Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal Catarinense
Câmpus Videira

Iury Krieger

UMA API WEB ORIENTADA A METADADOS COMO SERVIÇO DE RECOMENDAÇÃO HÍBRIDA

Orientador: Msc. Tiago Heineck

Coorientador: Msc. Wanderson Rigo

Videira - Santa Catarina

2017

Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal Catarinense
Câmpus Videira

Iury Krieger

UMA API WEB ORIENTADA A METADADOS COMO SERVIÇO DE RECOMENDAÇÃO HÍBRIDA

Trabalho de conclusão de curso submetido ao Instituto Federal Catarinense - Câmpus Videira como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação

Orientador: Msc. Tiago Heineck

Coorientador: Msc. Wanderson Rigo

Videira - Santa Catarina
2017

Iury Krieger

UMA API WEB ORIENTADA A METADADOS COMO SERVIÇO DE RECOMENDAÇÃO HÍBRIDA

Trabalho de conclusão de curso submetido ao Instituto Federal Catarinense - Câmpus Videira como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação

Videira (SC), 16 de Maio de 2017

Msc. Tiago Heineck
Instituto Federal Catarinense

Msc. Wanderson Rigo
Instituto Federal Catarinense

BANCA EXAMINADORA

Msc. Marcelo Cendron
Instituto Federal Catarinense

Maurício Ferreira
Instituto Federal Catarinense

Resumo

Devido a expansão massiva de dados produzidos e disponíveis na Internet, os usuários estão cada vez mais sobrecarregados de informação, não sabendo distinguir informações realmente úteis. Para sanar este problema, os sistemas de recomendação visam recomendar os itens mais úteis a cada usuário, através de técnicas de machine learning. Tais técnicas visam prever a avaliação de um usuário a um item, baseando-se nas avaliações já conhecidas. Este trabalho propõe o desenvolvimento de uma API Web de código aberto que recomenda itens a usuários, fazendo uso de um sistema de recomendação híbrido que analisa as estruturas pré definidas e proporciona recomendações, baseando-se nos metadados fornecidos, através do conteúdo do item e da filtragem colaborativa de usuários. Tal sistema poderá processar suas recomendações utilizando a GPU, minimizando o tempo da requisição de recomendação e consequentemente aumentando a eficiência da aplicação. Dessa forma é possível fornecer um serviço multi-propósito desprendido de qualquer ambiente e linguagem de programação, trazendo uma visão mais transparente dos sistemas de recomendação aos desenvolvedores.

Palavras-chaves: Sistemas de Recomendação. Aprendizado de Máquina. Metadados. Computação em GPU.

Abstract

Due to the massive expansion of data produced and available on the Internet, users are increasingly overloaded with information, not knowing how to distinguish which is really useful. To remedy this problem, recommendation systems aim to recommend the most useful items to each user through machine learning techniques. These techniques are intended to predict a user's rating of an item, based on previously known rating. This work proposes the development of an open-source Web API that recommends items to users, making use of a hybrid recommendation system that analyzes the pre-defined structures and provides recommendations, based on the metadata provided, through item content and collaborative filtering. Such a system can process its recommendations using the GPU, minimizing the time of the recommendation request and consequently increasing the application efficiency. Therefore is possible to provide a multi-purpose service detached from any environment and programming language, bringing a more transparent view of recommendation systems to developers.

key-words: Recommender Systems. Machine Learning. Metadata. GPU Computing.

Lista de Quadros

Lista de ilustrações

Lista de abreviaturas e siglas

IA	*Inteligência Artificial*
HTTP	*Hypertext Transfer Protocol*
API	*Application Program Interface*
REST	*Representational State Transfer*
JSON	*Javasript Object Notation*

Sumário

1 INTRODUÇÃO

Com o avanço crescente do campo tecnológico, os computadores vêm desempenhando tarefas antes incumbidas à seres humanos. O poder de computação provou-se muito eficaz ao desempenhar tarefas que possuíssem um padrão possível de se expressar através de um algoritmo, mais ainda, se este padrão fosse repetitivo.

Logo os computadores começaram a desempenhar funções nas mais diversas áreas, desde cálculos matemáticos à manipulação de imagens. Atualmente, das funções desempenhadas pelos computadores, a mais difícil de se reproduzir com precisão é o padrão de raciocínio humano.

Alguns autores defendem que para que um computador atinja tal nível, seria necessário que o mesmo possuísse consciência, assim como os seres humanos. Outros defendem que o raciocínio humano não consegue ser reproduzido, apenas emulado, devido à impossibilidade de se programar uma consciência computacional. Tal área de estudo, que tem como o foco o desenvolvimento de sistemas computacionais rumo a proximidade do método humano, chama-se inteligência artificial (RUSSELL; NORVIG, 2004; COPPIN, 2015).

Possível ou não, é inegável o avanço da inteligência artificial desde seu início nos primórdios da computação. Algumas tarefas, tais como a atribuição de uma consciência a um sistema computacional, deixaram de ser o foco da área, uma vez que não possuímos a tecnologia para construir sistemas muito mais complexos que os atuais (RUSSELL; NORVIG, 2004).

Entretanto, a inteligência artificial encontrou-se muito eficaz em outras áreas do método humano, tais como o aprendizado, um dos segmentos mais importantes da área, dentro da inteligência artificial chamado de aprendizado de máquina (*machine learning*) (COPPIN, 2015).

Desde os anos 90 a preocupação com o armazenamento e a expansão massiva de dados produzidos já existia, prevendo que usuários ficariam sobrecarregados de informação, não sabendo distinguir o que seria realmente útil (HILL *et al.*, 1995; ADOMAVICIUS; TUZHILIN, 2005). Na época, uma comunidade virtual de avaliação foi proposta para proporcionar aos usuários o mínimo de esforço ao encontrar informações úteis. Com a evolução da inteligência artificial e das técnicas de machine learning, este trabalho de avaliação e recomendação, antes feito por uma comunidade, hoje é atribuído aos sistemas de recomendação (HILL *et al.*, 1995).

Sistemas de recomendação (RSs) são ferramentas de software e técnicas que provém

sugestões de artefatos à usuários. Estes artefatos são definidos como os objetos de valor à serem recomendados (RICCI; ROKACH; SHAPIRA, 2011). Atualmente, o interesse em tais sistemas se mantém alto, devido a abundância de aplicações práticas (ADOMAVICIUS; TUZHILIN, 2005), exemplificadas nos casos de *E-commerce* por Schafer, Konstan e Riedl (2001), além de Linden, Smith e York (2003), onde são amplamente utilizados.

Desta forma, sistemas de recomendação vem sendo desenvolvidos para a resolução do problema descrito nas mais diversas áreas (BENNETT; LANNING *et al.*, 2007; GAVALAS *et al.*, 2014), desde aplicações hoteleiras como o TripAdvisor até aplicações de entretenimento como a Netflix, além da sua origem nos *E-commerces* citados anteriormente. Muitos destes sistemas são casos de RSs aplicados a itens e finalidades específicas (HUANG *et al.*, 2002; BROZOVSKY; PETRICEK, 2007), onde todo o motor de recomendação segue uma abordagem baseada no padrão que lhe foi dado.

Por outro lado, ao observar aplicações web de sistemas de recomendação, verifica-se a existência de soluções em forma de APIs, tais como o Google Cloud Platform e o Microsoft Cognitive Services, fornecidas como serviços transparentes. Entretanto, estas soluções proprietárias não são incorporadas a aplicação, mas sim utilizadas como serviços externos, dificultando a personalização.

