UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

RICARDO GODOY DE OLIVEIRA

SISTEMA DE RECOMENDAÇÃO PARA AUMENTAR O FATOR DE ESCALA NO ATENDIMENTO AOS ALUNOS EM CURSOS A DISTÂNCIA

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Leandro Krug Wives

Porto Alegre
2018

Agradeço à Kátia por todo o apoio e
companheirismo ao longo dos anos,
principalmente nesta reta final.
Todo meu amor a ti e a nossa família.

RESUMO

Com a popularização dos cursos online na modalidade a distância (por vezes chamados de MOOCs – *Massive Open Online Courses* –, quando abertos ao público, sem limite) nos últimos anos, quem antes não tinha acesso a certas áreas do conhecimento agora o tem de maneira facilitada. Contudo, para os instrutores, torna-se cada vez mais difícil atender individualmente cada aluno, pois esses, em diversos casos, somam um número elevado, chegando aos milhares. Nesse contexto, são necessárias ferramentas de auxílio aos instrutores e tutores, de tal modo que consigam ter uma maior vazão no suporte aos estudantes. O presente trabalho descreve um sistema de recomendação que, por meio de técnicas de agrupamento de mensagens, poderá ser utilizada na tentativa de minimizar interações repetitivas entre educadores e educandos.

Palavras-chave: Sistemas de Recomendação. ScolarTIC. Educação a Distância.

Recommender system to increase the scale factor in assistence to students in distance learning

ABSTRACT

With the popularization of distance learning through online courses (sometimes called MOOCs – Massive Open Online Courses –, when they are open to the public without limitation) in the last several years, access to certain knowledge domains, not easy attainable before, are now easily accessible. However, for the instructors, it has become harder than ever to answer students individually, for there are often thousands of them. In this context, it becomes a necessity to have tools capable of aiding in this matter, such that they can support more students. The present paper describes a tool that could, through the analyses similarity between messages, be used in an attempt to minimize repetitive interactions among teachers and students.

Keywords: Recommender System. ScolarTIC. Distance Learning.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

EAD	Educação a Distância
SR	Sistema(s) de Recomendação
FBC	Filtragem Baseada em Conteúdo
FC	Filtragem Colaborativa
AVA	Ambiente Virtual de Aprendizagem
SGBD	Sistema Gerenciador de Banco de Dados
API	Application Programming Interface
REST	Representational State Transfer
JSON	JavaScript Object Notation
RSLP	Removedor de Sufixos para a Língua Portuguesa

SUMÁRIO

1 INTRODUÇÃO

A educação a distância (EAD) existe desde o século XXI, fazendo uso do correio como forma de transmissão de informações. Serve de alternativa ao ensino tradicional. Mais recentemente, com a popularização da internet e dos computadores pessoais, criaram-se novas possibilidades para o ensino e a aprendizagem. Conforme colocam Oliveira, Carneiro e Munhoz (2009),

> Os ambientes virtuais de aprendizagem permitem a alunos e professores pensarem, refletirem, analisarem as informações recolhidas nas revisões bibliográficas, nas listas de discussão, nos bancos de dados. Permitem, ainda, relacionarem esse novo conhecimento ao seu conhecimento anterior, às outras informações disponíveis e construírem novos conhecimentos; produzirem novos documentos, disponibilizando-os no ciberespaço e/ou nos espaços tradicionais para que venham alimentar uma inteligência coletiva que, por sua vez, propiciará novos acessos, novos pensares, novas construções, novas produções e comunicações em um verdadeiro círculo construtivo e emancipador, uma inteligência coletiva dentro de sua própria cultura que manterá conexões com outras culturas, sem, no entanto, perder a sua característica territorial (OLIVEIRA; CARNEIRO; MUNHOZ, 2009, p. 4).

Nos últimos anos, o aumento na oferta e procura de cursos a distância foi impressionante. Embora em 2017 o número de novos estudantes não tenha sido tão acentuado como nos anteriores, a adoção ainda foi considerável: entre as maiores plataformas internacionais, aproximadamente 23 milhões de pessoas se inscreveram pela primeira vez, compondo parte do total de 81 milhões. Enquanto isso, o aumento na oferta de novos cursos se manteve, totalizando 9.400 em 2017, contra os 6.850 existentes no ano anterior (CLASS CENTRAL, 2018).

Nesse contexto, pensando no papel dos professores, fica evidente que é inviável o atendimento individual a todos os alunos, visto que isso demandaria tempo e esforço muito além do possível para poucos tutores. Sendo assim, estes carecem de mecanismos para aumentar a escala de apoio aos estudantes, ou, melhor ainda, evitar que seja necessária a interação direta entre aprendiz e mestre sempre que possível.

1.1 Objetivos

Com o presente trabalho, se busca um possível sistema de recomendação para auxiliar no atendimento aos alunos, de forma que a necessidade de contato direto entre estudante e professor seja minimizada. O resultado esperado é aumentar a autonomia do aprendiz e diminuir a carga de trabalho dos tutores, bem como o número de mensagens

repetidas.

Pontualmente, espera-se:

a) estudar e compreender os diferentes tipos de sistemas de recomendação, a fim de encontrar as técnicas mais adequadas para cada situação;
b) entender a estrutura do banco de dados da plataforma ScolarTIC para realizar o processamento das informações provenientes dele;
c) desenvolver uma ferramenta capaz de gerar recomendações a partir de mensagens enviadas aos fóruns, retornando outras semelhantes;
d) avaliar as recomendações feitas pelo sistema em um ambiente de testes;
e) discutir as possibilidades de integração da ferramenta desenvolvida à plataforma.

1.2 Contribuição

Como resultado deste trabalho, pretende-se disponibilizar às entidades responsáveis por administrar as informações da plataforma ScolarTIC uma ferramenta capaz de reduzir a duplicação de mensagens (principalmente dúvidas que os tutores precisam verificar), e aumentar o engajamento dos usuários através de recomendações de mensagens passadas relacionadas. Hoje, a plataforma não possui nenhuma funcionalidade parecida, logo, acredita-se que será uma adição válida para todos os envolvidos com os cursos.

1.3 Estrutura do texto

Após esta introdução, no capítulo 2, serão estudados os sistemas de recomendação, suas diferentes classes, vantagens e desvantagens, bem como alguns exemplos reais que fazem uso destes. O capítulo 3.2 apresentará brevemente a plataforma ScorlatTIC e seus tipos de curso. No capítulo 3, será detalhada a ferramenta desenvolvida para este trabalho, que tenta resolver alguns problemas na vazão do atendimento de alunos por parte dos professoras da plataforma. No capítulo 4, apresentar-se-ão os resultados obtidos após os testes com a ferramenta em um ambiente de testes. Finalmente, o capítulo 5 conterá as considerações finais, incluindo sugestões para utilizar a ferramenta criada em um cenário real.

2 SISTEMAS DE RECOMENDAÇÃO

Em geral, os avanços tecnológicos nos permitem criar e armazenar um volume de informações muito além do que conseguiríamos processar sem o auxílio de ferramentas automatizadas (SARWAR et al., 2001). Esse fenômeno é chamado de Sobrecarga de Informação – ou *Information Overload* –, e a principal maneira de lidar com ele é através do uso de filtros (preferencialmente automáticos), o que diminui o esforço requerido para extrair-se objetos de interesse, tal qual conhecimento útil, de uma grande quantidade de dados (BEZERRA, 2002).

Neste capítulo, portanto, serão apresentados os sistemas de recomendação (SR), instrumentos capazes de, levando em consideração características do usuário, selecionar os dados mais relevantes (BEZERRA, 2002). Também serão abordadas as técnicas tipicamente empregadas para esta finalidade.

2.1 Visão geral

É comum que pessoas troquem recomendações entre si no dia a dia – seja de filmes, seriados, livros, receitas, etc. Para muitos, essa é uma das principais influências para tomadas de decisão. Os sistemas de recomendação, então, tentam auxiliar durante tais escolhas (RESNICK; VARIAN, 1997).

2.2 Origem

Nos anos 1990 surgiu o termo Filtragem Colaborativa, cunhado pelos criadores do primeiro SR, *Tapestry*. Este era um sistema experimental de e-mail, desenvolvido no Xerox Palo Alto Research Center, cuja motivação foi o crescente volume de mensagens eletrônicas, que, por vezes, inundava as caixas de entrada dos usuários com documentos (DENNING; PALME; TERRY, apud GOLDBERG et al., 1992, p. 61).

Hoje, no entanto, prefere-se o termo mais genérico "sistemas de recomendação" por dois motivos principais. "First, recommenders may not explictly collaborate with recipients, who may be unknown to each other. Second, recommendations may suggest particularly interesting items, in addition to indicating those that should be filtered out" (RESNICK; VARIAN, 1997, p. 56).

2.3 Classes de sistemas de recomendação

Existem abordagens variadas possíveis ao se construir um SR, as quais se diferenciam pela maneira como as recomendações são feitas, utilizando-se, em diversas situações, uma combinação de diferentes técnicas. Entre elas, destacam-se (ELAHI; RICCI; RUBENS, 2016, p. 317):

a) estereotipagem;
b) filtragem baseada em conteúdo;
c) filtragem colaborativa;
d) coocorrência;
e) baseada em grafos;
f) relevância global;
g) híbrida.

2.3.1 Estereotipagem

Uma das formas mais antigas de modelagem de usuários e recomendação, a estereotipagem – ou *stereotyping* –, foi empregada no sistema *Grundy*, um recomendador de romances (RICH, 1979 apud ELAHI; RICCI; RUBENS, 2016). Sua autora se inspirou nos esteriótipos da psicologia para julgar pessoas a partir de poucas características. Rich (1979) define os estereótipos – ou "facetas" – como um grupo de características, e então as recomendações são manualmente associadas a estes.

O problema com esta abordagem é justamente que os usuários são rotulados com base em poucas informações, portanto, gerando muitas recomendações inadequadas. Além disso, o mapeamento manual entre as facetas e os conteúdos é muito trabalhoso, o que acaba limitando a quantidade viável de itens no sistema (BARLA, 2011 apud ELAHI; RICCI; RUBENS, 2016).

