Application of Social Network Information to Collaborative Filtering Recommender Systems using Matrix Factorization

José Amorim e Miguel Sandim

Faculdade de Engenharia da Universidade do Porto,
{ei12190, miguel.sandim}@fe.up.pt

Resumo Os sistemas de recomendação têm-se tornado nos últimos anos numa área vasta e importante na literatura, especialmente os sistemas 'colaborativos' (baseados no *collaborative filtering*). Mais recentemente, começa também a surgir um novo tipo de sistemas de recomendação: os sistemas de recomendação 'sociais', que incorporam nas suas previsões dados sociais entre os utilizadores dos sistema.
Este artigo apresenta uma visão global da área dos sistemas de recomendação (com especial ênfase nos sistemas de recomendação sociais colaborativos) e das abordagens utilizadas (especializando-se na mais frequente: as baseadas em fatorização matricial). Estas abordagens são postas à prova face a algumas abordagens 'não-sociais', o que revela uma melhoria da precisão das recomendações utilizando este tipo de técnicas.

Keywords: recommender systems, collaborative filtering, matrix factorization, social trust

1 Introdução

Na atual 'era digital' um dos maiores problemas a solucionar é o excesso de informação, pelo que é necessário tornar o acesso à informação considerada 'relevante' mais rápido e eficaz. Por exemplo, se um utilizador desejasse comprar um computador numa loja *online* e tivesse de comparar manualmente as especificações e *reviews* de todos os computadores disponíveis nesta, o processo de seleção seria bastante demoroso e cansativo. Os sistemas de recomendação (em inglês designados de 'recommender systems') apareceram como solução a este problema, tornando-se rapidamente numa ferramenta importante no dia-a-dia.

Nos últimos vinte anos, a área de investigação dos sistemas de recomendação tornou-se numa área forte e independente [1], grande parte desta dedicada ao desenvolvimento de novas abordagens e algoritmos que permitam melhor precisão nas recomendações. Algumas destas abordagens incluem [2, 1]: classificadores *bayesianos* e modelos probabilísticos, *clustering*, árvores de decisão, redes neuronais artificiais, regressão linear e fatorização matricial. Grande parte da literatura neste tópico foca a recomendação de filmes, contudo também existe

literatura centrada em diferentes tópicos [3], tais como: música, televisão, livros, documentos, *e-learning*, *e-commerce*, aplicação em mercados e sistemas de pesquisa *web*.

Por outro lado, o crescente aumento da popularidade de redes sociais *online* e aparecimento de serviços comerciais com componente social (nomeadamente serviços de televisão digital – como a NOS, serviços de *streaming* de música – como o Spotify, etc...) apresentam novas oportunidades para melhorar a precisão das recomendações face aos sistemas de recomendação tradicionais [1]. Na área dos sistemas de recomendação ditos 'sociais', a fatorização matricial é uma das técnicas que garante mais precisão, tendo também já sido apresentada como útil em áreas como visão por computador e análise de texto [1].

Este artigo encontra-se dividido da seguinte forma: a Secção 2 apresenta um visão geral dos diferentes tipos de sistemas de recomendação e várias abordagens usadas (com destaque para os sistemas de recomendação sociais e os métodos baseados em fatorização matricial), na Secção 3 é apresentada uma aplicação multi-agente desenvolvida capaz de utilizar diferentes algoritmos para recomendar filmes a utilizadores, na Secção 4 são apresentadas várias experiências relativas à eficácia deste tipo de algoritmos e são analisados os resultados e na Secção 5 é feita uma conclusão relativa ao trabalho desenvolvido.

