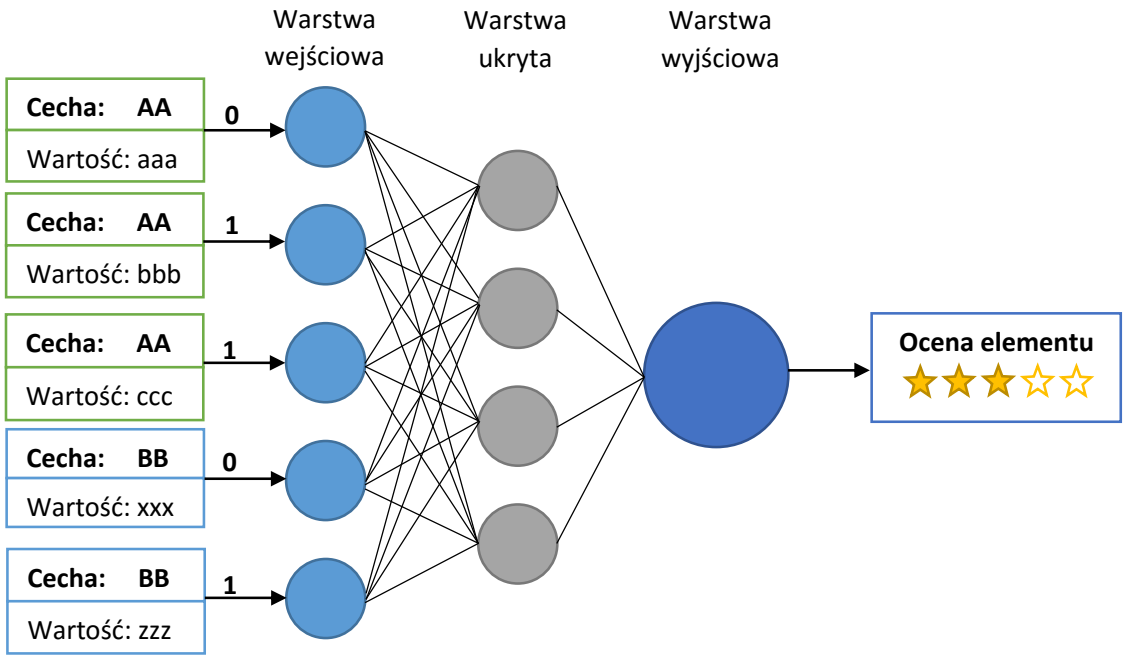
Struktura sieci i przebieg algorytmu

Pierwszym etapem algorytmu jest analiza cech elementów ocenionych przez użytkownika. Tworzona jest lista wszystkich występujących cech które powtarzają się minimum tyle razy, ile wynosi wartość parametru *minimumRepeatingFeatures*.

Następnie inicjowana jest sieć neuronowa. Ilość neuronów warstwy wejściowej jest równa ilości wyodrębnionych cech. Warstwa ukryta zawiera tyle neuronów ile jest to określone parametrem *hiddenLayerNeurons*. Warstwa wyjściowa składa się z tylko jednego neuronu (rys. 5.6).

W kolejnym etapie dla każdego elementu tworzona jest mapa cech. Jeżeli element zawiera daną cechę o danej wartości przypisywana jest wartość 1. W przeciwnym razie wstawiane jest 0. Rys. 5.7 przedstawia przykładową mapę cech. Tak przygotowana lista przekazywana jest do sieci neuronowej.

Na wyjściu sieć zwraca przewidywaną ocenę elementu.



Rys. 5.6: Schemat sieci neuronowej

Cecha	Wartość	Czy zawiera?
Aktor	Julia Roberts	1
Aktor	Al Pacino	1
Aktor	Brad Pitt	0
...		
Reżyser	Francis Ford Coppola	1
Reżyser	Darren Aronofsky	0
...		

Rys. 5.7: Mapa cech elementu. Wiadomo, że element X zawiera cechę „Aktor" o wartościach „Julia Roberts, Al Pacino" oraz cechę „Reżyser" o wartości „Francis Ford Coppola". Element nie zawiera cechy „Aktor" o wartości „Brad Pitt" ani cechy „Reżyser" o wartości „Darren Aronofsky" więc w te miejsca wstawiane jest 0.