2.3.2 Filtragem baseada em conteúdo

Filtragem baseada em conteúdo (FBC) – ou *content-based filtering* – é uma das classes de recomendação mais utilizadas e pesquisadas (BARLA, 2011 apud ELAHI;

RICCI; RUBENS, 2016). Dos 62 artigos avaliados por Elahi, Ricci e Rubens (2016), 34 (55%) empregam esta técnica.

Tipicamente, para a construção dos modelos, as características extraídas dos documentos são baseadas no próprio texto, como palavras, frases ou n-gramas. Contudo, também é possível levar em consideração elementos não textuais, como estilo, diagramação e marcações. Todas essas informações recebem um peso, e é comum que apenas as mais importantes sejam incluídas nos cálculos. Então, para se produzir recomendações, se compara o modelo do usuário (ou da base de documentos) com os dos itens candidatos. Para as comparações, pode-se usar, por exemplo, uma representação no espaço vetorial (ou *Vector Space Model*, VSM), onde cada componente é um peso referente a uma característica – tipicamente obtido por TF-IDF (*term frequency–inverse document frequency*) – e o coeficiente de similaridade dos cossenos (ELAHI; RICCI; RUBENS, 2016, p. 318-319).

A FBC possui diversas vantagens sobre a estereotipagem, como recomendações mais individualizadas entre os usuários. Ainda, evita a necessidade da classificação prévia manual dos documentos, já que os modelos podem ser construídos de forma automatizada.

A estereotipagem é superior à FBC no que diz respeito ao desempenho, pois, nesta última, é necessário que se analise as características, construa-se o modelo e então calcule-se as semelhanças. Conforme o número de itens e usuários cresce, toda esta computação começa a demandar recursos significativos.

Ademais, outra desvantagem da FBC é o fato de que os modelos podem ser demasiadamente específicos. Isso ocorre porque as recomendações são feitas puramente sobre a semelhança entre os conteúdos dos itens, o que, dependendo do objetivo, pode não ser desejado, já que o usuário talvez não esteja interessado em ver algo tão semelhante ao que já conhece. Também é possível que, para dois documentos diferentes, o sistema os considere igualmente relevantes, mesmo que um tenha sido produzido por uma autoridade na área, e o outro apenas parafraseie o primeiro.

2.3.3 Filtragem colaborativa

O termo filtragem colaborativa (FC) foi cunhado por Goldberg et al. (1992), propondo que "information filtering can be more effective when humans are involved in the filtering process". A intuição por trás disso, segundo Resnick e Varian (1997), é que usuários que já gostaram de coisas parecidas no passado também irão gostar de coisas

similares no futuro. A comparação acaba sendo feita entre os usuários, não diretamente sobre os documentos (BALAVANOVIC; SHOHAM, 1997).

De acordo com Torres Júnior (2004), as principais vantagens da FC são:

a) como quem realiza a filtragem são os próprios usuários, a análise acaba levando em consideração aspectos que vão além das palavras contidas em um texto, portanto, também distinguindo fatores como, por exemplo, as qualidades de conteúdo e de escrita (ou apresentação em geral), o que seria muito difícil de se alcançar através de um método automatizado;
b) é possível considerarmos basicamente qualquer tipo de item, não apenas textuais, como músicas, filmes, seriados, receitas, lugares, etc.;
c) em decorrência de realizar-se a comparação entre usuários em vez de documentos, poderão ser recomendados itens inesperados que não necessariamente tenham conteúdo semelhante aos documentos anteriormente vistos.

Quanto às desvantagens, Mooney (2010) e Adomavicius e Tuzhilin (2005) citam:

a) usuários com preferências peculiares provavelmente não receberão boas recomendações, pois poucos usuários compartilham de seus interesses;
b) é necessário haver informações sobre avaliações passadas dos usuários para que se possa realizar novas recomendações;
c) quanto menor o número de usuários, mais difícil se torna a tarefa de encontrar usuários semelhantes;
d) itens novos acabam não sendo recomendados por um certo tempo, já que, para isto ocorrer, é necessário que uma quantidade considerável de usuários avaliem estes positivamente.

2.3.4 Coocorrência

Filtragem por coocorrência, como o nome indica, considera elementos para recomendação conforme eles aparecem juntos em uma mesma fonte (ELAHI; RICCI; RUBENS, 2016, p. 320). Uma das utilizações mais comuns deste tipo de técnica pode ser vista em diversas lojas online quando, ao acessarmos qualquer página de produto, nos é apresentada uma seção "Frequentemente comprados juntos" e/ou outra "Quem viu este produto também viu", exemplos apresentados nas figuras 1 e 2.

Figura 1: Exemplo de recomendação de itens comumente comprados juntos

Fonte: página de produto na loja virtual da Amazon brasileira (<https://amazon.com.br>). Acesso em: 5 jun. 2018.

2.3.5 Filtragem baseada em grafos

Na filtragem baseada em grafos, se considera as conexões que podem ser feitas entre os itens através de suas características. Por exemplo, entre artigos científicos, podemos analisar: citações, autores, onde foi publicado, ano de publicação, entre outras informações. A partir da rede construída, pode-se então realizar caminhadas aleatórias com reentradas para se determinar os itens mais populares (ELAHI; RICCI; RUBENS, 2016, p. 320-321).

2.3.6 Relevância global

Em uma filtragem que usa relevância global, emprega-se apenas uma medida para todos os itens, não sendo consideradas características do usuário. Ou seja, é comum encontrarmos sua utilização em conjunto a outras técnicas, ajudando a decidir a ordem em que as recomendações serão apresentadas (ELAHI; RICCI; RUBENS, 2016, p. 321). Um exemplo seria considerar a popularidade geral de um item, o que pode ser aplicado ao caso de uma locadora de filmes: são recomendados os filmes com as melhores avaliações dos clientes; aqui, fica implícito que é esperado que uma pessoa irá gostar do que a maioria das outras gosta.

Figura 2: Exemplo de recomendação de itens visualizados em mesma sessão

Fonte: página de produto na loja virtual do Submarino (<https://submarino.com.br>). Acesso em: 5 jun. 2018.

2.3.7 Abordagens híbridas

As várias abordagens anteriores podem ser empregadas em conjunto para se chegar a um sistema híbrido. Como comentado anteriormente, é comum o uso de relevância global de alguma característica para ordenar, por exemplo, os resultados de uma FBC. Ainda, fazendo uso do método de grafos, é possível expandir ou restringir as recomendações conforme as relações entre os itens (ELAHI; RICCI; RUBENS, 2016, p. 321).

2.4 Exemplos de sistema de recomendação

Abaixo, são apresentados alguns exemplos de SR, muito comuns em plataformas digitais.

2.4.1 Amazon

A Amazon[1] é uma loja de comércio eletrônico presente em diversos países (WIKIPEDIA, 2018a). Para auxiliar e incentivar os clientes a comprar, utiliza SR em algumas partes de sua plataforma.

[1]Disponível em: <https://www.amazon.com/>.

Conforme já dito, na página de qualquer produto da loja, são exibidos itens que foram comprados em conjunto com o item consultado (vide figura 1). Além disso, também são apresentados outros itens adquiridos por compradores do produto em destaque, como pode ser visto na figura 3, o que ilustra o uso de coocorrência.

Figura 3: Produtos relacionados na loja da Amazon

Fonte: página de produto na Amazon (<https://www.amazon.com.br>). Acesso em: 5 jun. 2018.

2.4.2 Google

Se considerarmos o funcionamento do site de pesquisa Google[2], ele faz uso de diferentes técnicas de recomendação (MACMANUS, 2009):

a) baseando-se no histórico de pesquisas e local, por exemplo, ele personaliza os resultados da busca;

b) se o usuário possui uma conta do Google, é possível que os resultados também sejam influenciados pelo histórico de navegação deste indivíduo;

c) o próprio algoritmo de ranqueamento utilizado pela empresa é baseado em recomendações de outros usuários, pois um dos principais fatores considerados é a quantidade de links que apontam para uma determinada página;

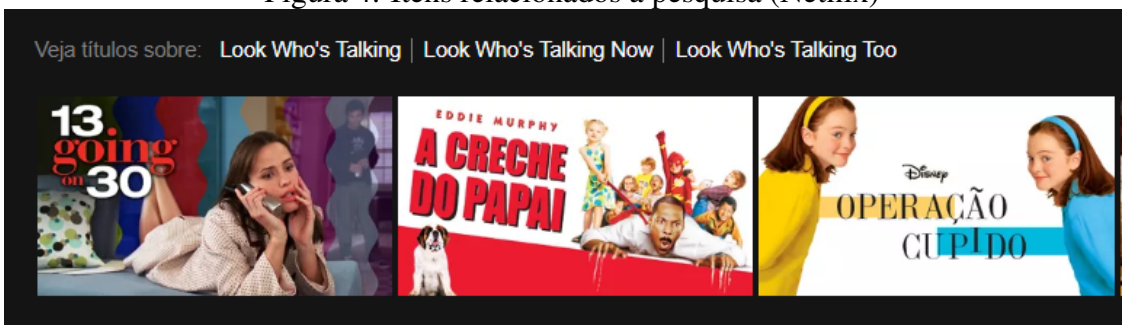
d) também pode-se considerar que a funcionalidade "Você quis dizer. . . " é uma forma de recomendação.

[2]Disponível em: <https://google.com>.

2.4.3 Netflix

No serviço de seriados e filmes sob demanda Netflix[3], também podemos observar o uso de SR. Ao se pesquisar por um filme que não está disponível em seu catálogo, por exemplo, o site apresenta filmes relacionados, como pode ser visto na figura 4 (após pesquisar-se por *Look who's talking*).

Figura 4: Itens relacionados à pesquisa (Netflix)



Fonte: página de pesquisa do Netflix (<https://www.netflix.com/search?q=look%20who%27s%20talking>). Acesso em: 5 abr. 2018.

2.5 Recomendações em ambientes do tipo fórum eletrônico

Serão detalhadas, a seguir, duas aplicações de SR em ambientes do tipo fórum eletrônico.