2 Visão Geral dos Sistemas de Recomendação

O problema da recomendação de items a utilizadores pode ser formulado do seguinte modo [2]:

Dado um conjunto de utilizadores U num sistema, um conjunto de produtos I possíveis de serem recomendados e uma função de utilidade f que mede o quão apropriado é um item i para um utilizador u, o objetivo de um sistema de recomendação é o de, para cada utilizador $u \in U$, recomendar um item $i'_u \in I$ que maximiza a função de utilidade para u:

$$\forall u \in U,\ i'_u = \arg\max_{i \in I} f(u, i) \tag{1}$$

onde:

i'_u : Item recomendado ao utilizador u

Nos casos em que o objetivo é o de recomendar os N melhores itens para um utilizador, obtêm-se os N itens cuja utilidade f é superior à dos restantes. Na maioria dos casos onde um sistema de recomendação é aplicado, a utilidade $f(u, i)$ corresponde um *rating* numérico que um utilizador u atribui a um item i [2]. De notar também que a função de utilidade f é geralmente representada através de uma matriz utilizador-item $R \in \mathbb{R}^{|U| \times |I|}$.

O principal desafio num sistema de recomendações é que a função f não está definida para todo o espaço $U \times I$, mas apenas para um sub espaço muitas vezes restrito deste [2] (já que é difícil um utilizador manifestar direta ou indiretamente

o seu interesse para todos os itens disponíveis, assim como é raro existir um *feedback* relativamente a um item por parte de todos os utilizadores no sistema). Assim, é necessário utilizar mecanismos que permitam obter uma função de utilidade aproximada $\hat{f}(u, i)$ que esteja também definida nos casos em que f não o está. Esta função é geralmente representada através de uma matriz $\hat{R} \in \mathbb{R}^{|U| \times |I|}$.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7
u_1	5		1	5			2
u_2	4	1		5		4	1
u_3	5		1		5	5	1
u_4			5			3	
u_5	2						5
u_6		2				5	

Figura 1. Exemplo de uma matriz de utilidades R para seis utilizadores e sete itens.

Os sistemas de recomendação podem ser classificados em três categorias diferentes, dependendo do modo como as recomendações são realizadas [2]:

- Sistemas baseados em Conteúdos (*Content-base*): Um utilizador receberá recomendações relativamente a itens 'semelhantes' a itens que foram da sua preferência no passado.
- Sistemas Colaborativos (*Collaborative* – também designados de baseados no *Collaborative Filtering*): Um utilizador receberá recomendações relativamente a itens da preferência de utilizadores com gostos semelhantes.
- Sistemas Híbridos (*Hybrid*): sistemas que combinam as abordagens baseadas em Conteúdos e Colaborativas.

As abordagens colaborativas podem ainda ser distinguidas em abordagens baseadas em modelos (*model-based*) e abordagens baseadas em vizinhos (*neighborhood-based*) [1]. No caso das abordagens *model based*, as utilidades já definidas são usadas para construir um modelo preditivo para estimar as utilidades em falta. Por sua vez, no caso das abordagens *neightborhood-based* as utilidades já definidas no sistema são utilizadas diretamente para estimar as utilidades não conhecidas de um utilizador u para certos itens i' [4]. Neste caso, esta estimação pode ser feita de duas maneiras distintas [4]:

- Métodos *user-based*, em que as utilidades são estimadas usando utilidades fornecidas por outros utilizadores semelhantes a u. Dois utilizadores consideram-se semelhantes se as suas utilidades estão correlacionadas (i.e., se avaliam os produtos com o mesmo *rating*).
- Métodos *item-based*, em que as utilidades são baseadas nas utilidades de u para itens semelhantes a i'. Neste caso, dois itens consideram-se semelhantes se as suas utilidades estão correlacionadas (i.e., se outros utilizadores no sistema avaliaram de forma semelhante estes dois itens).