Para sanar estes problemas, este trabalho propõe o desenvolvimento de uma API que proporcione uma visão mais transparente dos sistemas de recomendação, permitindo ao usuário desfrutar das funcionalidades, sem a necessidade de um profundo conhecimento dos detalhes que compõem as diferentes técnicas de recomendação, além dos problemas decorrentes do uso de cada uma das técnicas. Além disso, tal tecnologia será fornecida como um serviço de código aberto, podendo ser utilizada em qualquer ambiente.

Este trabalho está dividido em seis seções. A segunda seção apresenta o referencial teórico necessário para o entendimento total do escopo do trabalho. Na terceira seção são apresentadas as principais características do trabalho proposto, além de compará-lo com outros trabalhos relacionados. Em seguida, a quarta seção apresenta a metodologia a ser utilizada para realização do trabalho proposto na seção anterior. Mais à frente, na seção cinco, será abordado o cronograma a ser empregado para a realização do trabalho e, por fim, na sexta seção são apresentadas as considerações finais.

1 Objetivos

1.1 Objetivo Geral

Desenvolver uma API web de código aberto para recomendação híbrida de itens a usuários.

1.2 Objetivos Específicos

- Fornecer uma documentação das funcionalidades visando futura colaboração da comunidade e utilização por outros desenvolvedores.
- Proporcionar a recomendação das propriedades relevantes através das estruturas de metadados fornecidas.
- Possibilitar o processamento paralelo das recomendações em GPU.

2 Metodologia

A metodologia deste trabalho está dividida em três seções. Primeiramente serão implementadas todas as funcionalidades descritas anteriormente. Mais à frente, será feita a validação das funcionalidades implementadas e da eficácia das recomendações. Por fim, serão feitos os ajustes e correções necessárias de acordo com o resultado da validação das funcionalidades implementadas.

2.1 Implementação

Inicialmente serão implementados os algoritmos de recomendação híbrida, incluindo o processamento dos metadados fornecidos. Os algoritmos de recomendação resumem a eficácia da API e devem consumir a maior parte do tempo de desenvolvimento.

Ao completar a implementação das técnicas híbridas de sistemas de recomendação, serão implementadas as demais funcionalidades da API. Serão consideradas a identificação e entrada dos metadados, além do formato dos dados de saída.

Ao fim do desenvolvimento, o código será adaptado para processamento na GPU através de diretivas de compilação e bibliotecas especializadas.

2.2 Validação

Assim que a API esteja em um grau considerado funcional, será feita a validação da eficácia ao recomendar as estruturas fornecidas através de grupos de testes definidos, uma técnica amplamente utilizada na validação de técnicas de machine learning.

A validação será feita utilizando um grupo separado dos dados utilizados para testes, confrontando as recomendações feitas com o resultado esperado. Através desses resultados é medida a acurácia de um sistema de recomendação, métrica utilizada como medida de eficiência entre os diferentes métodos utilizados.

2.3 Ajustes e Correções

Por fim, serão feitos os ajustes e correções de erros recolhidos ao longo do processo, além de testar as funcionalidades e a utilização da API como um todo. A documentação será feita durante boa parte de todo o processo e, neste caso em específico, possui um foco especial, uma vez que o princípio da API é que a mesma seja utilizável por outros desenvolvedores, além de possibilitar a contribuição da comunidade.

3 Trabalhos Relacionados

Tendo como base as técnicas descritas acima, existem trabalhos como os apresentados por Guo *et al.* (2015), que abordam as técnicas em forma de biblioteca Java a ser incluída nos projetos. Esta abordagem torna a utilização mais simples devido ao fato do usuário poder utilizar apenas as funcionalidades da biblioteca, preocupando-se com o formato de entrada e saída dos dados, não com o processo de recomendação em si. Outra abordagem interessante é a proposta por Brozovsky e Petricek (2007) ao construir uma biblioteca C# multi-propósito, focando na recomendação de itens com base na avaliação em um esquema de *rating* (de uma a cinco estrelas), ou com base apenas em itens com avaliação positiva.

Do mesmo modo que a biblioteca desenvolvida por Brozovsky e Petricek (2007), a API proposta neste trabalho também visa ser multi-propósito e distribuída como código aberto pela licença pública GNU (**GPL**), porém, fornecendo tais funcionalidades como um serviço web independente de linguagem de programação, o que não acontece nos exemplos apresentados.

Além dos trabalhos apresentados, Nascimento (2013) aborda os sistemas de recomendação com uma perspectiva semelhante a este trabalho, focando mais no ganho de desempenho ao processar o método de filtragem colaborativa na GPU. Este trabalho não tem seu foco em desempenho, mas sim em uma proposta de **recomendação genérica**, que forneça recomendações a quaisquer modelos de usuários e itens através do método híbrido.

2 REFERENCIAL TEÓRICO

1 Aprendizado de Máquina

Um dos segmentos da inteligência artificial com grande importância na atualidade é o aprendizado de máquina. Responsável pela construção de agentes capazes de, a partir de uma coleção de pares de entrada e saída, aprender uma função que prevê a saída para novas entradas. Tais agentes são definidos como tudo que pode perceber seu ambiente através de sensores, além de atuar sobre o mesmo através de atuadores. Em outras palavras, o aprendizado de máquina resume-se em técnicas que proporcionam a um algoritmo a capacidade de melhorar seu desempenho de forma automática, através do conhecimento obtido pelas entradas existentes (COPPIN, 2015).

Dessa forma, considera-se que um agente está aprendendo se melhorar o seu desempenho nas tarefas para que foi designado, a partir de suas observações sobre o mundo. Este aprendizado proporciona às técnicas de *machine learning* a capacidade evolutiva, uma vez que é possível não só responder as entradas do mundo exterior como também tirar conclusões sobre as mesmas, melhorando cada vez mais a natureza da solução (RUSSELL; NORVIG, 2004).

Conforme apresentado por Carbonell, Michalski e Mitchell (1983), devido a capacidade de, além de solucionar problemas, melhorar automaticamente o desempenho da solução, os sistemas de aprendizagem tem suas aplicações nas mais diversas áreas, tais como agricultura, educação, sistemas especialistas de alta performance, reconhecimento de imagem, programação, etc. Através de um apanhado das aplicações nas áreas de utilização, Carbonell, Michalski e Mitchell (1983) dividem o campo de aprendizado do *machine learning* em três partes:

- **Estudos orientados à tarefa (*Task-oriented studies*):** composto pelo desenvolvimento e análise de sistemas de aprendizagem visando melhorar a performance na solução de determinadas tarefas.
- **Simulação cognitiva (*Cognitive simulation*):** formado pela investigação e simulação do processo de aprendizagem humano.
- **Análise teórica (*Theoretical analysis*):** exploração teórica do espaço de possíveis processos de aprendizado.

Analisando a taxonomia proposta por Carbonell, Michalski e Mitchell (1983), pode-se idenficar que o escopo deste trabalho encontra-se nos estudos orientados à tarefa, onde

o propósito é a melhoria da performance, neste caso através de recomendações orientadas à metadados.