2.5.1 Recomendação baseada em categorias

Fischer (2005) desenvolveu um SR utilizando FBC para o sistema Virtuale – ambiente EAD do Centro Universitário Feevale. Em seu trabalho, ele comenta sobre alguns problemas presentes neste tipo de ambiente, e, especialmente relacionado ao presente trabalho, o seguinte:

> Freqüentemente [...] se pode observar o fato de um usuário enviar uma pergunta (criação de novo tópico) sem antes verificar se tal assunto já não foi previamente abordado no fórum, o que eliminaria a necessidade de repetir a pergunta. Este fato obriga os usuários do fórum a responder novamente à pergunta ou observar que este assunto já foi discutido, quando não ocorre de simplesmente ignorarem a pergunta, o que causa uma má impressão entre os usuários (FISCHER, 2005, p. 19).

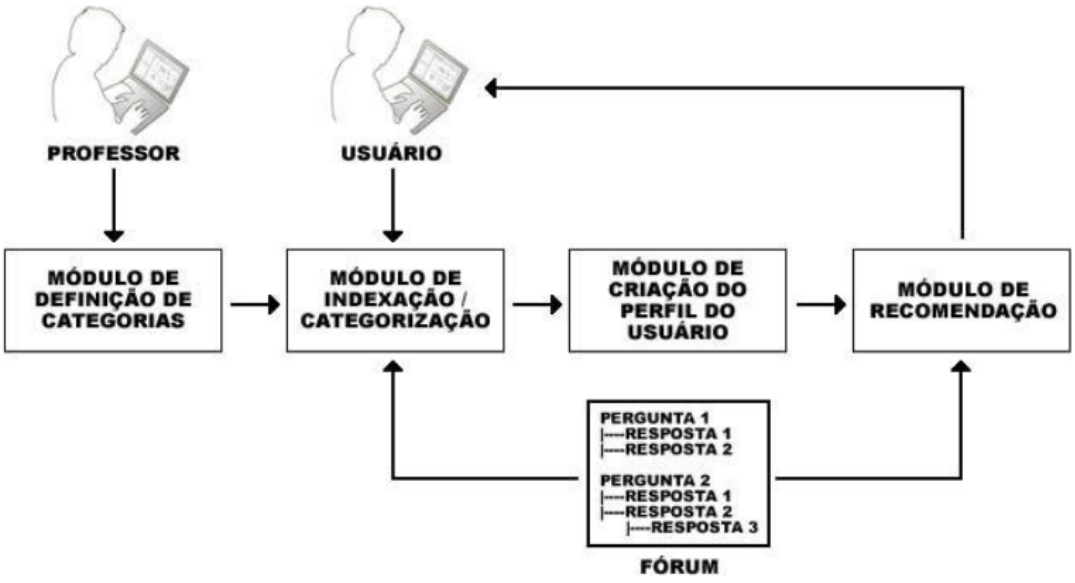
[3]Disponível em: <https://netflix.com>.

No que diz respeito ao ScorlarTIC, isso é particularmente problemático, pois uma pergunta de um aluno sendo ignorada poderá ser interpretada como falta de atenção dos tutores, potencialmente gerando reclamações através do sistema de suporte. Ainda, a duplicação de questões já abordadas desperdiça o tempo dos responsáveis, pois estes já investiram tempo no passado para sanar aquele ponto específico, enquanto poderiam estar produzindo novos materiais e/ou aulas.

O sistema proposto por Fischer (2005) é composto por quatro módulos (sua organização pode ser vista na figura 5):

a) definição de categorias;
b) indexação e categorização;
c) manutenção do perfil do usuário;
d) recomendação.

Figura 5: Arquitetura geral e interação entre os módulos do sistema proposto por Fischer (2005)



.

Fonte: Fischer (2005, p. 48).

As etapas de operação do sistema podem ser resumidas da seguinte forma (FISCHER, 2005 apud ZAMBERLAN, 2011):

a) o professor enumera as categorias que poderão ser utilizadas no fórum;
b) ao enviar uma mensagem, esta é categorizada e indexada de acordo, mas ainda não é exibida no fórum;
c) as categorias aplicadas à mensagem são adicionadas ao perfil do usuário;

d) são apresentadas recomendações de mensagens das mesmas categorias contidas no perfil do usuário, que podem ser avaliadas (se foram úteis ou não);
e) se a mensagem não puder ser encaixada em nenhuma categoria existente ou, ainda, se não existirem mensagens com as categorias desta, a mensagem se torna visível no fórum.

O sistema apresentado no presente trabalho é consideravelmente semelhante ao descrito acima, porém, com algumas diferenças-chave:

Além da distinção óbvia de ambiente virtual (Virtuale $\times$ ScolarTIC), Fischer (2005) faz uso de categorias para classificar mensagens, o que não acontece no sistema proposto aqui, na qual é considerado apenas o conteúdo. Igualmente, não serão mantidos perfis de usuários. Também a estrutura dos ambientes é relevante, pois no ScorlarTIC não é permitido que os alunos criem tópicos, o que serve para limitar os assuntos das discussões. Dito isso, os desenvolvimentos deste trabalho não partem de um sistema existente, apenas aproveita parte das bases teóricas já pesquisadas.

2.5.2 Recomendações no Moodle

Zamberlan (2011) criou uma extensão (ou *plug-in*) para o ambiente virtual de aprendizagem (AVA) Moodle, que estende as funcionalidades do fórum padrão, passando a contar com gerenciamento de categorias, recomendação de mensagens e visualização de relatórios.

Para encontrar as recomendações, a autora também utilizou FBC, o que ocorre no momento em que um usuário tenta enviar uma mensagem. Neste instante, são consideradas apenas outras mensagens da mesma categoria, e então elas são comparadas para identificar as mais semelhantes. Se o usuário recusa as sugestões oferecidas, a nova mensagem é então adicionada ao fórum. Ainda, as ações dos usuários são armazenadas para que, posteriormente, possam ser analisadas através de relatórios.

Semelhante ao trabalho de Fischer (2005), aqui as categorias também são gerenciadas pelo professor. Contudo, não implementa um perfil do usuário, e por isso considera apenas as categorias da mensagem sendo inserida.

3 IMPLEMENTAÇÃO DA FERRAMENTA

No presente capítulo, será detalhado o desenvolvimento do sistema proposto: o SR a ser utilizado como ferramenta de apoio aos tutores e alunos do ScolarTIC.

3.1 Visão geral

A plataforma ScolarTIC é desenvolvida por uma empresa espanhola, parceira da Fundação Telefônica[4]. Portanto, não há possibilidade de integrar a ferramenta proposta diretamente como uma funcionalidade dos fóruns dos cursos. Contudo, a empresa Hard-Fun Studios recebe diariamente, em forma de arquivos CSV[5] compactados, uma cópia dos dados referentes aos cursos do programa Escolas Conectadas.

Sendo assim, pelas limitações descritas acima, o sistema de recomendação a ser desenvolvido irá complementar a plataforma externamente. Será um sistema à parte que, consumindo os *backups* diários, irá – por exemplo – disparar as ações necessárias ao final de cada dia.

Por conseguinte, não será possível evitar que mensagens duplicadas sejam de fato enviadas aos fóruns, como acontece nos trabalhos de Zamberlan (2011) e Fischer (2005). O SR desenvolvido, neste momento, irá apenas calcular as recomendações, e é esperado que em um futuro breve isso seja utilizado para, por exemplo, disparar e-mails a quem escreveu nos fóruns com listas de mensagens relacionadas (seja na tentativa de responder dúvidas ou instigar novas discussões).

3.2 Plataforma ScolarTIC

Neste trabalho, os dados utilizados foram extraídos da plataforma ScolarTIC[6], desenvolvida pela Telefónica Educación Digital[7]. Os conteúdos oferecidos para o Brasil fazem parte do programa Escolas Conectadas[8], e são produzidos e promovidos pela

[4]Aproveito este espaço para agradecer à Fundação Telefônica por concordar em disponibilizar os dados para este trabalho. Também esclareço que qualquer dado que porventura seja exibido neste estudo estará anonimizado, evitando o vazamento de informações a respeito dos usuários.

[5]CSV, ou *comma-separated values*, é um arquivo de texto onde temos um registro por linha, e os campos são tipicamente separados por vírgula (WIKIPEDIA, 2018c).

[6]Disponível em: <https://www.scolartic.com/pt>.

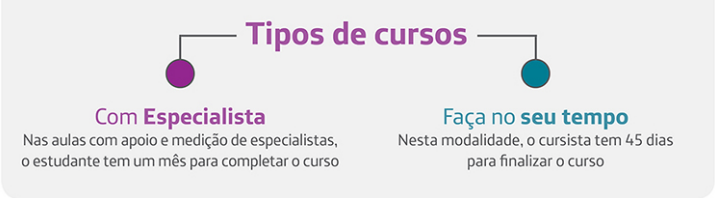
[7]Disponível em: <https://www.telefonicaeducaciondigital.com>.

[8]Disponível em: <http://fundacaotelefonica.org.br/projetos/escolas-conectadas/>.

Fundação Telefônica/Vivo[9], em parceria com outras empresas e organizações, como o Instituto Tear de Inovações[10] e a HardFun Studios[11,12]. Todo este material se destina à formação continuada de educadores, principalmente dos ensinos fundamental e médio.

No ScolarTIC, são oferecidos cursos em diferentes modalidades, conforme figura 6, entre elas: com especialista, na qual há o apoio e acompanhamento de um ou mais tutores; e "faça no seu tempo", em que os cursos não têm datas de começo, podendo ser iniciados a qualquer momento, e não há limite de alunos.

Figura 6: Tipos de cursos oferecidos pelo programa Escolas Conectadas



Fonte: Fundação Telefônica – Escolas Conectadas. Acesso em: 3 abr. 2018.