2.1 Sistemas de Recomendação Sociais

Um sistema de recomendação social procura melhorar a precisão de um sistema de recomendação tradicional, incorporando um novo *input*: a confiança computacional entre utilizadores numa rede social [1]. Esta confiança pode ser obtida explicitamente ou implicitamente (e.g., através da análise das interação entre utilizadores). Este *input* adicional é representado através de uma matriz de confiança S de dimensões $|U| \times |U|$ em que cada elemento $S_{u,v}$ toma um valor no intervalo $]0, 1]$. Cada valor $S_{u,v}$ pode ser interpretado como a confiança do utilizador u no utilizador v no contexto de uma rede social.

	u_1	u_2	u_3	u_4	u_5	u_6
u_1		0.9	0.7			
u_2	0.8					0.7
u_3		0.8				
u_4			0.2		0.9	
u_5				0.8		
u_6	0.6				0.4	

Figura 2. Exemplo de uma matriz de confiança S entre seis utilizadores.

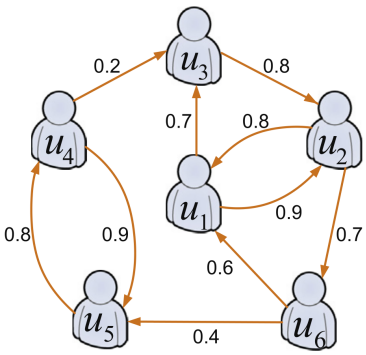

Figura 3. Grafo social correspondente à matriz S da figura 2.

Os sistemas de recomendação sociais baseiam-se geralmente em abordagens colaborativas, sendo que estas podem por sua vez ser classificadas em [1]:

– Abordagens sociais baseadas em Fatorização Matricial – em que a utilidade $f(u, s)$ é modelada segundo características latentes dos utilizadores e itens no sistema utilizando fatorização de matrizes.

Principais algoritmos [1]: 'Social recommendation model' (SoRec), 'Social trust ensemble model' (STE), 'Social Matrix Factorization model' (Social MF).

– Abordagens sociais baseadas em Vizinhos – em que a previsão da utilidade de certos itens i' para um utilizador u é obtida percorrendo o grafo de confiança (obtido a partir da matriz S) e obtendo as utilidades fornecidas por amigos diretos e indiretos de u.
Principais algoritmos [1]: ' Trust weighted', 'Bayesian inference based', 'Random walk based', 'Nearest neighbor' (NN).

2.2 Abordagens baseadas em Fatorização Matricial

O objetivo da fatorização matricial é o de, para uma matriz X, encontrar duas matrizes A e B tais que:

$$X = A \times B \tag{2}$$

A aplicação de métodos baseados na fatorização matricial parte do pressuposto de que existem *features* latentes (e portanto, não definidas *à priori*) que explicam como um utilizador u avalia um item i [5]. Com estas *features* descobertas, é possível prever qual o *rating* que o utilizador u dará ao item i. Estes métodos também assumem que $K \ll |U|, |I|$, onde K é o número de *features* latentes. Assim, aplicando este conceito aos conceitos apresentados anteriormente, é possível determinar uma matriz $P \in \mathbb{R}^{|U| \times K}$ e uma matriz $Q \in \mathbb{R}^{|I| \times K}$ tal que [1]:

$$R \approx \hat{R} = P \times Q^T + r_m \tag{3}$$

onde:

r_m : valor do *offset* escalar a adicionar

Estas matrizes podem também ser interpretadas do seguinte modo: cada elemento $P_{u,k}$ representa o quão associado um utilizador u se encontra com a *feature* k, enquanto que cada elemento $Q_{i,k}$ representa o quão associado um item i se encontra com a *feature* k.

O valores de P e Q podem ser determinados do seguinte modo [5]:

1. Inicializar P e Q com valores aleatórios.
2. Aplicar um algoritmo de otimização multidimensional (e.g., *gradient descent*), minimizando uma função de custo (que poderá ser, por exemplo a diferença entre os valores conhecidos na matriz R e os obtidos em $\hat{R}$).