Como exemplo do uso das técnicas de *machine learning*, Sebastiani (2002) apresenta um algoritmo de categorização de texto que, a partir de um conjunto de documentos pré-classificados (entradas), constrói um classificador para novos documentos (novas entradas). Outro exemplo, apresentado por Pang, Lee e Vaithyanathan (2002), reforça a ideia de melhora de desempenho para novas entradas através de um padrão aprendido a partir de entradas já existentes. Através de dados sobre avaliações de filmes, pode-se perceber que, mesmo as técnicas padrão de *machine learning*, acabam superando os patamares humanos na classificação de sentimentos.

2 Sistemas de Recomendação

Como ramificação do aprendizado de máquina, os sistemas de recomendação (RSs) são técnicas de software que provém sugestões a usuários de itens que os mesmos possam querer utilizar (RESNICK; VARIAN, 1997; SCHAFER; KONSTAN; RIEDL, 1999). Desta forma, recomendações seriam, em sua forma mais simples, rankings de itens, tais como os utilizados na maioria das soluções de produtos (livros mais lidos, filmes mais assistidos, etc.) (RICCI; ROKACH; SHAPIRA, 2011). O que os RSs trazem de novo é a tentativa de predizer, através da filtragem colaborativa ou da similaridade de conteúdo, qual o ranking mais adequado de produtos ou serviços a um usuário. A filtragem colaborativa, termo cunhado por Resnick e Varian (1997), recomenda itens baseando-se nos relacionamentos do usuário. Por outro lado, a similaridade de conteúdo baseia-se no conteúdo de itens já avaliados pelo usuário. Tais dados podem ser coletadas de forma explícita, na forma de perguntas diretas e avaliações do usuário sobre os itens, ou de forma interpretativa, inferindo sobre ações tomadas pelo usuário e atribuindo peso a elas.

Mais formalmente, os sistemas de recomendação podem ser descritos matematicamente da seguinte forma: sendo C o conjunto de todos os usuários e S o conjunto de todos os itens que podem ser recomendados, tanto o espaço S como o espaço C podem ser extremamente grandes, batendo os milhões de usuários e itens (ADOMAVICIUS; TUZHILIN, 2005; GOMEZ-URIBE; HUNT, 2016). Dessa forma, tem-se u como a função de utilidade de um item s para um usuário c. A função u utiliza-se do conjunto ordenado R, descrito como $C \times S \rightarrow R$, para encontrar o item $s \in S$ com a maior utilidade para o usuário c. Um exemplo de como as preferências são armazenadas no espaço de avaliações C x S pode ser visto no quadro 1.

De acordo com o quadro 1, o símbolo "$\emptyset$"representa os filmes ainda não avaliados pelos usuários. Estes itens, por sua vez, são os alvos das técnicas de recomendação que tentam predizer a avaliação de um usuário. Uma vez que o motor de recomendação pode

Quadro 1 – Exemplo de matriz de recomendações a filmes

	K-PAX	Life of Brian	Memento	Notorious
Alice	4	3	2	4
Bob	∅	4	5	5
Cindy	2	2	4	∅
David	3	∅	5	2

Fonte: Adomavicius e Tuzhilin (2005)

predizer as avaliações de um usuário, pode-se recomendar ao mesmo apenas os N itens com a maior avaliação estimada (ADOMAVICIUS; TUZHILIN, 2005).

Como consequência da importante participação dos sistemas de recomendação em sites com um grande número de público, tais como Netflix, eBay e Amazon.com, os mesmos tornaram-se ferramentas poderosas (SCHAFER; KONSTAN; RIEDL, 1999) e são considerados os propulsores de várias estatísticas, entre elas: o aumento da satisfação dos usuários, devido a precisão das recomendações; o aumento da fidelidade dos usuários, devido ao aumento de precisão quanto maior for a interação do usuário com o site; a aumento da capacidade do próprio serviço em entender melhor as intenções de seu público (RICCI; ROKACH; SHAPIRA, 2011). Tendo em vista o crescimento do número de aplicações que utilizam sistemas de recomendação e da variedade de soluções utilizadas em grandes sites, torna-se notável a importância dos mesmos.

Como próximo passo na evolução dos sistemas de recomendação, Adomavicius e Tuzhilin (2015) propõem que os RSs, além de considerarem a similaridade entre perfis, devem estar cientes do contexto da avaliação do usuário ao construírem o modelo de perfil. Chamados de sistemas cientes de contexto, estes sistemas de recomendação devem diferenciar a ação que o usuário toma ao apenas analisar um item (filme, produto, etc.), não necessariamente indicando que itens parecidos devem ser recomendados no futuro, da ação tomada ao consumir um item (comprar, assistir, etc.). A partir dessa distinção de contexto, os RSs poderiam atribuir pesos diferentes para cada ação, podendo assim fazer recomendações mais precisas.

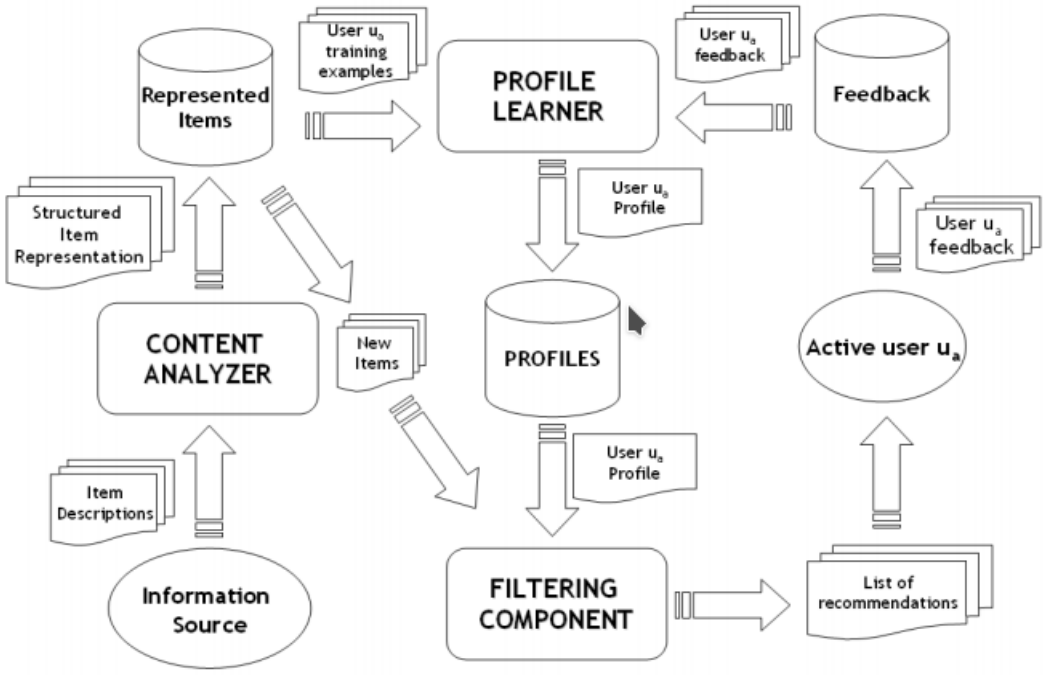
A seguir serão apresentados as diferentes técnicas dos sistemas de recomendação, além de qual técnica será utilizada por este trabalho e seus diferentes métodos através de algoritmos. Devido a existência de inúmeras técnicas e métodos de recomendação, este trabalho abordará apenas as técnicas necessárias para entendimento do mesmo, aprofundando-se apenas nos métodos que compõem a técnica utilizada.