Ademais, semelhante a muitas plataformas de educação a distância (EAD), a supracitada oferece, como meio de interação entre os participantes, um fórum de discussões. Neste, os tópicos são criados exclusivamente pelos tutores. É comum, portanto, termos tópicos para entregas de atividades, para sanar dúvidas e para envio de sugestões e críticas. O conteúdo destas discussões compõe o conjunto de dados utilizados para o desenvolvimento da ferramenta apresentada neste trabalho.

3.3 Fóruns e seus tópicos

Cada curso pode possuir diversos tópicos em seu fórum. Estes são criados exclusivamente pelos tutores responsáveis (para entrega de atividades ou solucionar dúvidas); tipicamente, correspondem aos módulos dos cursos, acrescidos de um ou dois tópicos para que os alunos possam ter conversas mais livres ou deixar os comentários desejados.

Dentre os arquivos recebidos, e para fins de detalhamento do que foi desenvolvido, basta analisarmos o arquivo `MbMessage.csv`. Este contém todos os dados e metadados de cada mensagem na plataforma. As colunas com dados relevantes são:

[9]Disponível em: <http://fundacaotelefonica.org.br>.
[10]Disponível em: <http://www.tearinovacoes.org.br>.
[11]Disponível em: <http://hardfunstudios.com>.
[12]Fui membro da HardFun Studios recentemente, e parte de meu tempo lá foi dedicado ao desenvolvimento da ferramenta apresentada neste trabalho. Deixo aqui meus agradecimentos aos sócios da empresa (e grandes amigos), Juliano Bittencourt (egresso deste instituto) e Robson Mendonça.

a) `messageId`, é o identificador único da mensagem no banco de dados original;
b) `userId`, é o identificador único do autor da mensagem no banco de dados original;
c) `subject`, é o texto referente ao assunto da mensagem;
d) `body`, é o corpo da mensagem propriamente dito.

Ainda, os seguintes metadados também são considerados importantes:

a) `createDate`, é a data em que a mensagem foi criada;
b) `modifiedDate`, é a data em que a mensagem foi alterada pela última vez;
c) `parentMessageId`, é o identificador da mensagem pai, no caso de uma resposta;
d) `rootMessageId`, é o identificador da mensagem raiz, ou seja, a mensagem de criação do tópico.

3.4 Arquitetura do sistema

O protótipo desenvolvido é composto, pode-se dizer, de quatro componentes principais: banco de dados, importador de dados, serviço de recomendações, e interface web. Cada um será detalhado a seguir.

3.4.1 Banco de dados

Foi utilizado um banco de dados PostgreSQL[13], que é relacional, mas também possui diversas funcionalidades não relacionais. A escolha deste sistema gerenciador de banco de dados (SGBD) foi tomada puramente com base na experiência do autor deste trabalho, que já possui familiaridade com a ferramenta.

As tabelas criadas para dar suporte ao sistema de recomendação foram: `messages` e `tdidf_pickles`. Ambas serão descritas nas subseções a seguir.

3.4.1.1 Tabela de mensagens

A tabela `messages`, como o nome indica, irá conter os dados referentes a todas as mensagens importadas, e, portanto, é considerada a mais importante. Sua estrutura é a seguinte (alguns campos foram omitidos por não serem relevantes ao SR, servindo apenas

[13]Disponível em: <https://www.postgresql.org>

para controle do próprio SGBD ou de outras ferramentas):

a) `original_id`, corresponde ao identificador original da mensagem no banco de dados externo;
b) `original_author_id`, corresponde ao identificador do autor da mensagem no banco de dados externo;
c) `original_subject`, corresponde ao assunto original da mensagem no banco de dados externo;
d) `original_body`, corresponde ao corpo da mensagem original no banco de dados externo;
e) `original_root_id`, corresponde ao identificador da mensagem raiz (de criação do tópico) no banco de dados externo;
f) `body_tokens`, contém os *tokens* obtidos a partir do corpo da mensagem, e será explicado em detalhes a seguir.

3.4.1.2 Tabela de cache do processamento

Também foi criada a tabela `tfidf_pickles`. Esta serve puramente para fins de otimização da ferramenta: no momento em que se faz necessário o cálculo da matriz TF-IDF, em vez de sempre calculá-la, primeiro se pesquisa nesta tabela se já não existe uma cópia da matriz, o que acelera bastante o processo dependendo da quantidade de mensagens na coleção considerada (todos os conceitos comentados aqui serão detalhados na seção 3).

Possui apenas três campos:

a) `root_id`, corresponde ao identificador do tópico considerado nas recomendações (no caso de a pesquisa ser restrita a um tópico; pode ser nulo, indicando que a pesquisa é global);
b) `corpus_size_on_last_update`, corresponde à quantidade de mensagens existentes no momento da recomendação, e serve para indicar se é preciso atualizar o modelo ou não;
c) `pickle`, corresponde à representação binária do objeto Python que representa o modelo utilizado para os cálculos de similaridade.

3.4.2 Importação e pré-processamento das informações

Com a estrutura do arquivo descrita anteriormente, é possível processá-lo e importá-lo para o banco de dados. Durante esta importação, o sistema também já irá realizar algumas etapas do pré-processamento necessário para chegarmos às recomendações.

Este módulo é, na verdade, acoplado ao serviço de recomendações, pois eles compartilham algumas características: ambos foram escritos na linguagem de programação *Python*[14] e utilizam a biblioteca NLTK (*Natural Language Toolkit*) para auxiliar nas tarefas que dizem respeito a processamento textual. Inclusive, este foi um dos motivos da escolha da linguagem: existência de bibliotecas de fácil utilização para tarefas básica de manipulação de textos.

A primeira etapa é importar o arquivo `MbMessage.csv`. Para isso, foi escrito um utilitário de linha comando, com auxílio da biblioteca *Invoke*[15].

Antes da importação propriamente dita, precisou-se eliminar caracteres inválidos do arquivo (mais precisamente o caractere nulo, que por razões desconhecidas estava presente em alguns pontos). Isso foi feito com uma função `eliminate_invalid_chars`, que recebe como parâmetro o nome do arquivo e substitui as sequências inválidas por um espaço.

Agora, com dados válidos podendo ser lidos a partir do arquivo, pode-se iniciar o processo de importação. De início, a primeira linha é lida separadamente, pois esta é o cabeçalho do arquivo, usado para determinar o índice das colunas que iremos carregar. A partir daqui, o conteúdo é lido linha a linha, e além de simplesmente armazenar alguns dados originais, são realizados os seguintes passos para obtenção dos *tokens* do corpo da mensagem:

a) remover todas as ocorrências de *tags* do tipo BBCode (ou *Bulletin Board Code*, é uma linguagem de marcação, criada em 1998, usada para formatar textos em fóruns eletrônicos de maneira simples (WIKIPEDIA, 2018b));
b) remover todos os excessos de espaços e quebras de linha;
c) separar as palavras do texto utilizando os espaços entre elas como separador, gerando uma lista (mas mantendo a ordem em que aparecem);
d) cada item da lista é então transformado em sua raiz a fim de agrupar palavras seme-

[14] Disponível em: <https://www.python.org>.
[15] Disponível em: <http://www.pyinvoke.org>. Facilita a criação de tarefas executáveis a partir da linha de comando.

lhantes, o que é feito através de uma ferramenta de *stemming*, presente no NLTK, que implementa o algoritmo RSLP – primeiramente apresentado por Orengo e Huyck (2001), este processo basicamente remove sufixos das palavras, reduzindo variações como "casa" e "casinha" para "cas", como ilustrado na figura 7;

e) são removidas todas ocorrências de *stopwords* – estas são palavras de alta frequência na língua, mas que não oferecem muito conteúdo significativo, como artigos e preposições, logo, são comumente excluídas antes de diversos tipos de processamento de linguagem natural (WIKIPEDIA, 2018g).

Figura 7: Exemplo de *stemming* pelo algoritmo RSLP

```
Python 3.6.3 |Anaconda, Inc.| (default, Nov 20 2017, 20:41:42)
[GCC 7.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import nltk
>>> nltk.stem.RSLPStemmer().stem('casa')
'cas'
>>> nltk.stem.RSLPStemmer().stem('casinha')
'cas'
```

Fonte: própria autoria.

Detalhando um pouco mais a penúltima operação (de *stemming*, importante para obter resultados satisfatórios), é um processo muito discutido no âmbito da língua inglesa e de outras com flexões gramaticais – incluindo o português –, pois exibem resultados mais significativos. O algoritmo comentado, RSLP (ou Removedor de Sufixos para a Língua Portuguesa), compõe-se por diversas etapas, cada uma com suas regras, e está ilustrado na figura 8. Os resultados desse algoritmo conseguiram superar, para o português, os do algoritmo de Porter, que é referência neste tipo de processamento (ORENGO; HUYCK, 2001).