5.2.2. Uczenie sieci neuronowej

Aby sieć zwracała jak najlepsze wyniki musi wcześniej zostać nauczona preferencji użytkownika. Uczenie sieci polega na odnalezieniu odpowiednich wag dla każdego wejścia każdego perceptronu sieci (budowa perceptronu zob. 5.2.1). Proces ten można zapisać w postaci

$$W_{ij}(n+1) = W_{ij}(n) + \Delta W_{ij}(n)\,, \tag{5.8}$$

gdzie

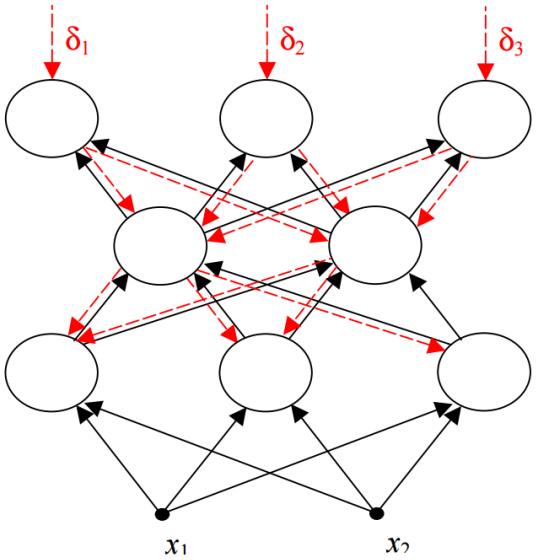
$W_{ij}(n)$	to poprzednie wagi wejść
$W_{ij}(n+1)$	to nowe wagi wejść
n	numer cyklu uczącego

Dostrajanie wartości wag można wykonać na wiele sposobów. Można wyróżnić uczenie nadzorowane (z nauczycielem), uczenie z krytykiem (ang. *reinforcement learning*) i uczenie samoorganizujące się (bez nadzoru) [37].

Na potrzeby systemu rekomendacji content-based opisywanego w tej pracy wykorzystane zostały trzy metody uczenia: algorytm propagacji wstecznej, algorytm RPROP i algorytm genetyczny. Wymienione metody należą do kategorii uczenia nadzorowanego.

5.2.3. Propagacja wsteczna

Algorytm propagacji wstecznej jest jedną z popularniejszych metod uczenia nadzorowanego jednokierunkowych sieci neuronowych. Zbiór uczący składa się z danych wejściowych (wektory cech elementu) i z informacji o oczekiwanym wyniku (ocena elementu). Różnica między wynikiem zwróconym przez sieć a wartością oczekiwaną stanowi miarę błędu sieci neuronowej.



Rys. 5.8: Algorytm propagacji wstecznej w sieci trójwarstwowej - idea działania [26]

Wiadomo, że funkcja celu jest funkcją ciągłą. Bazując na gradientowych metodach optymalizacji, wagi w sieci neuronowej aktualizowane są w następujący sposób:

$$W_{ij}(n+1) = W_{ij}(n) + \Delta W_{ij}(n)\,, \tag{5.9}$$

$$\Delta W_{ij}(n) = \eta p(W)\,, \tag{5.10}$$

gdzie

η to współczynnik uczenia (ang. *learning rate*)
$p(W)$ to kierunek w przestrzeni wielowymiarowej W

Aby wyznaczyć kierunek $p(W)$) dla wszystkich warstw sieci należy przejść przez kolejne etapy uczenia [14][37][26][15][48].

1. W kroku pierwszym sieć neuronowa poddawana jest analizie o zwykłym kierunku przepływu sygnałów. Wynikiem są wartości sygnałów wychodzących z neuronów warstw wyjściowej i ukrytych oraz pochodne funkcji aktywacji w kolejnych warstwach.
2. W kroku drugim kierunek przepływu sygnałów zostaje odwrócony (stąd nazwa propagacja wsteczna). Funkcje aktywacji zostają zastąpione przez swoje pochodne. Na oryginalne wyjście sieci (aktualnie wejście) podana zostaje wartość równa różnicy pomiędzy wynikiem oczekiwanym a wynikiem zwróconym przez sieć.
3. Obliczana zostaje wartość różnic wstecznych.
4. W kolejnym kroku następuje wreszcie proces adaptacji wag. Odbywa się on zarówno dla sieci zwykłej jak i dla sieci o propagacji wstecznej. Reguła modyfikacji ma postać:

$$\Delta W_{ij} = \eta \sum_{wzroce} \delta_{wyjcie} \cdot x_{wejcie}\,, \tag{5.11}$$

5. Powyższe kroki potarzane są dla każdej pary dane wejściowe - oczekiwany wynik tak długo, aż poziom błędu spadnie poniżej akceptowalnej wartości bądź osiągnięta zostanie maksymalna liczba iteracji.

Algorytm propagacji wstecznej obrazuje schemat blokowy 5.9.