Uma possível função de custo pode ser a seguinte [1]:

$$\sum_{(u,i) \text{ obs.}} \left(R_{u,i} - \hat{R}_{u,i}\right)^2 + \lambda\left(||P||_F^2 + ||Q||_F^2\right) \tag{4}$$

onde:

(u, i) obs. : Valores de u e i para os quais $R_{u,i}$ está definido
λ : Parâmetro de regularização (de modo a evitar *overfitting*) (≥ 0)
$||P||_F$: Norma de 'Frobenius' da matriz P

2.2.1 'Social recommendation model' (SoRec)

A abordagem SoRec [1] é uma das possíveis que faz uso da matriz S para melhorar precisão de $\hat{R}$. Neste caso, além da fatorização descrita na expressão 3, é realizada a seguinte fatorização adicional (de notar que $Z \in \mathbb{R}^{|U| \times K}$):

$$\hat{S^*} = Q \times Z^T + s_m \tag{5}$$

onde:

s_m : valor do *offset* escalar a adicionar

Neste caso a função de custo é dada por:

$$\sum_{(u,i) \text{ obs.}} \left(R_{u,i} - \hat{R}_{u,i}\right)^2 + \sum_{(u,v) \text{ obs.}} \left(S^*_{u,v} - \hat{S^*}_{u,v}\right)^2 + \lambda\left(||P||_F^2 + ||Q||_F^2 + ||Z||_F^2\right) \tag{6}$$

onde:

(u, i) obs. : Valores de u e i para os quais $R_{u,i}$ está definido
(u, v) obs. : Valores de u e v para os quais $S_{u,v}$ está definido
λ : Parâmetro de regularização (≥ 0)

A matriz S^* é calculada a partir de S:

$$S^*_{u,v} = S_{u,v}\sqrt{\frac{d_v^-}{d_u^+ + d_v^-}} \tag{7}$$

onde:

d_v^- : *out-degree* do utilizador v no grafo social obtido a partir de S (número de utilizadores que confiam em v)
d_u^+ : *in-degree* do utilizador u no grafo social obtido a partir de S (número de utilizadores em que u confia)

2.2.2 Modelo 'Social MF'

A abordagem Social MF [1] é outra das possíveis que faz uso da confiança entre utilizadores. Nesta abordagem, é realizada a fatorização descrita na expressão 3.

Neste caso a função de custo é dada por:

$$\sum_{(u,i)\ \text{obs.}} \left(R_{u,i} - \hat{R}_{u,i}\right)^2 + \beta \sum_{u \in U} ||Q_u - \sum_{v \in \gamma_u^+} S^*_{u,v} Q_v||^2 + \lambda\Big(||P||_F^2 + ||Q||_F^2\Big) \quad (8)$$

onde:

(u,i) obs. : Valores de u e i para os quais $R_{u,i}$ está definido
β : Parâmetro de peso da informação da matriz S (≥ 0) (de notar que caso tome o valor de 0 a informação da confiança entre utilizadores é ignorada)
γ_u^+ : Utilizadores em quem o utilizador u confia
λ : Parâmetro de regularização (≥ 0)

A matriz S^* é calculada a partir da normalização dos valores de S, de modo a que $S^* \propto S$ e que $\sum_{v \in U} S^*_{u,v} = 1,\ \forall u \in U$.

3 Desenvolvimento da Aplicação

De modo a colocar em prática os algoritmos em análise, foi desenvolvida uma aplicação multi-agente com interface gráfica de recomendação de filmes a utilizadores.

3.1 JADE

A aplicação possui dois tipos de agentes: o agente 'recomendador' e os agentes 'utilizadores'. A implementação dos agentes foi feita através da *framework JADE*. Esta *framework* permite o desenvolvimento de aplicações multi-agente em ambiente *Java* seguindo o padrão *FIPA*. O *FIPA* define um modelo de arquitetura que permite a criação, gestão e operação de agentes.

O *JADE* possui na sua estrutura:

1. Agent Management System (AMS): representa a autoridade da plataforma, permitindo a criação e a destruição de agentes. Contém a lista de identificadores de agentes (AID) certificando-se que cada agente tem um identificador único.
2. Directory Facilitator (DF): disponibiliza o serviço de Páginas Amarelas, onde um agente pode procurar outro agente que forneça um serviço que necessite.