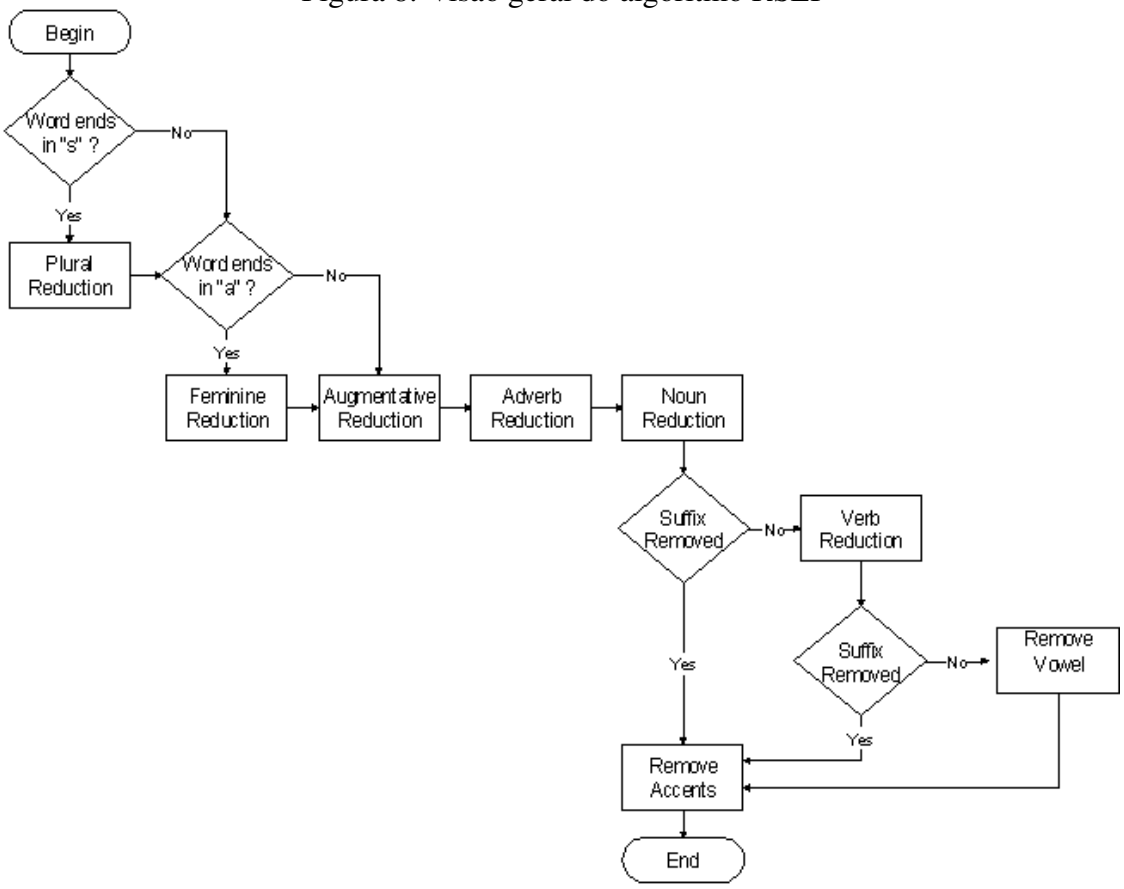
Um breve exemplo do resultado da execução dos passos descritos para o processo de importação desenvolvido pode ser visto na figura 9.

3.5 Serviço de recomendações

O serviço de recomendações foi construído com técnicas comuns hoje em dia em sistemas web. Ele é uma Web API (ou *Application Programming Interface*) que utiliza a metodologia REST (ou *Representational State Transfer*), com troca de mensagens no formato de arquivos JSON (ou *JavaScript Object Notation*).

Como um dos objetivos era não acoplar este serviço à interface utilizada pelo

Figura 8: Visão geral do algoritmo RSLP



Fonte: <http://www.inf.ufrgs.br/~viviane/rslp/>.

usuário, o emprego de um sistema *back-end* desse tipo é adequado, pois será possível consumi-lo a partir de qualquer cliente capaz de se comunicar pela internet (WIKIPEDIA, 2018h), visto que a arquitetura REST permite a comunicação através do protocolo HTTP (WIKIPEDIA, 2018f). Ainda, o formato das mensagens torna muito prática a construção de interfaces web sem muitas dependências, já que esse tipo de objeto (JSON) é nativo ao JavaScript (WIKIPEDIA, 2018e), linguagem que pode ser executada em todos os navegares populares, tanto em computadores convencionais e notebooks como em dispositivos móveis (WIKIPEDIA, 2018d).

A única funcionalidade deste componente é fornecer recomendações de mensagens similares a uma nova. Também é possível escolher se deverão ser consideradas apenas as mensagens dentro do mesmo tópico ou não, caso em que todos fóruns da plataforma serão incluídos na pesquisa.

A partir do identificador da nova mensagem, a indicação de que se quer ou não restringir a pesquisa a um tópico e um número máximo de recomendações a retornar, o serviço então realiza o processo de análise de similaridades entre as mensagens. Os passos para que isso ocorra (baseados, parcialmente, nos processos de categorização descritos

Figura 9: Exemplo de mensagem original e seus *tokens*

	original_body text	body_tokens text[]
1	[quote] Segue em anexo, a atividade. [/quote]	{seg,anex,ativ}
2	Segue atividade 4 em anexo	{seg,ativ,4,anex}
3	Atividade proposta: Nicolau tinha uma ideia...	{ativ,propost,nicolau,ide}
4	Segue em anexo proposta de atividade.	{seg,anex,propost,ativ}
5	Achei sua proposta muito produtiva.	{ach,propost,produ}

Fonte: própria autoria.

por Rizzi et al. (2000)) são descritos abaixo:

a) a definição da coleção de textos a ser utilizada é feita com base no parâmetro `scoped`, que limita ou não a pesquisa ao tópico da mensagem analisada (sendo esta informada pelo parâmetro `new_message_id`), conforme comentado anteriormente);

b) como os termos dos documentos já foram identificados na etapa de importação e pré-processamento (descrita em 3.4.2), aqui apenas se recupera do banco de dados as informações das mensagens;

c) Rizzi et al. (2000) também comentam sobre a definição da lista de *stopwords*, porém, isso já foi feito na seção 3.4.2;

d) ignorando a etapa de categorização em Rizzi et al. (2000), se executa o cálculo de similaridade entre as mensagens da coleção definida e a nova mensagem, o que será melhor descrito na seção 3.5.1;

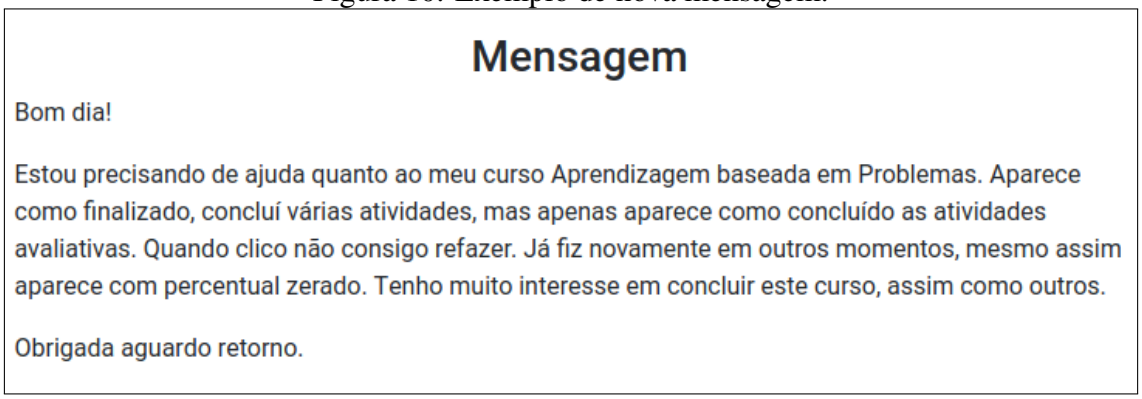
e) por fim, ao obter a lista de mensagens e seus índices de similaridade em relação à nova mensagem, são retornadas no máximo a quantidade de mensagens solicitada através do parâmetro `how_many`.

Para fins de ilustração, pode-se observar nas figuras 10 e 11 um exemplo de mensagem nova, e outro das mensagens julgadas semelhantes pelo sistema, considerando apenas o mesmo tópico.

3.5.1 Cálculo da similaridade entre as mensagens

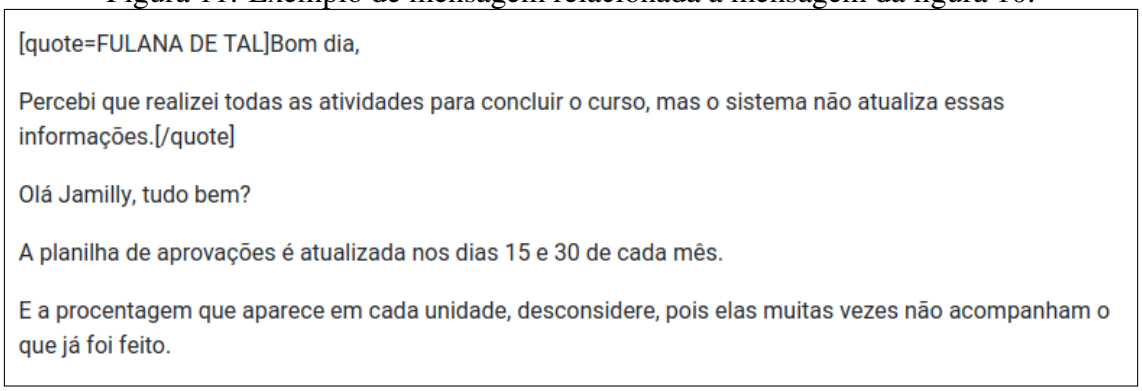
As várias operações realizadas durante a fase de importação e pré-processamento preparam os dados para o cálculo de similaridade, pois aqui é necessário que os textos já estejam, por exemplo, separados em *tokens*. Adicionalmente, os filtros e as transforma-

Figura 10: Exemplo de nova mensagem.



Fonte: própria autoria.

Figura 11: Exemplo de mensagem relacionada à mensagem da figura 10.



Fonte: própria autoria.

ções aplicados (mais especificamente a remoção de *stopwords* e o *stemming*) fazem com que as computações não considerem dados que não acrescentariam valor aos resultados, e também agrupam termos semelhantes a fim de lidar com a variabilidade de representação de conceitos similares ou idênticos.

A seguir, são detalhados os métodos de cálculo de similaridade.

3.5.1.1 Medidas de frequência

Para se encontrar similaridade entre textos, é muito comum a frequência das palavras ser utilizada como peso nas diferentes possibilidades de comparações, portanto, é um fator crucial para obtenção de bons resultados. De acordo com Wives (2004), as principais maneiras de se calcular estas frequências são assim descritas:

a) Frequência absoluta (ou *term frequency*) apenas se preocupa em contar as repetições de cada palavra para um dado texto. Não considera o tamanho deste, muito menos os outros da coleção.

b) Frequência relativa (ou *relative frequency*) consiste em normalizar a frequência absoluta de um termo pela contagem total de termos no texto, resultando em um valor entre zero e um, indicando a relevância do termo para o documento. Ainda assim, não considera o restante da coleção, debilitando a capacidade de contextualizar esta frequência dentro do conjunto.

c) Inverso da frequência nos documentos (ou *inverse document frequency*) faz uso da frequência absoluta dos termos, mas também considera a quantidade de documentos em que os termos aparecem. A fórmula utilizada neste trabalho para o cálculo desta frequência é $idf(d, f) = \log \frac{n}{df(d,t)}$, onde n é o número total de documentos, d é a coleção de documentos, t é o termo, e $df(d, t)$ é o número de documentos em que o termo aparece. Com isso, obtemos um valor entre zero e um que, quando próximo de 1, indica que o termo aparece em poucos documentos.