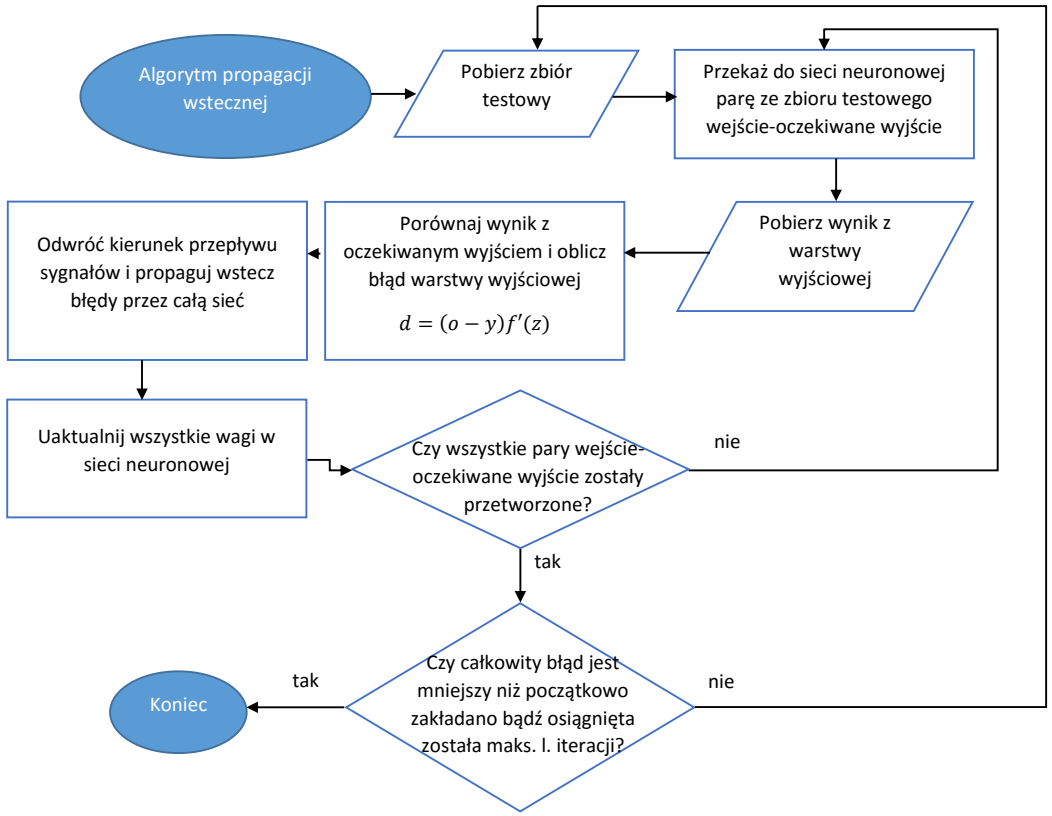
Metoda momentum

W celu zwiększenia efektywności działania algorytmu została zastosowana metoda momentum. I tak też standardowy sposób aktualizacji wag sieci (równanie 5.9) został zmodyfikowany w następujący sposób:

$$W_{ij}(n+1) = W_{ij}(n) + \Delta W_{ij}(n)\,, \tag{5.12}$$

$$\Delta W_{ij}(n) = \eta(n) \cdot p(n) + \alpha(W_{ij}(n) - W_{ij}(n-1))\,, \tag{5.13}$$

gdzie α jest współczynnikiem momentum z zakresu $[0, 1]$. Im wyższa wartość współczynnika tym większy jego wpływ na ostateczny kształt kolejnych wag. W dalszej części pracy przeprowadzono eksperymenty mające na celu dobranie optymalnej wartości współczynnika momentum.



Rys. 5.9: Algorytm propagacji wstecznej

5.2.4. Algorytm RPROP

Pomimo dużej popularności algorytmu propagacji wstecznej nie jest on pozbawiony wad. Sporym problemem jest jego relatywnie niska szybkość działania, szczególnie w przypadku bardzo dużych sieci neuronowych - a więc w przypadku odpowiadającemu potrzebom systemów rekomendacji. W odpowiedzi na te problemy Martin Riedmiller i Heinrich Braun zaproponowali w 1992 roku alternatywny algorytm - Resilient Backpropagation (RPROP).

Główną różnicą jest wykorzystywanie jedynie informacji o dodatniości lub ujemności każdej składowej gradientu zamiast o ich wartości (jak to się odbywa w oryginalnym algorytmie). Ponadto, współczynnik uczenia modyfikowany jest w każdym kolejnym kroku.