Os agentes 'utilizadores' utilizam o serviço de Páginas Amarelas (DF) para procurarem um agente 'recomendador' que disponibiliza o serviço 'recomendação'. O agente 'recomendador' é criado no *main container* enquanto cada agente 'utilizador' é criado num *container* diferente.

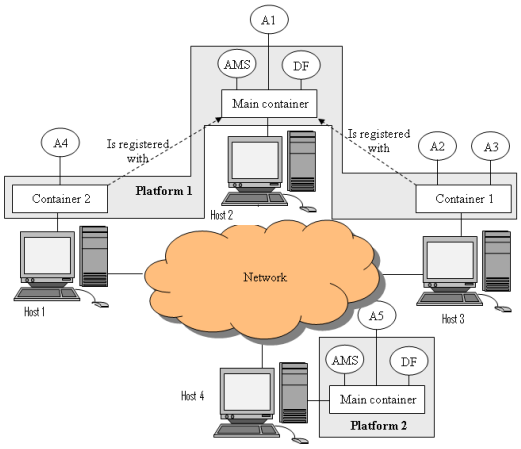

Figura 4. Modelo da arquitetura do *JADE*.

3.2 Protocolo de Interação

O protocolo de interação utilizado entre o agente de recomendação e os agentes utilizadores foi o protocolo *FIPA-Request*.

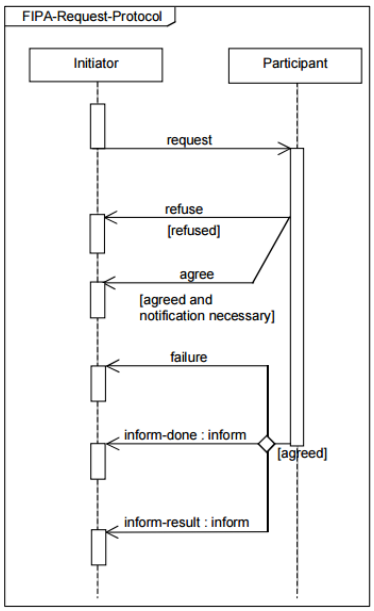

Figura 5. Protocolo de Interação FIPA-Request.

Neste caso o agente 'utilizador' é o *Initiator* e o agente 'recomendador' é o *Participant*.

O agente 'utilizador' inicia a interação fazendo um pedido ao agente 'recomendador', podendo este pedido ser:

1. Obter a listagem de itens não avaliados por este (aqueles para os quais $R_{u,i}$ não está definido);
2. Avaliar um item;
3. Obter a listagem de k itens recomendados para o agente;
4. Mudar o valor da confiança para com outro utilizador;
5. Atualizar modelo de recomendação (com base nos valores atualizados das matrizes R e S) – de notar que o modelo do sistema de recomendação é apenas atualizado quando este pedido é realizado para evitar *overhead* nos restantes pedidos).

Estes diferentes pedidos estão ilustrados na figura 6. O agente 'recomendador' tenta executar cada pedido recebido, sendo que caso algum erro ocorra ou o agente não possa completar a ação, o 'recomendador' responde ao agente 'utilizador' com uma mensagem de erro (*Failure*). Caso contrário, o agente 'recomendador' reponde com o resultado do pedido.