Neste trabalho, a representação das mensagens é a conhecida por *bag-of-words*, em que – normalmente – não se assume ordem ou relação entre as palavras. Contudo, por decorrência de características das ferramentas utilizadas no desenvolvimento, a ordem dos *tokens*, mesmo após todas as transformações, se mantém. Sendo assim, além das palavras serem consideradas independentemente (unigramas), elas também são consideradas aos pares consecutivos (bigramas), o que aumenta bastante a quantidade de termos a serem processados, mas também amplia a eficácia na detecção de similaridade entre documentos[16].

3.5.1.2 Medida de similaridade

Existem diferentes abordagens possíveis para se calcular a semelhança entre dois documentos. Aqui será descrita apenas a similaridade do cosseno. Por ser relativamente simples, popular e, mesmo assim, apresentar resultados bons, ela está embutida em diversas bibliotecas que contêm funções de processamento de linguagem natural – PLN (ou NLP para *Natural Language Processing*). Por esses motivos, foi a escolhida.

A similaridade do cosseno consiste em obter o cosseno do ângulo entre dois vetores. Isso é útil devido à propriedade do cosseno de resultar em 1 para vetores idênticos, e 0 para vetores ortogonais. Ainda, se os vetores forem unitários – o que acontece quando se utiliza um esquema de pesos normalizado, como o TF-IDF (*Term Frequency–Inverse*

[16]Relação observada empiricamente pelo autor durante a fase de testes das diferentes formas de cálculo dos pesos para os termos, executando a função de recomendação dezenas de vezes para mensagens distintas. Incluindo-se trigramas, não foi observada melhora.

Document Frequency), por exemplo –, o cosseno do ângulo entre eles é equivalente ao produto escalar (SINGHAL et al., 2001, p. 3), logo, podemos simplificar a função de similaridade para algo como:

$$Sim(\vec{D}, \vec{Q}) = \sum_{t_i \in Q,D} w_{t_{iQ}} \cdot w_{t_{iD}}, \tag{3.1}$$

onde $\vec{D}$ é o vetor de um documento da coleção, $\vec{Q}$ é o vetor do documento de consulta (a nova mensagem, no caso deste trabalho), enquanto $w_{t_{iQ}}$ e $w_{t_{iD}}$ são os pesos da i-ésima componente dos vetores $\vec{Q}$ e $\vec{D}$, respectivamente.

3.5.1.3 Utilizando a medida de similaridade

Nesta subseção, serão detalhadas as ações do serviço de recomendação a partir do momento em que este recebe uma requisição. Esta, por sua vez, conterá o identificador da nova mensagem, uma *flag* indicando se deve-se ou não restringir a pesquisa ao tópico, e um número máximo de recomendações a retornar.

O primeiro passo é buscar do banco de dados as informações da nova mensagem e de todas as mensagens da coleção a ser considerada. Realiza-se uma primeira consulta na tabela `messages`, filtrando pelo identificador da nova mensagem recebida na requisição.

Em seguida, a fim de se obter a coleção de mensagens passadas, verificamos o parâmetro `scoped` da requisição: caso seja igual a 1, buscamos apenas os documentos com o campo `original_root_id` idêntico ao da nova mensagem; caso contrário, todas as mensagens são recuperadas. Abaixo, o trecho de código referente aos passos descritos até aqui (simplificado):

```
session = db.Session()
new_msg = session.query(db.Message).filter(
    db.Message.original_id == new_msg_id).first()
current_scope = new_msg.original_root_id if scoped else None

if current_scope:
    corpus = session.query(db.Message).filter(
        db.Message.original_root_id == current_scope).all()
else:
    corpus = session.query(db.Message).all()
```

Agora, com todos os dados em memória, pode-se começar a calcular a matriz TF-IDF (composta pelos vetores correspondentes às mensagens da coleção), bem como o vetor da nova mensagem. Em um primeiro momento, isso foi implementado manualmente conforme as técnicas apresentadas anteriormente. Para a versão final a ser exibida aqui, por motivos de performance, foram utilizadas rotinas contidas na biblioteca Scikit-learn (PEDREGOSA et al., 2011). Em especial, as funções da classe `TfidfTransformer`, que servem para transformar as sequências de *tokens* em vetores de características que contêm os valores TF-IDF.

Para calcular a matriz TD-IDF da coleção de mensagens, primeiro transformamos esta em vetores que a biblioteca Scikit-learn reconhece. Fazemos isso instanciando um objeto da classe `TfidfTransformer`, o configurando para considerar unigramas e bigramas (os parâmetros `tokenizer`, `stop_words` e `lowercase` recebem valores que fazem a biblioteca não alterar os *tokens*, pois eles já sofreram todas as transformações necessárias):

```
vectorizer = TfidfVectorizer(
    tokenizer=useless_tokenizer,
    stop_words=None,
    lowercase=False,
    ngram_range=(1, 2))
```

Então, podemos carregar neste objeto os *tokens* correspondentes às mensagens que constam em nossa coleção, já calculando a matriz TF-IDF:

```
documents = [x.body_tokens for x in corpus]
tfidf = vectorizer.fit_transform(documents)
```

Em seguida, calculamos o vetor de pesos para a nova mensagem (aqui realizando o processamento para obter os *tokens*, mesmo eles já estando no banco de dados, pois assim é possível simular a situação em que a mensagem ainda não estaria salva):

```
new_message_tfidf = vectorizer.transform(
    [utils.tokenize(new_msg.original_body)])
```

Com essas informações, é possível, em uma única operação (através da função `linear_kernel` do Scikit-learn, neste caso, equivalente ao produto escalar e à simila-

ridade do cosseno[17]), calcular a semelhança de todos os documentos da coleção contra a nova mensagem:

```
cosine_similarities = linear_kernel(new_message_tfidf,
                                    tfidf).flatten()
```

Agora, com um único vetor contendo os índices de semelhanças, podemos ordená-lo e pegar os índices dos documentos correspondentes:

```
docs_indices = cosine_similarities.argsort()[::-1]
```

A partir dessa lista de índices, conseguimos acessar os documentos originais em conjunto com seus índices de similidade à nova mensagem, limitando o número de resultados ao parâmetro `how_many`, enviado na requisição:

```
hits = []
for idx in docs_indices[:(how_many + 1)]:
    msg = self.message_corpus[idx]
    hits.append({
        'id': msg.original_id,
        'root_id': msg.original_root_id,
        'similarity': self.cosine_similarities[idx],
        'body': msg.original_body})
```

Um detalhe importante a ser observado no trecho acima é a expressão how_many+1. Essa soma serve para compensar o fato de que a nova mensagem para a qual estamos buscando recomendações também se encontra entre a coleção de mensagens. Por conseguinte, a primeira recomendação (com índice igual a 1) será a própria mensagem usada na pesquisa.

A lista de recomendações pode então ser retornada ao cliente, destacando a primeira delas como sendo a mensagem de pesquisa original:

```
return json.dumps({
    'query': hits[0],
    'results': hits[1:]})
```

[17]Como pode ser verificado no manual de referência da biblioteca, em <http://scikit-learn.org/stable/modules/metrics.html#cosine-similarity>.

3.6 Interface do usuário

A interface do usuário foi construída como um sistema à parte, utilizando a linguagem de programa Ruby[18] e o *framework* Rails[19]. A escolha dessas tecnologias se deve ao fato da experiência do autor do presente trabalho com elas, e também porque são conhecidas por serem excelente ferramentas de prototipagem de aplicações web.

O propósito aqui é avaliar as recomendações e, analisando os resultados destas avaliações, decidir se a ferramente foi útil ou não. Isso poderá ser feito posteriormente através das informações registradas no banco de dados.