2.1 Método Baseado em Conteúdo

Sistemas de recomendação que implementam o método baseado em conteúdo (*content-based*) analisam um conjunto de documentos/descrições de itens previamente avaliados pelo usuário, construindo um modelo dos interesses baseando-se nas características dos itens avaliados (MLADENIC, 1999; ADOMAVICIUS; TUZHILIN, 2005; LOPS; GEMMIS; SEMERARO, 2011). Este modelo serve para ser cruzado com o conteúdo de outros itens ainda não avaliados pelo usuário. Quanto maior o grau de semelhança entre o modelo do usuário e as características do item, maior a probabilidade do mesmo ter interesse.

Para que o modelo de interesses do usuário seja criado e confrontado com outros conteúdos ainda não avaliados, são necessários três atores principais que dividem a recomendação baseada em conteúdo: **analisador de conteúdo**, **aprendiz de perfis** e **componente de filtragem** (LOPS; GEMMIS; SEMERARO, 2011). A estrutura completa destes agentes pode ser vista na figura 1.

Figura 1 – Arquitetura de um sistema baseado em conteúdo



Fonte: Lops, Gemmis e Semeraro (2011)

Note que na figura 1, a primeira parte do processo começa com o **analisador de conteúdo** (*content analyzer*), transformando dados não estruturados em estruturas de atributos e características (LOPS; GEMMIS; SEMERARO, 2011; MLADENIC, 1999), armazenando-as no repositório de itens representados (*represented items*). Para a construção e atualização do perfil de interesses do usuário ativo (representado na figura 1 por u_a), as

avaliações do usuário para novos itens são armazenadas no repositório de feedback. O tipo de avaliação depende de cada aplicação, podendo ser expressado de forma **explícita**, como as avaliações binárias (*like/dislike*) e avaliações em forma de rating (0 a 5; 1 a 5 estrelas) utilizadas em muitos sites, ou mesmo por avaliações **implícitas**, onde uma ação sobre um item (seleção, por exemplo) possui um peso atribuído (PAZZANI; BILLSUS, 2007).

De posse do repositório de itens representados, o **aprendiz de perfis** varre os itens I_k do usuário ua em prol de construir o conjunto treinamento TR_a. O conjunto de treinamento é um conjunto de pares $\{I_k, r_k\}$, onde r_k é a avaliação dada pelo usuário ua a representação do item I_k. Após a construção do conjunto de treinamento TR_a, o **aprendiz de perfis** aplica algoritmos de aprendizagem supervisionada para gerar o modelo de interesses do usuário u_a. Os modelos de interesses são armazenados no repositório de perfis (representado na figura 1 por *profiles*) para uso futuro pelo **componente de filtragem**.

Quando a representação de um novo item é adicionada ao conjunto de itens representados, o componente de filtragem prediz se o mesmo será de interesse do usuário u_a, atraveś da comparação entre os atributos e características do novo item e o modelo de interesses do usuário. Em consequência, o componente de filtragem ranqueia os itens com os maiores potenciais de interesse, agrupando-os em uma lista de recomendações L_a e apresentando-a ao usuário u_a. Dessa forma o usuário ua pode prover novas avaliações (**feedbacks**) dos itens da lista L_a, fazendo com que o aprendiz de perfis atualize seu modelo de interesses através da reconstrução do conjunto de treinamento TR_a (LOPS; GEMMIS; SEMERARO, 2011).

Atualmente Pazzani e Billsus (2007) apresentam que, devido ao grande crescimento de informação disponível para treinamento, os métodos atuais reduzem o conjunto de treinamento para algumas centenas de linhas, porém altamente relevantes (através de técnicas como o TF-IDF [1]). Dessa forma, por mais que as bases de dados aumentem, o conjunto de treinamento se mantém relevante e não é necessário percorrer todo o conjunto ordenado R.

2.2 Método Baseado em Colaboração

O método de filtragem colaborativa (*collaborative-based* - CF) baseia-se no processo de avaliar itens através da opinião de outras pessoas. Tal processo, que começou com a filtragem da natureza de repositórios de texto, passou a ser mais informal, abrangendo até listas de discussão e arquivos de *e-mail*. No começo, usuários tinham que acessar sites específicos, tais como o MovieLens, para receberem recomendações de filmes. Conforme os sistemas baseados em CF foram se popularizando, os sites começaram a utilizar estes

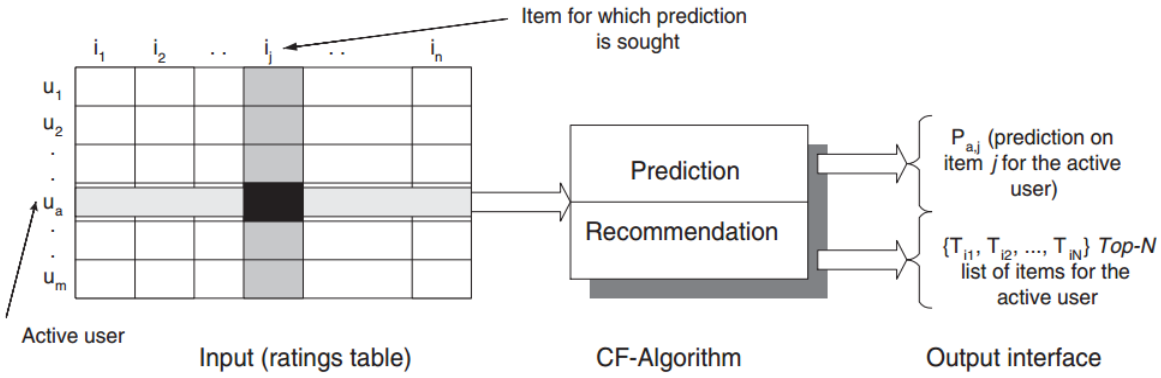
[1] Frequência do termo inverso da frequência nos documentos. Medida estatística para indicar a importância de uma palavra de um documento em relação a uma coleção de documentos. É frequentemente usada na mineração de dados.

sistemas para adequar seu conteúdo para cada usuário (SCHAFER *et al.*, 2007).

Assim como os sistemas baseados em conteúdo, sistemas de filtragem colaborativa também levam em consideração as avaliações de itens (mesmo que de outros usuários similares), através dos métodos de avaliação já descritos. Segundo Adomavicius e Tuzhilin (2005), a diferença entre estes dois processos existe pelo fato de que a utilidade $u(c, s)$ de um item s a um usuário c é medida não pela utilidade $u(c, s_i)$ dos itens s_i similares ao item s, mas sim pela utilidade $u(c_j, s)$ do item s baseado nos usuários c_j **similares** ao usuário c. Em outras palavras, na filtragem colaborativa, os itens considerados úteis a um usuário são os itens úteis a usuários similares a ele.

Partindo desta premissa, Sarwar *et al.* (2001) abordam os sistemas de filtragem colaborativa a partir do seguinte cenário: uma lista de **m** usuarios $U\{u_1, u_2, \ldots, u_m\}$ e uma lista de **n** itens $I\{i_1, i_2, \ldots, i_n\}$. Cada usuário u_i possuindo uma lista Iu_i de itens, avaliados ou não. Conforme na figura 2, o algoritmo de filtragem colaborativa (**CF**) opera sobre a matriz de avaliações $n \times m$.

Figura 2 – Processo de recomendação colaborativa.