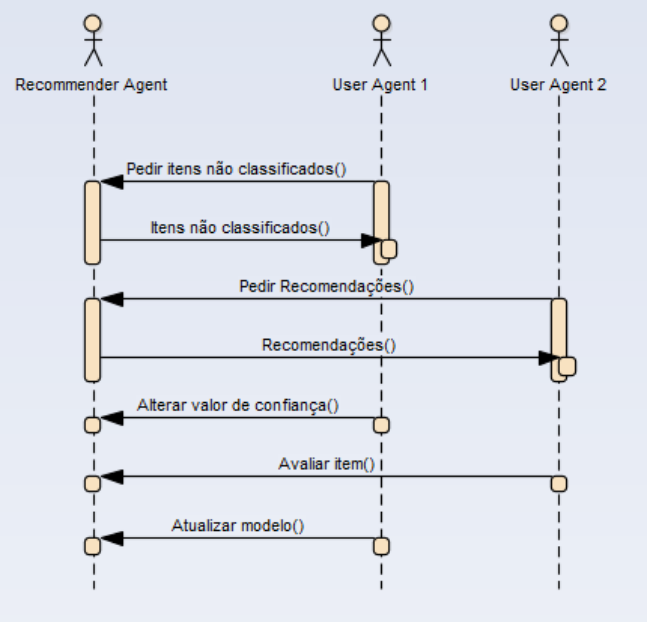

Figura 6. Diagrama de sequência da aplicação, ilustrando os principais pedidos/respostas associadas com dois agentes do tipo 'utilizador'.

3.3 LibRec

'LibRec' [6] é uma biblioteca *open-source* implementada em *Java* para sistemas de recomendação. A biblioteca implementa diversos algoritmos de recomendação que se encontram em três grupos: algoritmos de previsão da avaliação de produtos (matriz $\hat{R}$), algoritmos de *ranking* de produtos e algoritmos mistos (que podem ser usados em ambas as situações) – ver figura 7. Estes algoritmos e as configurações destes podem ser escolhidos através de ficheiros de configuração. A 'LibRec' também disponibiliza uma série de métricas de avaliação que permitem facilmente comparar o desempenho de diferentes algoritmos num determinado *dataset*.

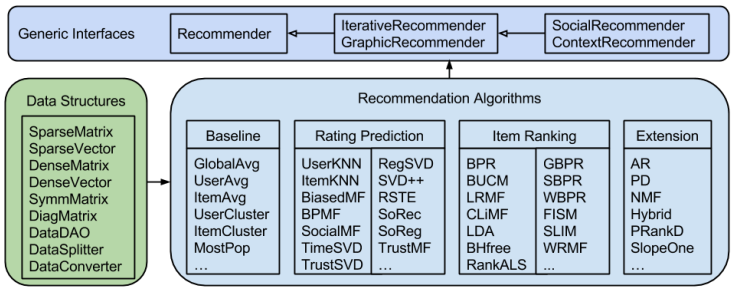

Figura 7. Diagrama de classe da biblioteca 'LibRec', listando os diferentes algoritmos disponibilizados.

3.4 GUI

Quando um agente 'utilizador' é criado, uma GUI é iniciada permitindo de forma gráfica enviar os pedidos enumerados anteriormente (ver figura 8). Os pedidos feitos pelo agente 'utilizador' serão relativos ao utilizador no *dataset* cujo ID foi fornecido na criação deste.

4 Experiências e Análise dos Resultados

4.1 Dataset

O *dataset* utilizado provém da rede social 'FilmTrust' [7] de recomendação de filmes. Esta rede social, apesar de já desativada, permitia a cada utilizador adicionar outros utilizadores como 'amigos' na rede [8] e atribuir-lhes uma 'pontuação' com base na sua confiança nestes para recomendação de filmes.

Este *dataset* possui 35497 *ratings* relativamente a 2071 filmes provenientes de 1508 utilizadores diferentes, a que corresponde uma matriz R em que 1.14% dos elementos se encontram preenchidos (de notar que este fenómeno é bastante