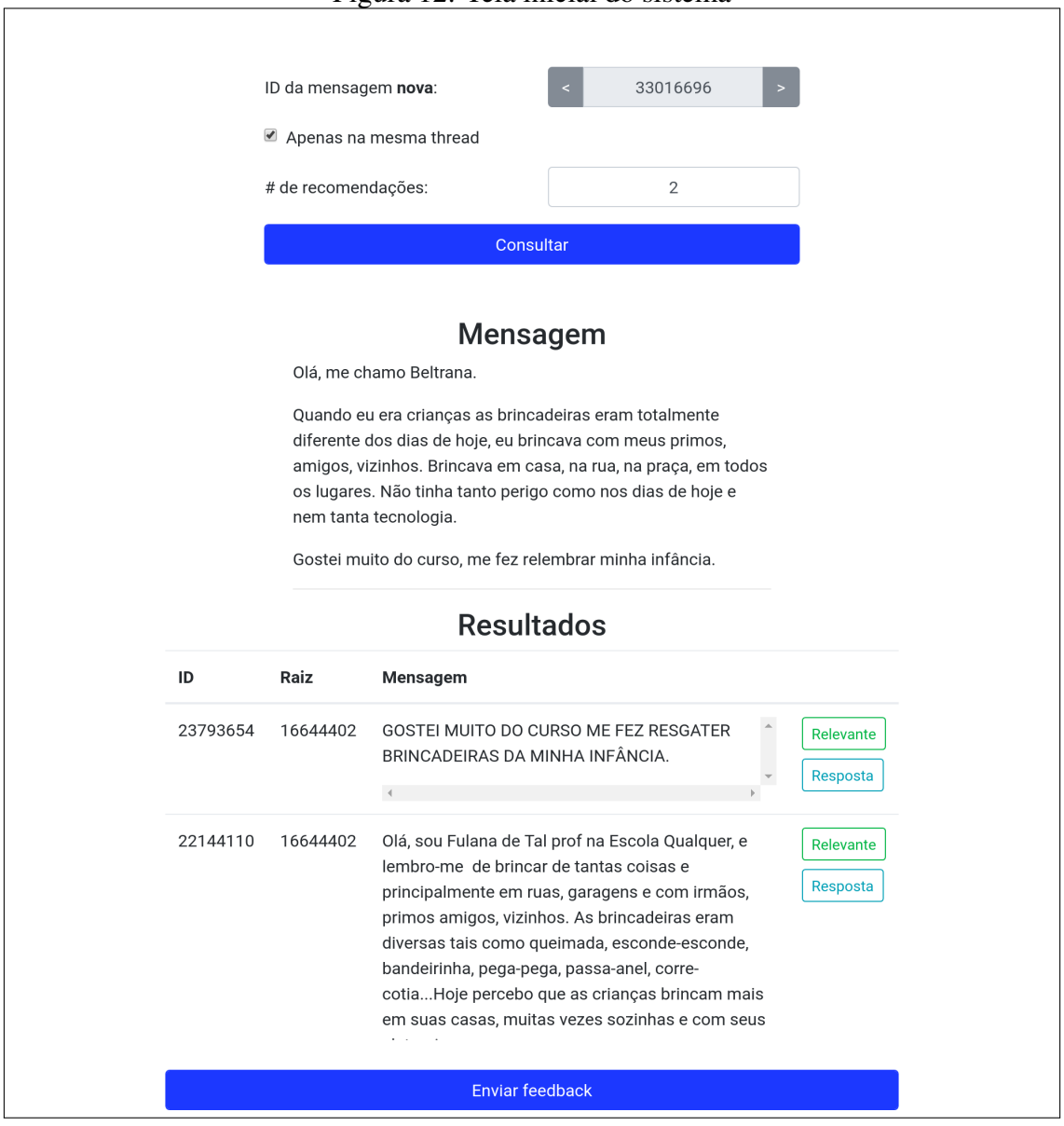
Por praticidade, a aplicação consome o mesmo banco de dados do serviço de recomendação. Logo na abertura, ela já apresenta a mensagem mais recente encontrada, e algumas recomendações de mensagens, além de opções de configuração das consultas, como pode ser visto na figura 12. Também é possível ver que, ao lado de cada recomendação, foram incluídos botões para marcar estas como válidas (botão "Relevante") e/ou se respondem a uma possível pergunta feita na nova mensagem (botão "Resposta", todavia, este não foi utilizado para o presente trabalho, tendo sido incluído para futuras análises). Ainda, para fins de usabilidade, é possível clicar em uma mensagem para ver seu conteúdo na íntegra, como ilustra a figura 13.

Ao clicar no botão presente no fim da página, "Enviar feedback", são salvos no banco de dados registros referenciando a nova mensagem a cada uma das recomendações, junto com a variável que sinaliza se a recomendação é relevante. Subsequentes envios do mesmo par de mensagens nova e recomendada apenas atualiza os valores. A tabela que salva essas informações se chama `user_reviews` e contém os seguintes campos:

a) `query_message_id`, referente ao identificador da "nova mensagem", utilizada como elemento de pesquisa nas requisições ao servidor de recomendações;
b) `related_message_id`, contendo o identificador da mensagem recomendada;
c) `is_related`, uma *flag* informando se essa relação é relevante ou não.

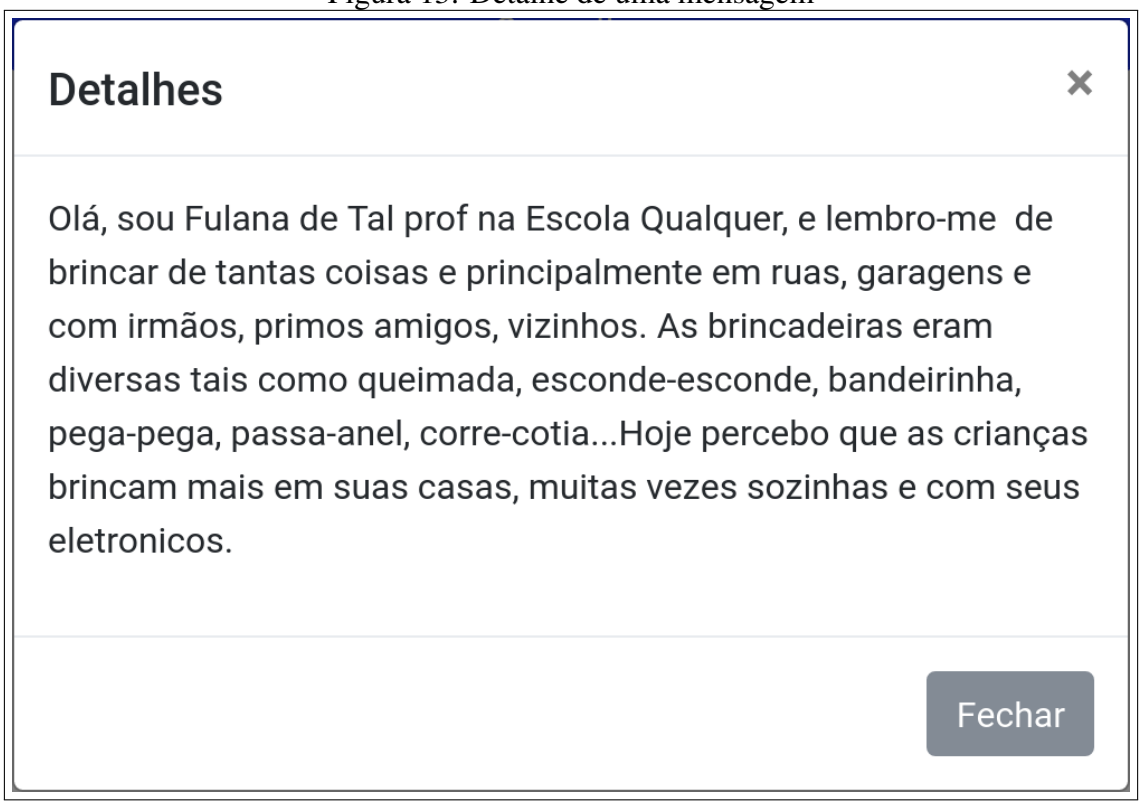
[18] Disponível em: <https://www.ruby-lang.org/pt/>.
[19] Disponível em: <https://rubyonrails.org>.

Figura 12: Tela inicial do sistema



Fonte: própria autoria.

Figura 13: Detalhe de uma mensagem

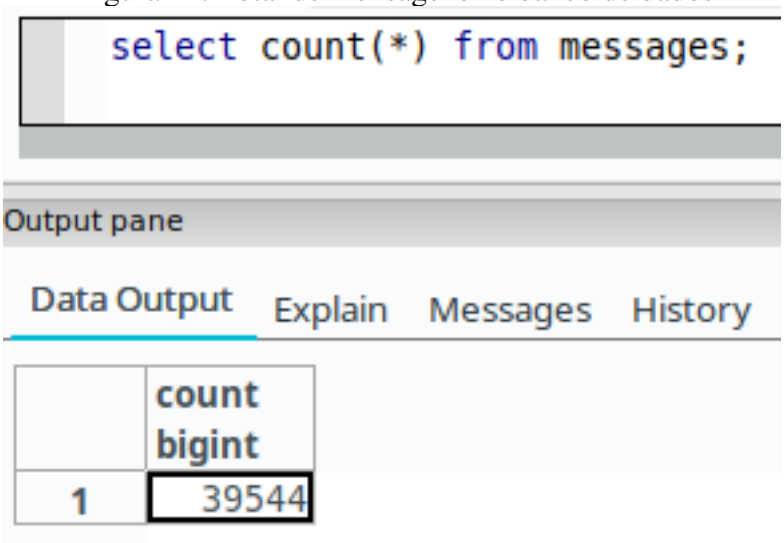


Fonte: própria autoria.

4 TESTES COM A FERRAMENTA

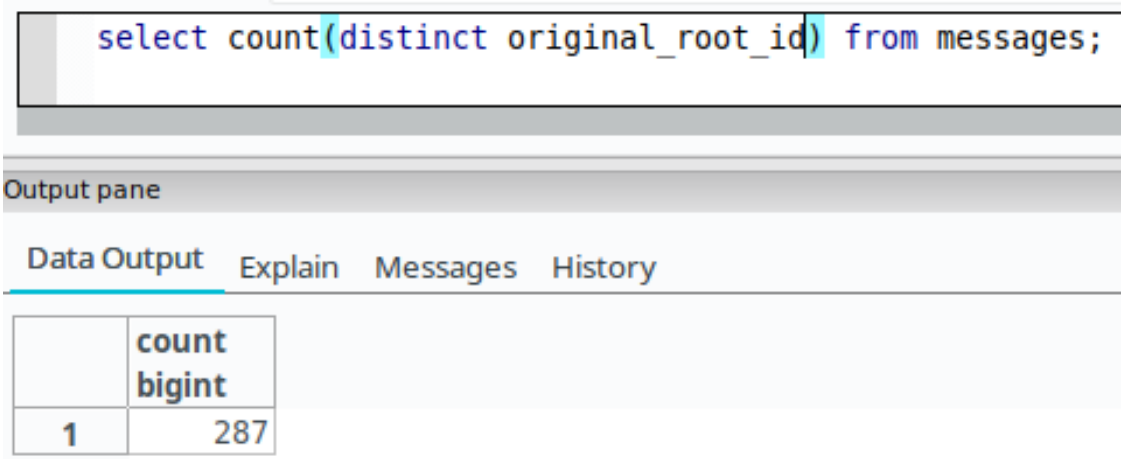
A fim de verificar se a ferramenta é capaz de retornar recomendações relevantes e que poderiam ser utilizadas em uma situação real, foram efetuados testes, pelo próprio autor deste trabalho, com dezenas de mensagens. O banco de dados utilizado contém 39.544 mensagens (figura 14), distribuídas entre 287 tópicos (figura 15). Os números nos levam a pensar em uma média de 137 mensagens por tópico, mas, na realidade, o que ocorre é que alguns cursos são muito mais populares e contém a maioria das mensagens.