Modyfikacja współczynników odbywa się zgodnie ze wzorem:

$$\eta(t) = \begin{cases} min\{a\eta(t-1), \eta_{max}\}, & \text{gdy } \frac{\partial E^2(t)}{\partial v(t)} \frac{\partial E^2(t-1)}{\partial v(t-1)} > 0 \\ max\{b\eta(t-1), \eta_{min}\}, & \text{gdy } \frac{\partial E^2(t)}{\partial v(t)} \frac{\partial E^2(t-1)}{\partial v(t-1)} < 0 \\ \eta(n-1), & \text{w każdym innym przypadku} \end{cases}, \tag{5.14}$$

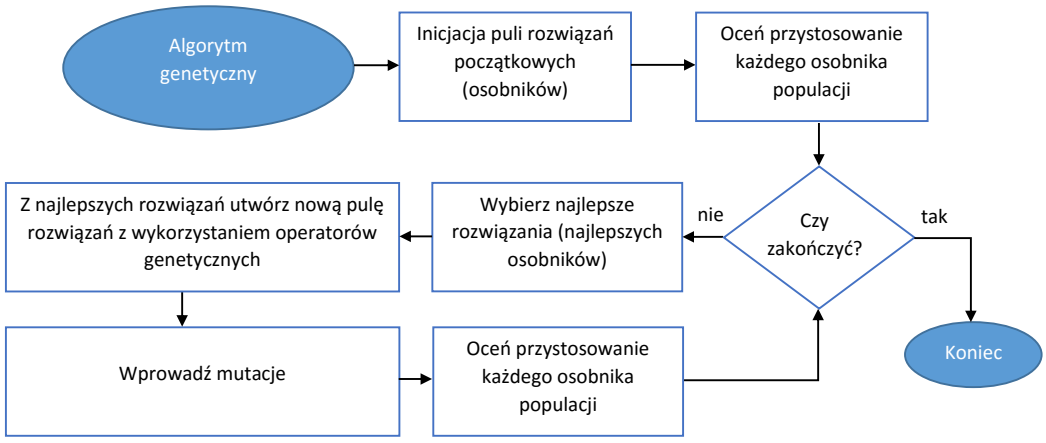
gdzie $\frac{\partial E^2(t)}{\partial v(t)}$ jest dokładną wartością składowej v gradientu. Ponadto stałe a, b, η_{min} i η_{max} są zdefiniowane następująco:

$a = 1.2$
$b = 0.5$
$\eta_{min} = 10^{-6}$
$\eta_{max} = 50$

Algorytm RPROP jest efektywniejszy pod kątem prędkości działania względem tradycyjnej propagacji wstecznej, więc dobrze sprawdza się tam, gdzie szybkość jest ważniejsza od wysokiej poprawności wyników [41][42].

5.2.5. Algorytm genetyczny

Koncepcja algorytmów genetycznych została zaproponowana już 1960 roku przez Johna Hollanda. Należą one do klasy algorytmów ewolucyjnych, inspirowanych zasadą doboru naturalnego Darwina. Analogią do środowiska naturalnego jest pewien problem, dla którego poszukiwane jest optymalne rozwiązanie. Populacja składa się z potencjalnych rozwiązań, które są oceniane funkcją przystosowania. Zgodnie z zasadą Darwina zwycięża najsilniejszy, czyli najlepsze (chociaż nie zawsze optymalne) rozwiązanie [38]. Schemat działania algorytmów genetycznych przedstawia rys. 5.10.



Rys. 5.10: Ogólny schemat algorytmu genetycznego

Jako, że algorytmy genetyczne należą do grupy algorytmów optymalizacyjnych mogą zostać wykorzystane do uczenia sieci neuronowych. W przypadku tej implementacji populację stanowią propozycje wag dla każdego perceptronu. Preferowane są takie zestawy wag, dla których sieć zwraca rezultaty obarczone najmniejszym błędem. Najlepsze zestawy są ze sobą krzyżowane oraz podlegają mutacji. W wyniku takich mechanizmów ewolucyjnych powstaje rozwiązanie, dla których sieć neuronowa zwraca oczekiwane rezultaty [22][31].

5.3. Algorytmy hybrydowe

5.4. Analiza złożoności i poprawności

Analiza złożoności i poprawności

Rozdział 6

Ocena eksperymentalna

6.1. Opis metody badawczej

6.1.1. Miara oceny

W celu zbadania jakości algorytmów zostały zastosowane miary oceny: średnia kwadratowa błędów (RMSE) i średni błąd bezwzględny (MAE).

Średnia kwadratowa błędów

Średnia kwadratowa błędów (ang. RMSE – *root mean square error*) jest często wykorzystywaną miarą służącą zmierzeniu różnicy pomiędzy wartościami przewidywanymi a rzeczywistymi (obserwowanymi).

RMSE jest stosunkowo dobrą miarą dokładności ale tylko w celu porównania różnych modeli dla tego samego zestawu danych. RMSE jest zależne od skali, zatem nie sprawdza się najlepiej w przypadku porównywania ze sobą różnych zmiennych [18].