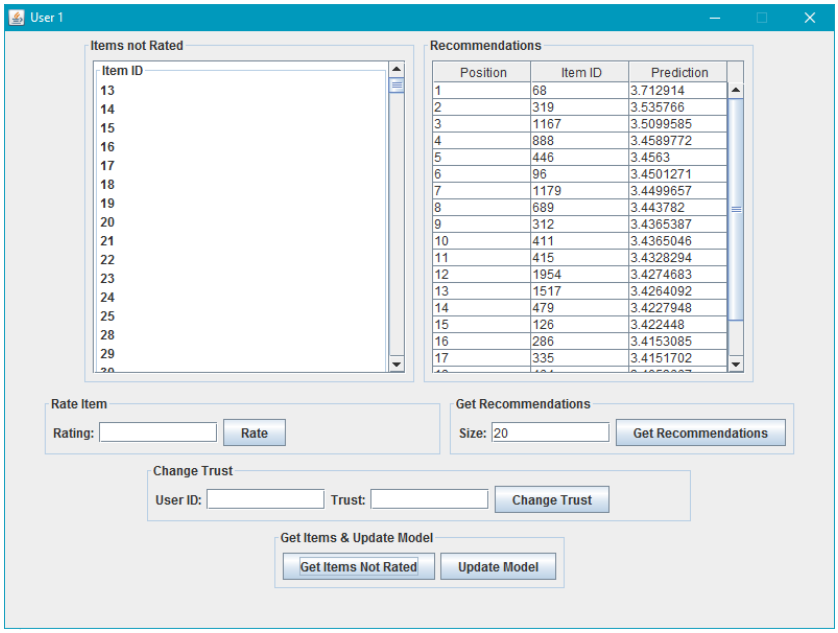

Figura 8. Interface gráfica da aplicação desenvolvida que permite executar as ações correspondentes aos agente 'utilizador'.

frequente neste tipo de *datasets*). São também fornecidas 1853 avaliações de confiança entre 1642 utilizares (neste *dataset* a confiança apenas toma o valor de 1).

4.2 Metodologia Experimental

De modo a investigar a potencial melhoria de precisão na previsão dos *ratings* no uso de informações de redes sociais em sistemas de recomendação, foram aplicados diferentes algoritmos de fatorização matricial (alguns fazendo uso desta informação e outros não) no *dataset* 'FilmTrust'.

Foram selecionados dois algoritmos que incorporam a matriz de confiança S (Social MF, SoRec – já abordados anteriormente) e dois algoritmos que apenas fazem uso da matriz de *ratings* R (NMF [9] – 'Non-negative matrix factorization', SVD++ [10]).

Todos os algoritmos foram executados recorrendo-se à biblioteca 'LibRec' (que usa o algoritmo de otimização *stochastic gradient descent*), tendo-se definido os parâmetros de configuração da tabela 1.

Os algoritmos foram avaliados usando *k-fold cross-validation* com $k = 10$ e a métrica RMSE (*root-mean-square error*) [1]:

Tabela 1. Valores dos parâmetros de configuração dos algoritmos testados.

Parâmetros	**Valor**
Número máximo de iterações	200
Learning rate (*alpha*)	0.001
Regularização (λ)	0.001
β (Social MF)	0.001

$$\sqrt{\frac{\sum_{(u,i)\in T}(\hat{R}_{u,i} - R_{u,i})^2}{|T|}} \tag{9}$$

onde:

T : Conjunto de pares (u, i) a usar para teste

4.3 Análise dos Resultados

Na tabela 2 estão presentes os resultados da aplicação da metodologia experimental descrita. Uma primeira análise da tabela permite concluir que os algoritmos que fazem uso da informação relativa à confiança entre utilizadores conseguiram resultados melhores do que os restantes para todos os valores de K, tendo-se obtido no melhor caso um RMSE de 0.8359 com o algoritmo SoRec para $K = 5$ (este algoritmo obteve os melhores resultados para qualquer um dos valores testados, apesar de muito semelhantes aos do algoritmo Social MF).

É possível observar-se que todos os algoritmos pioraram no valor do RMSE quando K aumentou, pelo que se conclui que o número de *features* latentes mais apropriado a usar-se neste *dataset* é de 5. De notar que para $K = 5$ o algoritmo SVD++ obtêm resultados bastante próximos dos dois algoritmos 'sociais' (Social MF e SoRec), mas para $K > 5$ o valor do erro sobe consideravelmente (quando comparado com o aumento nos restantes algoritmos). A eficácia dos algoritmos 'sociais' foi pouco afetada pela alteração do valor de K.