Fonte: Sarwar *et al.* (2001)

De posse da matriz $n \times m$, o algoritmo **CF** faz a predição/recomendação ao usuário corrente, demonstrado na figura 2 por u_a. O usuário u_a é visto pelo algoritmo como o alvo atual para o qual serão feitas as predições/recomendações. Sarwar *et al.* (2001) também especificam a predição como um valor numérico que expressa a probabilidade prevista do item ser de interesse do usuário u_a, sendo este um item ainda não pertencente ao conjunto de Iu_a. Por outro lado, a recomendação é descrita como uma lista de **N** itens, cada item I_r dentre os itens com a maior probabilidade de utilidade ao usuário ua e ainda não avaliados pelo mesmo. Esta forma de recomendação também é conhecida como recomendação **Top-N** (ADOMAVICIUS; TUZHILIN, 2005).

Diferentemente do método baseado em conteúdo, a filtragem colaborativa não possui apenas uma abordagem. Tanto Sarwar *et al.* (2001 apud BREESE; HECKERMAN;

KADIE, 1998) quanto Adomavicius e Tuzhilin (2005) dividem a filtragem colaborativa em duas ramificações:

- **Baseada em memória (*memory-based*)**: implica na utilização de toda a matriz $n \times m$ para obter um conjunto de usuários vizinhos (*neighbor-users*), ou seja, usuários que tendem a avaliar diferentes itens similarmente ou itens similares diferentemente ao usuário u_a. Ao obter o conjunto, os métodos baseados em memória combinam as preferências dos usuários, fornecendo uma recomendação Top-N ao usuário u_a.
- **Baseada em modelo (*model-based*)**: ao invés de utilizar toda a matriz $n \times m$, esta técnica constrói um modelo das avaliações de cada usuário através de diferentes técnicas de *machine learning*, tais como modelos de *cluster* e redes Bayesianas. Devido a complexidade destas técnicas e das mesmas não pertencerem ao escopo da solução apresentada neste trabalho , não abordaremos mais a fundo seu funcionamento.

Dessa forma, sistemas de filtragem colaborativa podem ser usados nos casos em que se deseja recomendar itens úteis a um usuário ou fornecer uma previsão ao usuário da probabilidade do mesmo gostar de um item em particular. Além disso, é possível recomendar ao usuário não só itens, mas também usuários ou grupos de usuários que o mesmo possa gostar, o que não é possível nos sistemas baseados em conteúdo (SCHAFER *et al.*, 2007).

Considerando tais utilidades, tanto Schafer *et al.* (2007) quanto Adomavicius e Tuzhilin (2005) expõem os sistemas de recomendação baseados em conteúdo e colaborativos como complementares, uma vez que o método baseado em conteúdo prediz a relevância de itens sem avaliações, enquanto o método colaborativo prediz a relevância através de recomendações alheias. A união destas técnicas, em prol de maximizar a eficiência e compensar as limitações (seção 2.2.4), deu origem ao **método híbrido** que será abordado a seguir.

2.3 Método Híbrido

Sistemas de recomendação híbridos seriam quaisquer sistemas que combinam múltiplas técnicas de recomendação para produzir seu resultado (BURKE, 2002; BURKE, 2007). Como apresentado por Adomavicius e Tuzhilin (2005), as técnicas de recomendação possuem limitações de acordo com a abordagem utilizada. Sendo assim, é possível combinar diferentes técnicas para obter o desempenho e precisão desejadas.

Como pode ser visto no quadro 2, Burke (2002) apresenta uma série de métodos de recomendação além dos mais comuns abordados neste trabalho. Estes métodos, combinados entre si, podem gerar sistemas híbridos categorizados da seguinte forma:

Quadro 2 – Técnicas de recomendação.

Technique	Background	Input	Process
Collaborative	Ratings from **U** of items in **I**.	Ratings from **u** of items in **I**.	Identify users in **U** similar to **u**, and extrapolate from their ratings of **i**.
Content-based	Features of items in **I**	**u**'s ratings of items in **I**	Generate a classifier that fits **u**'s rating behavior and use it on **i**.
Demographic	Demographic information about **U** and their ratings of items in **I**.	Demographic information about **u**.	Identify users that are demographically similar to **u**, and extrapolate from their ratings of **i**.
Utility-based	Features of items in **I**.	A utility function over items in **I** that describes **u**'s preferences.	Apply the function to the items and determine **i**'s rank.
Knowledge-based	Features of items in **I**. Knowledge of how these items meet a user's needs.	A description of **u**'s needs or interests.	Infer a match between **i** and **u**'s need.

Fonte: Burke (2002)

- **Atribuição de peso (*Weighted*):** Consiste na atribuição de peso para cada um dos métodos empregados no sistema híbrido. Baseado no histórico de acertos entre um método e outro, é possível ajustar o peso de cada um, dando um peso maior ao método atualmente mais eficiente.

- **Escalonamento (*Switching*):** Consiste na utilização de um critério pré-definido para escolher qual método será utilizado no momento. Por exemplo, se o método colaborativo não fornecer uma recomendação com confiança suficiente, o sistema pode trocar para o método baseado em conteúdo.

- **Misto (*Mixed*):** Consiste em usar tanto recomendações de um método quanto de outro, apresentando os resultados de ambos ao usuário.

- **Combinação de características (*Feature Combination*):** Consiste em utilizar a informação colaborativa apenas como características adicionais no conjunto utilizado pelo método baseado em conteúdo.

- **Cascata (*Cascade*):** Este método em especial consiste em refinamento por estágio, ou seja, o primeiro método é utilizado para gerar um conjunto de recomendações, enquanto o segundo é responsável por refinar o conjunto gerado e assim por diante.

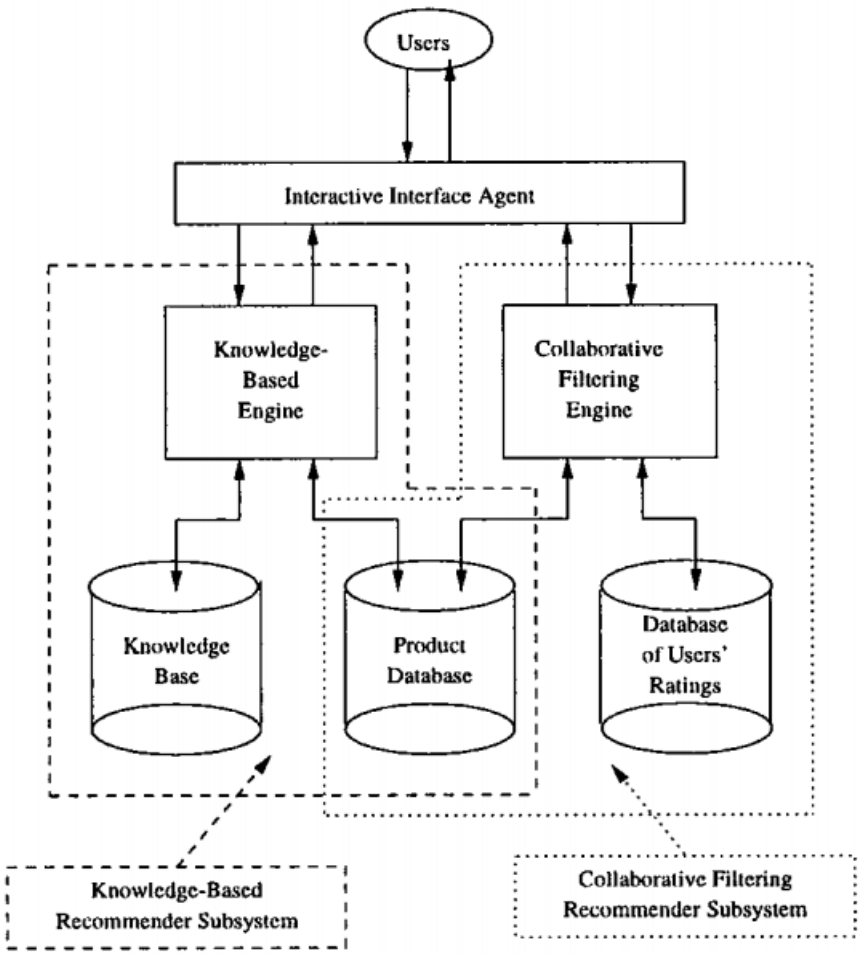
- **Aumento de Recursos (*Feature Augmentation*):** Esta técnica utiliza a recomendação gerada pelo primeiro método como informação para o processamento do segundo método.