Średnią kwadratową błędów wylicza się ze wzoru:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}\,, \tag{6.1}$$

gdzie

$\hat{y}_i$ to wartość przewidywana
y_i to wartość rzeczywista

Im niższa wartość RMSE tym bardziej zbliżone są wartości przewidywane do rzeczywistych, zatem tym lepszy jakościowo jest model.

Średni błąd bezwzględny

Inną miarą mierzenia jakości modeli predykcyjnych jest MAE (ang. *mean absolute error*). Podobnie jak RMSE miara ta jest zależna od skali, zatem najlepiej sprawdza się w działaniu na tym samym zestawie danych [18].

Średni błąd bezwzględny wylicza się ze wzoru:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|\hat{y}_t - y_t|\,, \tag{6.2}$$

gdzie

$\hat{y}_i$ to wartość przewidywana
y_i to wartość rzeczywista

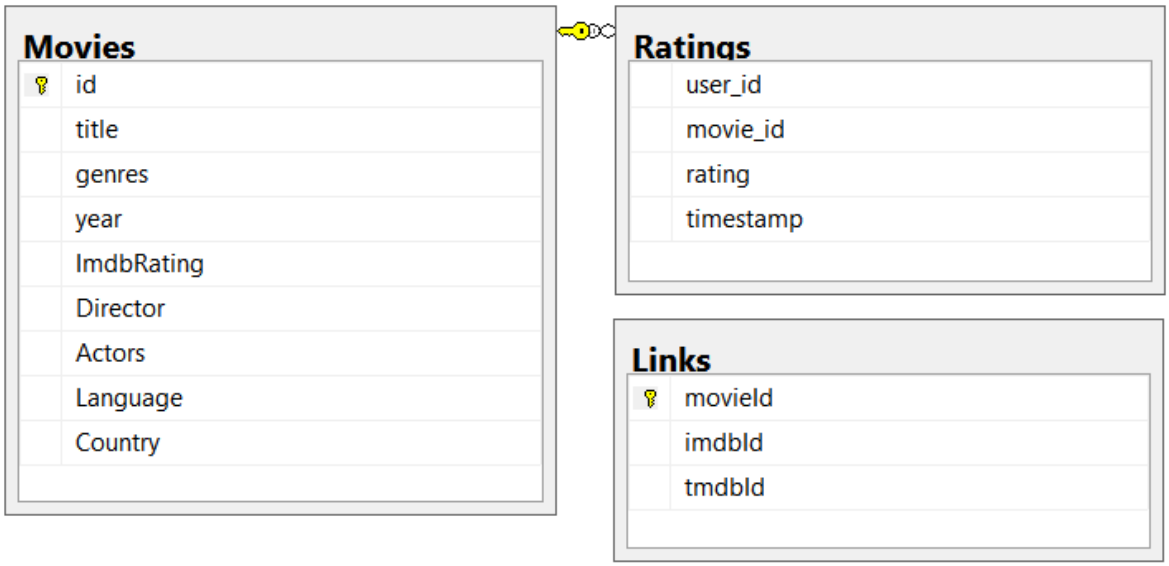
6.1.2. Zbiory danych

By uzyskać jak najbardziej miarodajne wyniki, badania zostały przeprowadzone na trzech różnych bazach danych z trzech różnych domen.

MovieLens

MovieLens [13] to baza zawierająca oceny filmów przez użytkowników portalu movielens.org. Baza zawiera 3706 filmów i 1000209 ocen wystawionych przez 6040 unikalnych użytkowników pomiędzy 25 kwietnia 2000 a 28 lutym 2003. Filmy oceniane są w skali od 1 do 5, gdzie 1 jest oceną najgorszą a 5 najlepszą.

Baza zawiera tabelę łączącą numery identyfikacyjne filmów z bazą IMDB.com. Korzystając z tego autorka pracy rozszerzyła oryginalną bazę filmów o informacje pobrane z IMDB.com. Ostateczny kształt bazy widoczny jest na rys. 6.1.