Tabela 2. Valor do RMSE para cada um dos algoritmos avaliados no *dataset* FilmTrust para diferentes valores de K (número de *features* latentes). O melhor valor do RMSE para cada um dos valores de K é apresentado a negrito.

Algoritmo	$K = 5$	$K = 10$	$K = 20$
Social MF	0.8396	0.8455	0.8519
SoRec	**0.8359**	**0.8381**	**0.8507**
NMF	0.9211	0.9540	0.9728
SVD++	0.8585	0.9017	0.9590

5 Conclusão e Trabalho Futuro

O presente artigo apresentou de forma geral o tópico dos sistemas de recomendação, com especial foco nas abordagens que fazem uso de informação social entre os utilizadores e da técnica de fatorização matricial. Foram feitas várias análises experimentais a vários algoritmos, que permitiram concluir que os algoritmos testados que fazem uso da confiança entre utilizadores obtêm resultados com maior precisão. Contudo, para $K = 5$ concluiu-se que não existe uma grande distinção entre os resultados com os algoritmos 'sociais' e o algoritmo SVD++.

Dado que muitas plataformas de serviços e *e-commerce* (como o Spotify) estão a ganhar cada vez mais funcionalidades sociais, é de esperar que os sistemas de recomendação sociais evoluam nos próximos anos.

Sâo propostos, como trabalho futuro, os seguintes pontos:

- Realizar mais experiências com mais algoritmos e variações nos seus parâmetros;
- Realizar experiências com um maior número de algoritmos;
- Repetir o procedimento experimental em outros *datasets* para confirmar os resultados obtidos no *dataset* FilmTrust – de notar que este *dataset* apenas possui um único valor possível de confiança (1), o que pode condicionar a precisão máxima alcançável pelos algoritmos 'sociais', pelo que este será um dos pontos mais importantes a explorar.

Referências

1. Xiwang Yang et al. ≪A survey of collaborative filtering based social recommender systems≫. Em: *Computer Communications* 41 (2014), pp. 1–10.
2. Gediminas Adomavicius e Alexander Tuzhilin. ≪Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions≫. Em: *Knowledge and Data Engineering, IEEE Transactions on* 17.6 (2005), pp. 734–749.
3. Jesús Bobadilla et al. ≪Recommender systems survey≫. Em: *Knowledge-Based Systems* 46 (2013), pp. 109–132.
4. Christian Desrosiers e George Karypis. ≪A comprehensive survey of neighborhood-based recommendation methods≫. Em: *Recommender systems handbook*. Springer, 2011, pp. 107–144.
5. Albert Yeung. *Matrix Factorization: A Simple Tutorial and Implementation in Python*. 2010. URL: `http://www.quuxlabs.com/blog/2010/09/matrix-factorization-a-simple-tutorial-and-implementation-in-python` (acedido em 14/06/2016).
6. Guibing Guo et al. ≪Librec: A java library for recommender systems≫. Em: *Posters, Demos, Late-breaking Results and Workshop Proceedings of the 23rd International Conference on User Modeling, Adaptation and Personalization*. 2015.

7. G. Guo, J. Zhang e N. Yorke-Smith. ≪A Novel Bayesian Similarity Measure for Recommender Systems≫. Em: *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*. 2013, pp. 2619–2625.
8. Jennifer Golbeck, James Hendler et al. ≪Filmtrust: Movie recommendations using trust in web-based social networks≫. Em: *Proceedings of the IEEE Consumer communications and networking conference*. Vol. 96. Citeseer. 2006, pp. 282–286.
9. Daniel D Lee e H Sebastian Seung. ≪Algorithms for non-negative matrix factorization≫. Em: *Advances in neural information processing systems*. 2001, pp. 556–562.
10. Yehuda Koren. ≪Factorization meets the neighborhood: a multifaceted collaborative filtering model≫. Em: *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2008, pp. 426–434.