- **Meta-nível (*Meta-level*):** Consiste em utilizar o modelo de saída de um método como entrada para o outro. Diferente do aumento de recursos, nesta técnica todo o modelo gerado pelo primeiro método é utilizado.

Tendo em vista a taxonomia apresentada por Burke (2002), nota-se que a recomendação híbrida não refere-se ao funcionamento das recomendações, mas sim sobre como os diferentes métodos **interagem entre si**. Esta interação pode ser insensível à ordem, nos casos de métodos como a atribuição de peso, misto, escalonamento e combinação de características. Já nas outros métodos apresentados, a ordem de execução dos métodos de recomendação alteram o resultado final, uma vez que a saída de um, direta ou indiretamente é a entrada de outro.

Por exemplo, Tran e Cohen (2000) apresentam uma arquitetura híbrida, utilizando os métodos baseado em colaboração (collaborative-based) e baseado em conhecimento (knowledge-based), ambos exemplificados através da arquitetura ilustrada na figura 3.

Figura 3 – Exemplo de arquitetura híbrida.



Fonte: Tran e Cohen (2000)

Como ilustrado na figura 3, a arquitetura descrita exemplifica um sistema híbrido de escalonamento (*switching*). Sendo assim, dependendo da situação atual, o sistema pode trocar entre a recomendação colaborativa e a baseada em conhecimento, visando prover melhores recomendações. Considerando que inicialmente a abordagem colaborativa não seria muito eficiente, enquanto a base de dados não possui muitos usuários com modelos conhecidos e não existem itens avaliados o suficiente, Tran e Cohen (2000) optaram por escalonar para o método baseado em conhecimento.

Através dessas limiares, toda vez que o usuário requisita uma recomendação, o agente de interface interativa (*interactive interface agent*) verifica se as mesmas já foram atendidas. Se sim, o agente utiliza a recomendação do método de filtragem colaborativa, se não, o método baseado em conhecimento é utilizado.

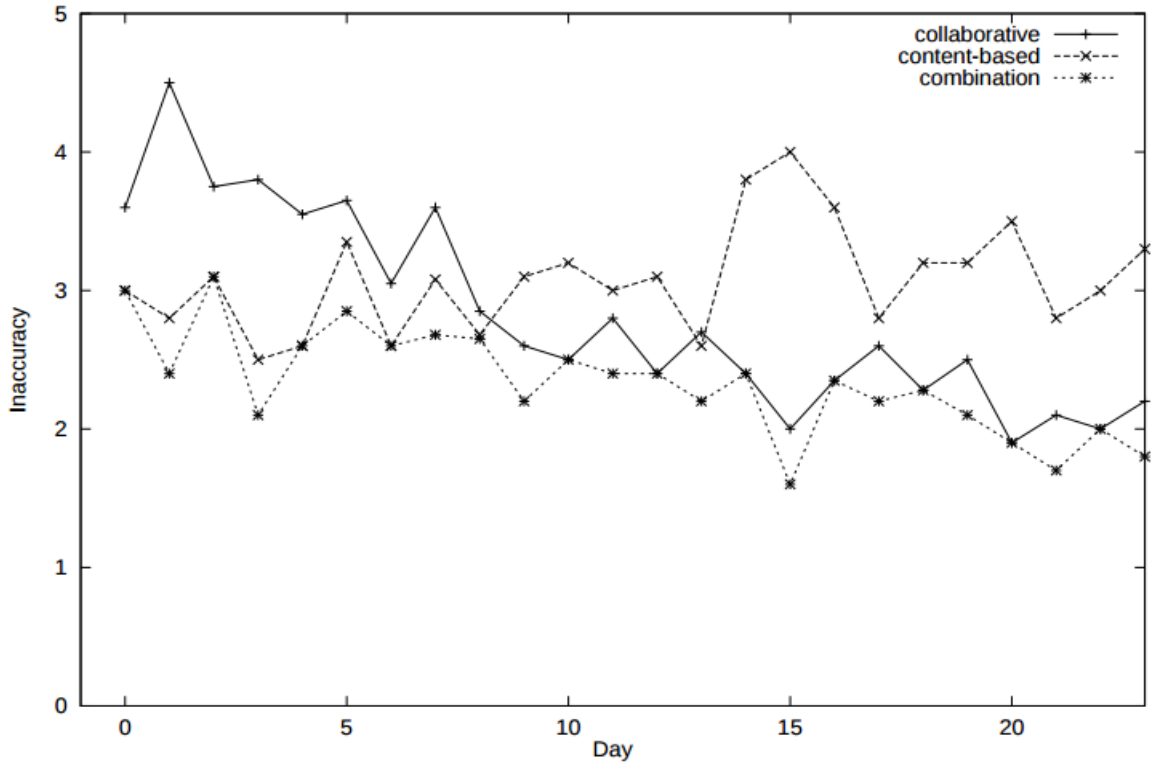
Tanto Balabanović e Shoham (1997) quanto Claypool *et al.* (1999) utilizam sistemas híbridos compostos de duas técnicas combinadas: **baseado em conteúdo** e **baseado em colaboração**. Dessa forma, é possível utilizar o método colaborativo para gerar o conjunto de **N** usuários vizinhos (*neighbor-users*) já descrita neste trabalho. A partir do conjunto gerado é aplicado o método baseado no conteúdo destes usuários próximos, aumentando a precisão da recomendação gerada.

Ao invés de se utilizar apenas um método, a utilização de sistemas híbridos pode trazer uma série de benefícios: ao executar recomendações baseadas em conteúdo, o sistema colaborativo pode lidar com novos usuários que ainda não tem seu modelo definido; torna-se possível fazer recomendações precisas a um usuário, mesmo que não existam usuários similares ao mesmo; pode-se recomendar itens não avaliados por nenhum dos usuários, cruzando o modelo dos mesmos com o conteúdo do item (BALABANOVIĆ; SHOHAM, 1997).

Como forma de verificar a eficácia do método híbrido em relação aos métodos utilizados de forma individual, Claypool *et al.* (1999) utilizam como métrica a inexatidão, sendo o termo referente a discrepância entre a recomendação obtida e o resultado esperado. A inexatidão dos métodos em relação a seu tempo de utilização pode ser visto através do resultado ilustrado na figura 4.