Rys. 6.1: Schemat bazy MovieLens

Yahoo Music

Opisać Yahoo Music

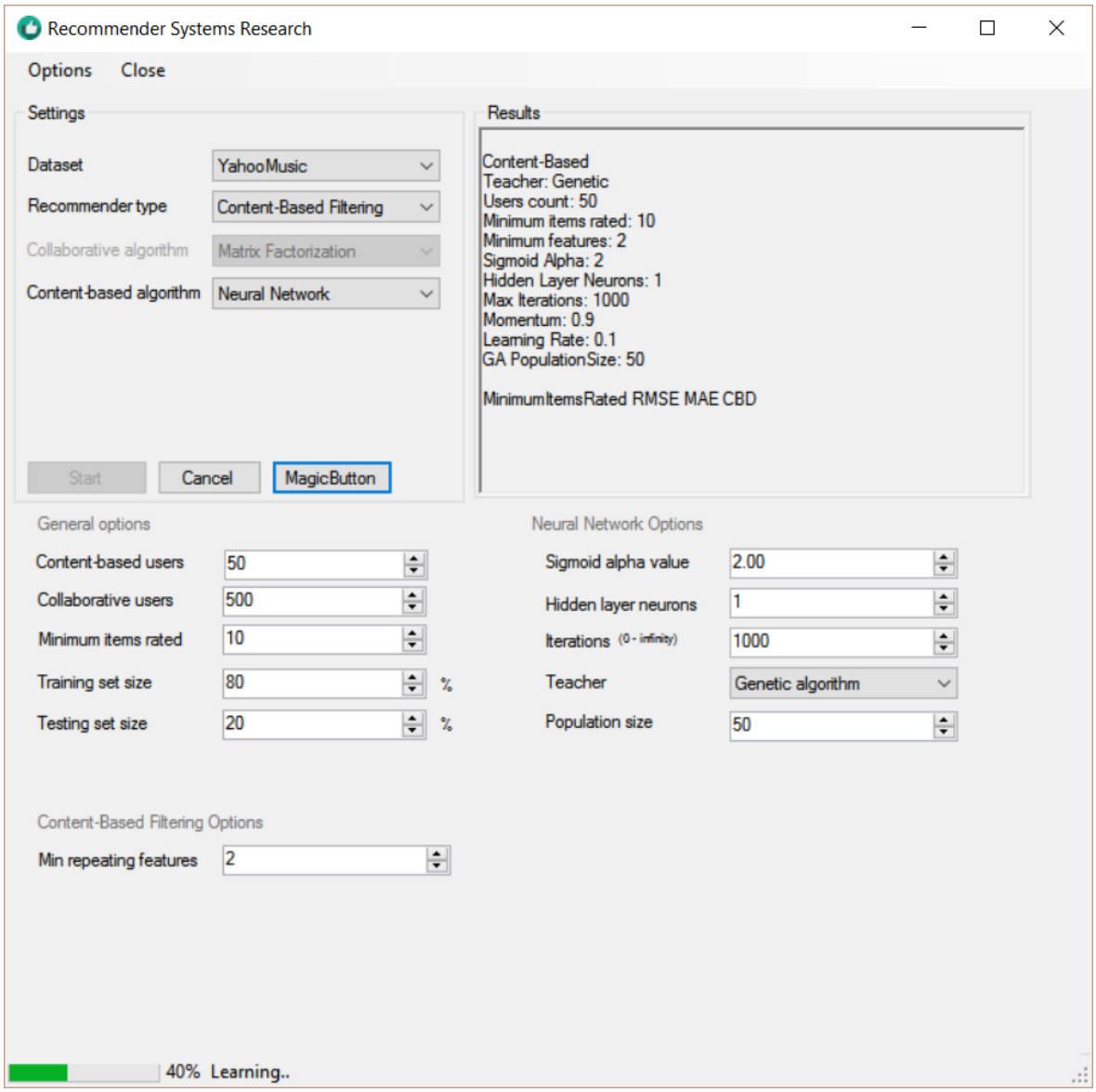
Amazon Meta

Opisać Amazon Meta

[27]

6.2. Środowisko symulacyjne

Rys. 6.2 przedstawia zrzut ekranu środowiska symulacyjnego stworzonego przez autorkę pracy na potrzeby przeprowadzenia badań algorytmów. Główny interfejs programu składa się z trzech części: ustawienia podstawowe, okno z wynikiem oraz panel sterujący ustawieniami zaawansowanymi.



Rys. 6.2: Zrzut ekranu środowiska symulacyjnego

W sekcji z ustawieniami podstawowymi możliwy jest wybór:

- bazy danych, która zostanie wykorzystana do pomiarów;
- typu algorytmu rekomendującego (content-based, collaborative bądź hybrydowy)
- w przypadku wyboru kolaboratywnego filtrowania lub filtrowania hybrydowego możliwy jest wybór typu algorytmu: Matrix Factorization, Biased Matrix Factorization lub SVD++.

- w przypadku wyboru filtrowania z analizą zawartości lub filtrowania hybrydowego automatycznie wybierany jest algorytm oparty na sieci neuronowej.

Widok ustawień zaawansowanych zmienia się w zależności od wybranego filtrowania. W przypadku wyboru content-based istnieje możliwość regulowania parametrów sieci neuronowej. Zawsze istnieje możliwość regulowania kryteriów doboru zestawu danych.

Kryteria doboru zestawu danych są następujące:

- liczba użytkowników do pobrania do algorytmu content-based;
- liczba użytkowników do pobrania do algorytmu collaborative;
- minimum elementów, jakie zostały ocenione przez każdego pobranego użytkownika;
- stosunek wielkości zbioru treningowego do zbioru testowego (domyślnie 80%-20%).