Analisando a figura 4 pode-se verificar que nos primeiros dias, a inexatidão do método colaborativo era maior devido a falta de completude no modelo dos usuários, construído por meio de usuários que ainda não avaliaram itens, ou de usuários que não se beneficiam da opinião de outros (CLAYPOOL *et al.*, 1999). Conforme o método colaborativo foi estabelecendo relações entre os usuários, este ficou mais preciso e o método baseado em conteúdo começou a ser menos viável. Porém, independente dos picos de inexatidão dos métodos separados mostrados na figura 4, quando combinados (**recomendação híbrida**), é possível notar uma constância muito maior, possuindo o mais baixo nível de inexatidão em todos os momentos.

Figura 4 – Inexatidão entre os métodos de recomendação.



Fonte: Claypool *et al.* (1999)

Em resumo os sistemas híbridos foram criados para unir técnicas de recomendação com objetivo de **compensar as limitações** apresentadas pela utilização dessas mesmas técnicas individualmente (BALABANOVIĆ; SHOHAM, 1997). Tais limitações e seus efeitos no resultado das recomendações serão abordadas na seção a seguir.

2.4 Limitações

Conforme apresentado por Adomavicius e Tuzhilin (2005), decorrente da utilização das técnicas acima descritas, tanto os sistemas baseados em conteúdo quanto os sistemas colaborativos possuem limitações. Estas limitações, motivo da criação dos sistemas híbridos (BALABANOVIĆ; SHOHAM, 1997), possuem características claras de acordo com o tipo de recomendação utilizado, sendo divididas da seguinte maneira:

- **Análise de conteúdo limitada (*limited content analysis*):** presente nas técnicas baseadas em conteúdo, devido as mesmas serem limitadas por uma quantidade específica de características relevantes para a recomendação. Além disso, essas características precisam ser extraídas de forma explícita, o que dificulta muito a extração de atributos através de conteúdo como vídeo, imagem, etc.

- **Problema do novo usuario (*new user problem*):** comum nas técnicas que utilizam as preferências do usuário como métrica, consiste no fato de que um usuário precisa ter um número suficiente de avaliações para que o sistema entenda suas preferências e forneça recomendações precisas.

- **Sobre-especialização (*over-specialization*):** comum nas técnicas de recomendação baseada em conteúdo, consiste no fato de que se o sistema apenas recomenda ao usuário itens semelhantes aos que ele já avaliou de forma positiva, o usuário será limitado à apenas recomendações de itens já avaliados, reduzindo cada vez mais a recomendação de novos itens.

- **Problema do novo item (*new item problem*):** sistemas colaborativos baseiam-se apenas nas preferências dos usuários para fazer as recomendações. Sendo assim, novos itens que ainda não foram avaliados por nenhum usuário não serão recomendados.

- **Esparsidade (*Sparsity*):** quando um item é raramente recomendado devido a sua esparsidade no conjunto de usuários e itens, ou seja, um item que é pouco recomendado pelos usuários tende a ser cada vez menos recomendado em sistemas colaborativos, devido ao pouco número de avaliações que o mesmo possui.

Sendo assim, grande parte das pesquisas relacionadas a sistemas de recomendação tem como objetivo principal melhorar a precisão das técnicas, reduzindo o impacto das limitações descritas. Porém, como apresentado por McNee, Riedl e Konstan (2006), nem sempre os itens mais precisos em relação às métricas de cada método são os mais úteis aos usuários.

Considerando que os sistemas de recomendação usualmente abordam apenas algumas, das muitas métricas que definem a utilidade de um item ao usuário, McNee, Riedl e Konstan (2006) ressaltam que cada vez mais a utilização de sistemas de recomendação leva a construção de um conjunto de itens muito similar. Isso ocorre pois quando um usuário avalia um item, as próximas recomendações levarão o mesmo em consideração, recomendando itens cada vez mais parecidos com o item avaliado. Este processo acaba gerando o que McNee, Riedl e Konstan (2006) definem como "**buraco de similaridade**", onde o sistema tende a fazer apenas recomendações excepcionalmente similares.

3 API: *Application Programming Interface*

Seção sobre API

3.1 O Padrão RESTful

Descrever o padrão RESTful

3.2 Metadados?

Encaixar os metadados aqui

3 API DE RECOMENDAÇÃO ORIENTADA À METADADOS

Através dos benefícios da utilização de mais de um método de recomendação em um sistema em forma de serviço, este trabalho propõe o desenvolvimento de uma API Web[1] que utiliza metadados[2] para fazer a alimentação do conjunto de usuários, itens e avaliações. A combinação destes conjuntos será utilizada para fornecer recomendações às requisições dos clientes.

Partindo da premissa que a API deve atender qualquer modelo de usuário e item, serão fornecidos, em sua inicialização, **metadados** correspondentes a estrutura de ambos, além da estrutura de avaliação. Assim que os metadados sejam fornecidos à API e os dados sejam fornecidos dentro das estruturas definidas, a requisição de recomendações torna-se simples, uma vez que a API já tem conhecimento do usuário e dos itens avaliados por ele. Este método faz com que o processo de recomendação torne-se **transparente** ao cliente.

Como forma de recomendação será utilizado o método híbrido, composto dos métodos **baseado em conteúdo** e **filtragem colaborativa**, escalonando através do método com melhor precisão momentânea. Dessa forma é possível atender qualquer tipo de metadado, fornecendo recomendações independente do número de usuários, itens e avaliações na base de dados.

De posse das estruturas de metadados fornecidas na inicialização, o cliente poderá alimentar o sistema através das interfaces desenvolvidas na API, constituída pelas funções: **novos usuários**, **novos itens** e **novas avaliações**. Estas três ações de alimentação serão responsáveis por preencher os respectivos conjuntos anteriormente descritos e, a partir deles, construir os modelos de cada usuário para o método colaborativo, bem como a matriz de avaliações para o método baseado em conteúdo. Durante a requisição de uma recomendação, a API definirá o método a ser utilizado e o mesmo fará as recomendações com base nos dados conhecidos, respeitando as propriedades descritas nos metadados.

Em um primeiro momento, a API utilizará o método baseado em conteúdo para fornecer as recomendações e predições, uma vez que poucos itens estarão avaliados e o método colaborativo não terá modelos de usuários suficientes. Assim que as métricas de relações entre usuários e quantidade de modelos processados sejam supridas, a API passará a utilizar o método de filtragem colaborativa, caso este esteja gerando recomendações mais precisas.

---

[1] "*Application Programming Interface*- Conjunto de rotinas e padrões de programação para acesso a um aplicativo de software ou plataforma baseado na Web.

[2] Dado sobre o dado ou informação sobre a informação.

Além disso, a API também possuirá a opção do processamento na GPU, agilizando ainda mais o retorno das requisições dos clientes. Este processamento fará uso da tecnologia **CUDA**, utilizando a GPU nos casos em que seja necessário percorrer a matriz de avaliações, além dos casos de processamento dos modelos de usuários.

1 Visão Geral

2 O Analisador de Dados

3 O Motor de Recomendações

3.1 Motor de Conteúdo

3.2 Motor Colaborativo

4 Persistência

5 Interface de Usuário

4 RESULTADOS E DISCUSSÃO

5 CONSIDERAÇÕES FINAIS

Devido ao grande número de implementações dos sistemas de recomendação nas mais diversas áreas que aqui foram apresentadas, torna-se notável a vasta gama de aplicações das técnicas e, mais do que isso, a necessidade de um serviço multi-propósito desprendido do uso de linguagens de programação específicas. Sendo assim, ao final do cronograma, espera-se a obtenção de uma API que satisfaça estes requisitos.