Parametry sieci neuronowej podlegające regulacji to:

- Sigmoidalna wartość alfa;
- liczba neuronów w warstwie ukrytej;
- maksymalna liczba iteracji uczenia sieci neuronowej;
- nauczyciel sieci neuronowej: propagacja wsteczna, rprop (resilient backpropagation) lub algorytm genetyczny;
- w przypadku wyboru algorytmu genetycznego – rozmiar każdej kolejnej populacji;
- minimalna ilość powtórzeń danej cechy aby była brana pod uwagę w trakcie budowania rekomendacji.

6.3. Metodologia

Metodologia

6.4. Przeprowadzone eksperymenty

Przeprowadzone eksperymenty

Rozdział 7

Wnioski

Rozdział 8

CHAPTER 1

8.1. SECTION

Algorytm 5 Alghoritm 4

$T \leftarrow$ text under analysis
for each word $w \in T$ **do**
  $S_w \leftarrow FIND\_SENTIMENT(w)$
  if $S_w = POSITIVE$ **then**
    $Sentiment[POSITIVE] + +$
  else if $S_w = NEGATIVE$ **then**
    $Sentiment[NEGATIVE] + +$
  else
    $Sentiment[NEUTRAL] + +$
  end if
end for
return $\arg\max_x Sentiment[x]$

Rys. 8.1: Schema 1

¡GRAPHIC¿

8.2. Section 2

8.2.1. Subsection 1

Subsubsection 1
Definicja 1
Definicja - *pierwsza*

Dodatek A

Appendix 1

Spis rysunków

Spis wzorów

Spis algorytmów

Bibliografia

[1] About the Music Genome Project. http://www.pandora.com/about/mgp. Data dostępu: 2016-07-19.

[2] Allegro – korzystanie z systemu rekomendacji. http://faq.allegro.pl/artykul/27613/korzystanie-z-systemu-rekomendacji. Data dostępu: 2016-07-19.

[3] Avron H., Kale S., Kasiviswanathan S., Sindhwani V. Efficient and practical stochastic subgradient descent for nuclear norm regularization. https://www.cs.cmu.edu/~yuxiangw/docs/SSGD.pdf. Data dostępu: 2016-09-07.

[4] Bottou L. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer, 2012.

[5] Celma O. *The Long Tail in Recommender Systems*, pages 87–107. Springer-Verlag Berlin Heidelberg, 2010.

[6] Cheng J., Liu Y., Zhang H., Wu X., Chen F. A new recommendation algorithm based on user's dynamic information in complex social network. *Mathematical Problems in Engineering*, 2015, 2015.

[7] Davidson J., Liebald B., Liu J., Nandy P., Van Vleet T., Gargi U., Gupta S., He Y., Lambert M., Livingston B. et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296. ACM, 2010.

[8] Desrosiers C., Karypis G. *Recommender Systems Handbook*, chapter A Comprehensive Survey of Neighborhood-based Recommendation Methods, pages 107–144. Springer, New York Dordrecht Heidelberg London, 2010.

[9] Filmweb – najczęściej zadawane pytania. http://www.filmweb.pl/help. Data dostępu: 2016-07-19.

[10] Gantner Z., Rendle S., Drumond L., Freudenthaler C. Mymedialite recommender system library. http://www.mymedialite.net/. Data dostępu: 2016-09-05.

[11] Gantner Z., Rendle S., Freudenthaler C., Schmidt-Thieme L. Mymedialite: a free recommender system library. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 305–308. ACM, 2011.

[12] Gupta P., Goel A., Lin J., Sharma A., Wang D., Zadeh R. Wtf: The who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 505–514. ACM, 2013.

[13] Harper F. M., Konstan J. A. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19, 2016.

[14] Haykin S. Neural networks: A comprehensive foundation: Macmillan college publishing company. *New York*, 1994.

[15] Hertz J. A., Krogh A. S., Palmer R. G., Jankowski S. *Wstęp do teorii obliczeń neuronowych*. Wydawnictwa Naukowo-Techniczne, 1993.

[16] Huttner J. From Tapestry to SVD: A survey of the algorithms that power recommender system. Master's thesis, Haverford College Department of Computer Science, 05 2009.

[17] Huynh T., Hoang K. Modeling collaborative knowledge of publishing activities for research recommendation. In *International Conference on Computational Collective Intelligence*, pages 41–50. Springer, 2012.

[18] Hyndman R. J., Koehler A. B. Another look at measures of forecast accuracy. *International journal of forecasting*, 22(4):679–688, 2006.

[19] IMDb database statistics. `http://www.imdb.com/stats`. Data dostępu: 2016-07-19.

[20] Ji K., Sun R., Shu W., Li X. Next-song recommendation with temporal dynamics. *Knowledge-Based Systems*, 88:134–143, 2015.

[21] Kirillov A. AForge.NET framework. `http://www.aforgenet.com/framework/`. Data dostępu: 2016-09-05.

[22] Kirillov A. AForge.NET framework – evolutionarylearning class. `http://www.aforgenet.com/framework/docs/html/cc8bebc5-da54-5c56-6ddf-6a93aec7b9cd.htm`. Data dostępu: 2016-09-06.

[23] Koren Y. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[24] Koren Y., Bell R. *Recommender Systems Handbook*, chapter Advances in Collaborative Filtering, pages 145–186. Springer, New York Dordrecht Heidelberg London, 2010.

[25] Koren Y., Bell R., Volinsky C. et al. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[26] Kwater T. Algorytmy uczenia sieci neuronowych. `http://www.neurosoft.edu.pl/media/pdf/tkwater/sztuczna_inteligencja/2_alg_ucz_ssn.pdf`. Data dostępu: 2016-09-06.

[27] Leskovec J., Adamic L. A., Huberman B. A. The dynamics of viral marketing. *ACM Transactions on the Web (TWEB)*, 1(1):5, 2007.

[28] Linden G., Smith B., York J. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[29] Lops P., de Gemmis M., Semeraro G. *Recommender Systems Handbook*, chapter Content-based Recommender Systems: State of the Art and Trends, pages 73–100. Springer, New York Dordrecht Heidelberg London, 2010.

[30] Maleszka M., Mianowska B., Nguyen N. T. A method for collaborative recommendation using knowledge integration tools and hierarchical structure of user profiles. *Knowledge-Based Systems*, 47:1–13, 2013.

[31] Montana D. J., Davis L. Training feedforward neural networks using genetic algorithms. In *IJCAI*, volume 89, pages 762–767, 1989.

[32] Mymedialite: Example experiments. `http://www.mymedialite.net/examples/datasets.html`. Data dostępu: 2016-07-24.

[33] Netflix Prize (I tried to resist, but...). `https://www.snellman.net/blog/archive/2006-10-15-netflix-prize.html`. Data dostępu: 2016-07-08.

[34] Netflix Prize: forum. `http://www.netflixprize.com/community/viewtopic.php?id=1537`. Data dostępu: 2016-07-08.

[35] Netflix Prize Rankings. `http://www.hackingnetflix.com/2006/10/netflix_prize_r.html`. Data dostępu: 2016-07-08.

[36] Netflix Prize Rules. `http://www.netflixprize.com//rules`. Data dostępu: 2016-07-08.

[37] Osowski S. *Sieci neuronowe w ujęciu algorytmicznym.* Wydawnictwa Naukowo-Techniczne, 1996.

[38] Pena-Reyes C. A., Sipper M. Evolutionary computation in medicine: an overview. *Artificial Intelligence in Medicine*, 19(1):1–23, 2000.

[39] Pogue D. A Stream of Movies, Sort of Free. *The New York Times*, 2007.

[40] Ricci F., Rokach L., Shapira B. *Recommender Systems Handbook*, chapter Introduction to Recommender Systems Handbook, pages 1–35. Springer, New York Dordrecht Heidelberg London, 2010.

[41] Riedmiller M., Braun H. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *Neural Networks, 1993., IEEE International Conference On*, pages 586–591. IEEE, 1993.

[42] Riedmiller M., Rprop I. Rprop-description and implementation details. 1994.

[43] Rohrmann T. Computing recommendations at extreme scale with apache flink. *http://data-artisans.com/computing-recommendations-at-extreme-scale-with-apache-flink*, 2015.

[44] Rubens N., Kaplan D., Sugiyama M. Active learning in recommender systems. In Kantor P., Ricci F., Rokach L., Shapira B., editors, *Recommender Systems Handbook*, pages 735–767. Springer, 2011.

[45] Sarwar B., Karypis G., Konstan J., Riedl J. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[46] Schafer J., Frankowski D., Herlocker J., Sen S. *The Adaptive Web*, chapter Collaborative filtering recommender systems, page 291–324. Springer Berlin / Heidelberg, 2007.

[47] Sharma R., Singh R. Evolution of Recommender Systems from Ancient Times to Modern Era: A Survey. *Indian Journal of Science and Technology*, 9(20), 2016.

[48] Timothy M. Sieci neuronowe w praktyce. *WNT, Warszawa*, 1996.

[49] Willmott C. J., Matsuura K. Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance. *Climate research*, 30(1):79–82, 2005.

[50] Zhang H.-R., Min F., He X., Xu Y.-Y. A hybrid recommender system based on user-recommender interaction. *Mathematical Problems in Engineering*, 2015, 2015.