Além disso, é importante ressaltar preocupação na construção de uma documentação direta e coesa ao longo de todo o processo, visando uma futura colaboração externa da comunidade, sendo a API de código aberto. Desta forma, o trabalho pode servir não só como uma alternativa *open-source* aos sistemas de recomendação, mas também como uma tecnologia de utilização simples para futuros estudos na área.

Referências

ADOMAVICIUS, Gediminas; TUZHILIN, Alexander. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. **IEEE transactions on knowledge and data engineering**, IEEE, v. 17, n. 6, p. 734–749, 2005. Citado 8 vezes nas páginas 8, 9, 13, 14, 15, 17, 18 e 22.

_____. Context-aware recommender systems. In: **Recommender systems handbook**. [S.l.]: Springer, 2015. p. 191–226. Citado na página 14.

BALABANOVIĆ, Marko; SHOHAM, Yoav. Fab: content-based, collaborative recommendation. **Communications of the ACM**, ACM, v. 40, n. 3, p. 66–72, 1997. Citado 2 vezes nas páginas 21 e 22.

BENNETT, James; LANNING, Stan *et al.* The netflix prize. In: NEW YORK, NY, USA. **Proceedings of KDD cup and workshop**. [S.l.], 2007. v. 2007, p. 35. Citado na página 9.

BREESE, John S; HECKERMAN, David; KADIE, Carl. Empirical analysis of predictive algorithms for collaborative filtering. In: MORGAN KAUFMANN PUBLISHERS INC. **Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence**. [S.l.], 1998. p. 43–52. Citado na página 18.

BROZOVSKY, Lukas; PETRICEK, Vaclav. Recommender system for online dating service. **arXiv preprint cs/0703042**, 2007. Citado 2 vezes nas páginas 9 e 11.

BURKE, Robin. Hybrid recommender systems: Survey and experiments. **User modeling and user-adapted interaction**, Springer, v. 12, n. 4, p. 331–370, 2002. Citado 3 vezes nas páginas 18, 19 e 20.

_____. Hybrid web recommender systems. **The adaptive web**, Springer, p. 377–408, 2007. Citado na página 18.

CARBONELL, Jaime G; MICHALSKI, Ryszard S; MITCHELL, Tom M. An overview of machine learning. In: **Machine learning**. [S.l.]: Springer, 1983. p. 3–23. Citado na página 12.

CLAYPOOL, Mark *et al.* Combining content-based and collaborative filters in an online newspaper. In: CITESEER. **Proceedings of ACM SIGIR workshop on recommender systems**. [S.l.], 1999. v. 60. Citado 2 vezes nas páginas 21 e 22.

COPPIN, Ben. **Inteligência artificial**. [S.l.]: Grupo Gen-LTC, 2015. Citado 2 vezes nas páginas 8 e 12.

GAVALAS, Damianos *et al.* Mobile recommender systems in tourism. **Journal of Network and Computer Applications**, Elsevier, v. 39, p. 319–333, 2014. Citado na página 9.

GOMEZ-URIBE, Carlos A; HUNT, Neil. The netflix recommender system: Algorithms, business value, and innovation. **ACM Transactions on Management Information Systems (TMIS)**, ACM, v. 6, n. 4, p. 13, 2016. Citado na página 13.

GUO, Guibing *et al.* Librec: A java library for recommender systems. In: **UMAP Workshops**. [S.l.: s.n.], 2015. Citado na página 11.

HILL, Will *et al.* Recommending and evaluating choices in a virtual community of use. In: ACM PRESS/ADDISON-WESLEY PUBLISHING CO. **Proceedings of the SIGCHI conference on Human factors in computing systems**. [S.l.], 1995. p. 194–201. Citado na página 8.

HUANG, Zan *et al.* A graph-based recommender system for digital library. In: ACM. **Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries**. [S.l.], 2002. p. 65–73. Citado na página 9.

LINDEN, Greg; SMITH, Brent; YORK, Jeremy. Amazon. com recommendations: Item-to-item collaborative filtering. **IEEE Internet computing**, IEEE, v. 7, n. 1, p. 76–80, 2003. Citado na página 9.

LOPS, Pasquale; GEMMIS, Marco De; SEMERARO, Giovanni. Content-based recommender systems: State of the art and trends. In: **Recommender systems handbook**. [S.l.]: Springer, 2011. p. 73–105. Citado 2 vezes nas páginas 15 e 16.

MCNEE, Sean M; RIEDL, John; KONSTAN, Joseph A. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: ACM. **CHI'06 extended abstracts on Human factors in computing systems**. [S.l.], 2006. p. 1097–1101. Citado na página 23.

MLADENIC, Dunja. Text-learning and related intelligent agents: a survey. **IEEE intelligent systems and their applications**, IEEE, v. 14, n. 4, p. 44–54, 1999. Citado na página 15.

NASCIMENTO, Vinicius Dalto do. **FILTRAGEM COLABORATIVA COMO SERVIÇO UTILIZANDO PROCESSAMENTO NA GPU**. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2013. Citado na página 11.

PANG, Bo; LEE, Lillian; VAITHYANATHAN, Shivakumar. Thumbs up?: sentiment classification using machine learning techniques. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10**. [S.l.], 2002. p. 79–86. Citado na página 13.

PAZZANI, Michael; BILLSUS, Daniel. Content-based recommendation systems. **The adaptive web**, Springer, p. 325–341, 2007. Citado na página 16.

RESNICK, Paul; VARIAN, Hal R. Recommender systems. **Communications of the ACM**, ACM, v. 40, n. 3, p. 56–58, 1997. Citado na página 13.

RICCI, Francesco; ROKACH, Lior; SHAPIRA, Bracha. **Introduction to recommender systems handbook**. [S.l.]: Springer, 2011. Citado 3 vezes nas páginas 9, 13 e 14.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência artificial**. [S.l.]: Elsevier, 2004. Citado 2 vezes nas páginas 8 e 12.

SARWAR, Badrul *et al.* Item-based collaborative filtering recommendation algorithms. In: ACM. **Proceedings of the 10th international conference on World Wide Web**. [S.l.], 2001. p. 285–295. Citado na página 17.

SCHAFER, JHJB *et al.* Collaborative filtering recommender systems. **The adaptive web**, Springer, p. 291–324, 2007. Citado 2 vezes nas páginas 17 e 18.

SCHAFER, J Ben; KONSTAN, Joseph; RIEDL, John. Recommender systems in e-commerce. In: ACM. **Proceedings of the 1st ACM conference on Electronic commerce**. [S.l.], 1999. p. 158–166. Citado 2 vezes nas páginas 13 e 14.

SCHAFER, J Ben; KONSTAN, Joseph A; RIEDL, John. E-commerce recommendation applications. In: **Applications of Data Mining to Electronic Commerce**. [S.l.]: Springer, 2001. p. 115–153. Citado na página 9.

SEBASTIANI, Fabrizio. Machine learning in automated text categorization. **ACM computing surveys (CSUR)**, ACM, v. 34, n. 1, p. 1–47, 2002. Citado na página 13.

TRAN, Thomas; COHEN, Robin. Hybrid recommender systems for electronic commerce. In: **Proc. Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, Technical Report WS-00-04, AAAI Press**. [S.l.: s.n.], 2000. Citado 2 vezes nas páginas 20 e 21.