Figura 14: Total de mensagens no banco de dados



Fonte: própria autoria.

Figura 15: Total de tópicos no banco de dados



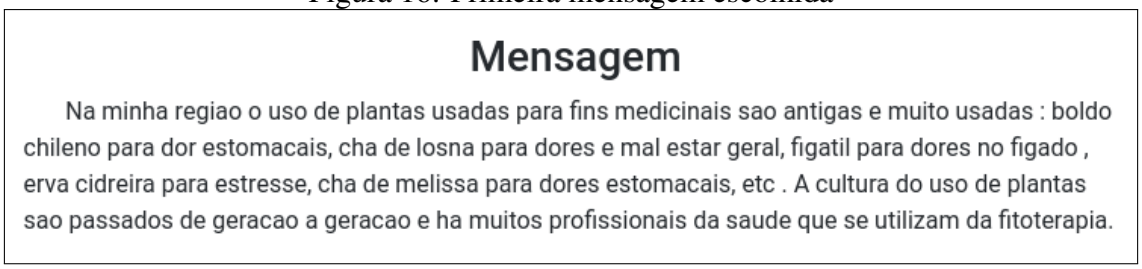
Fonte: própria autoria.

Estes dados foram importados de um arquivo real vindo da equipe de desenvolvimento da plataforma ScolarTIC. A data do backup utilizado é de 10 de outubro de 2017.

4.1 Primeiro exemplo

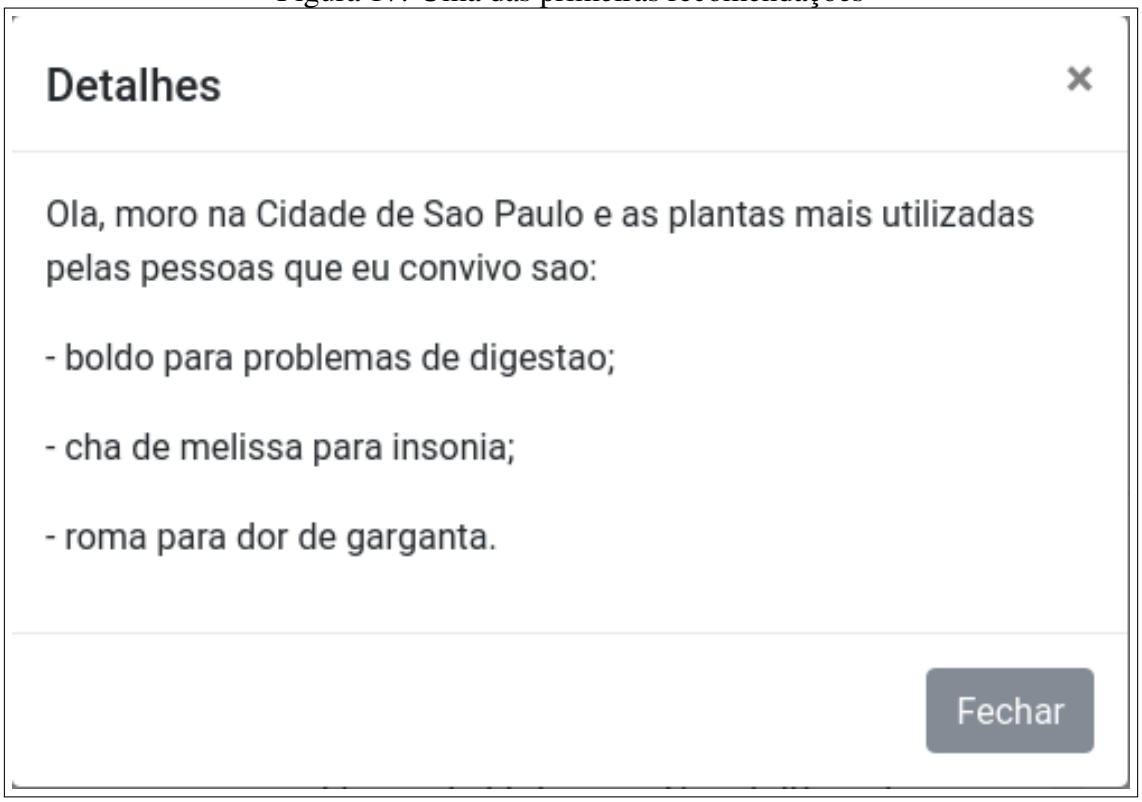
A figura 16 mostra uma mensagem escolhida ao acaso. Nas figuras 17 e 18, podemos ver duas das recomendações feitas, ranqueadas como segunda e quarta mais semelhantes, respectivamente. As recomendações intermediárias a estas (primeira e terceira) não foram escolhidas, pois são respostas de tutoras a outras mensagens, portanto, quase a totalidade de seu corpo é uma citação, e, ao final, apenas um agradecimento.

Figura 16: Primeira mensagem escolhida



Fonte: própria autoria.

Figura 17: Uma das primeiras recomendações



Fonte: própria autoria.

4.2 Segundo exemplo

Neste exemplo, foi escolhida uma mensagem de assunto diferente. Aqui se fala sobre a água, como pode ser visto na figura 19. Neste caso, as recomendações podem ser consideradas excelentes, pois todas as do topo da lista apresentada são de fato semelhantes, mas com informações distintas (ou apresentadas de maneiras diferentes).

As recomendações exibidas nas figuras 20 e 21 são as duas primeiras apresentadas para esta mensagem. As demais, não ilustradas aqui, seguem o mesmo viés, embora algumas sejam apenas comentários de professoras que contêm outra mensagem citada.

4.3 Terceiro exemplo

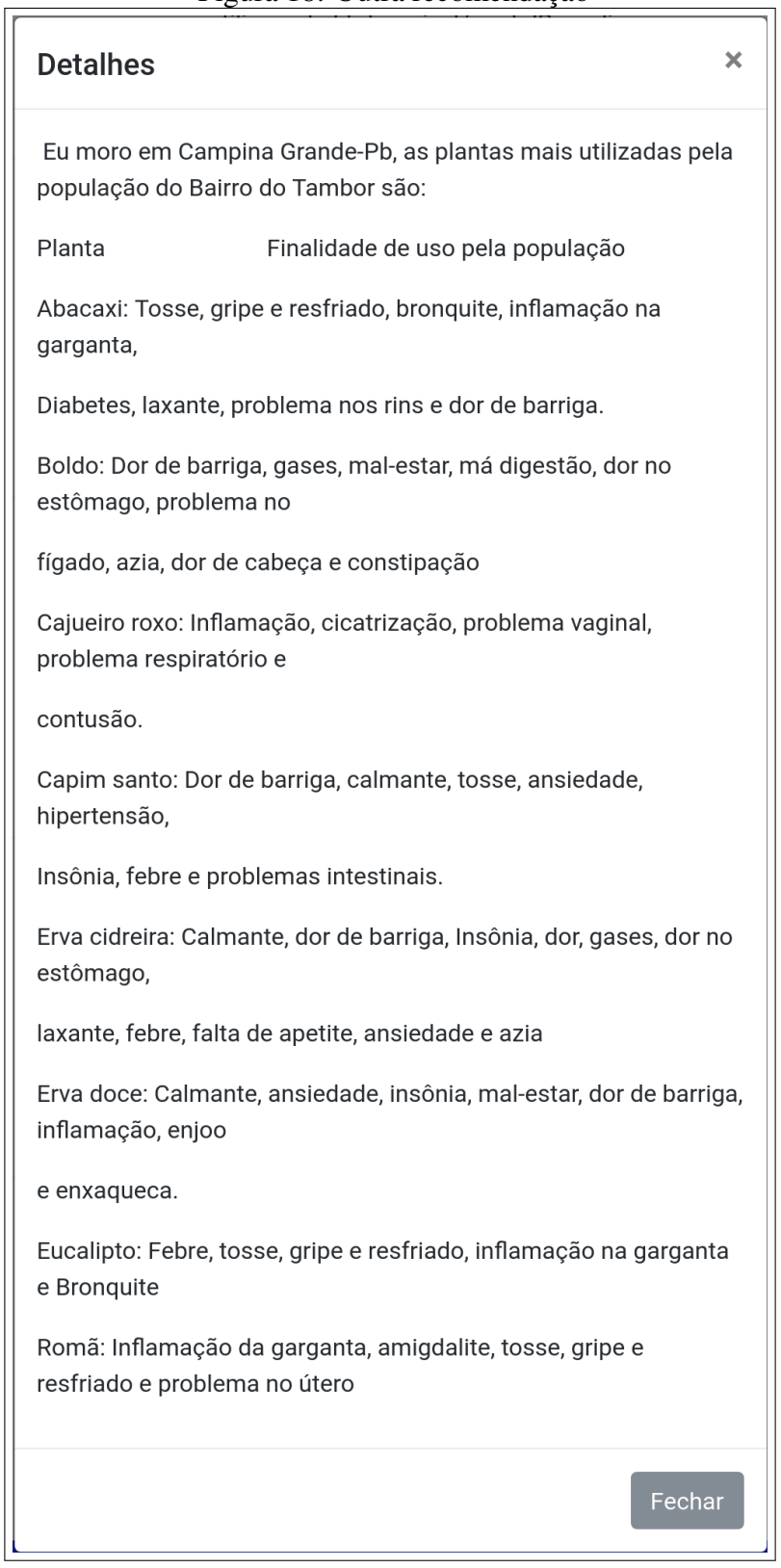
Como será mostrado neste exemplo, quando as mensagens são muito curtas, o sistema não consegue encontrar recomendações adequadas. Isso se deve ao fato de, ao reduzir muito o número de termos, ficarmos praticamente sem características para considerar, o que ainda pode ser agravado se as poucas palavras forem comuns entre as outras mensagens da coleção (devido às características do TF-IDF citadas no capítulo 3).

A figura 22 mostra um caso de mensagem muito curta. Nas figuras 23 e 24 podemos ver que as recomendações não são relevantes.

4.4 Quarto exemplo

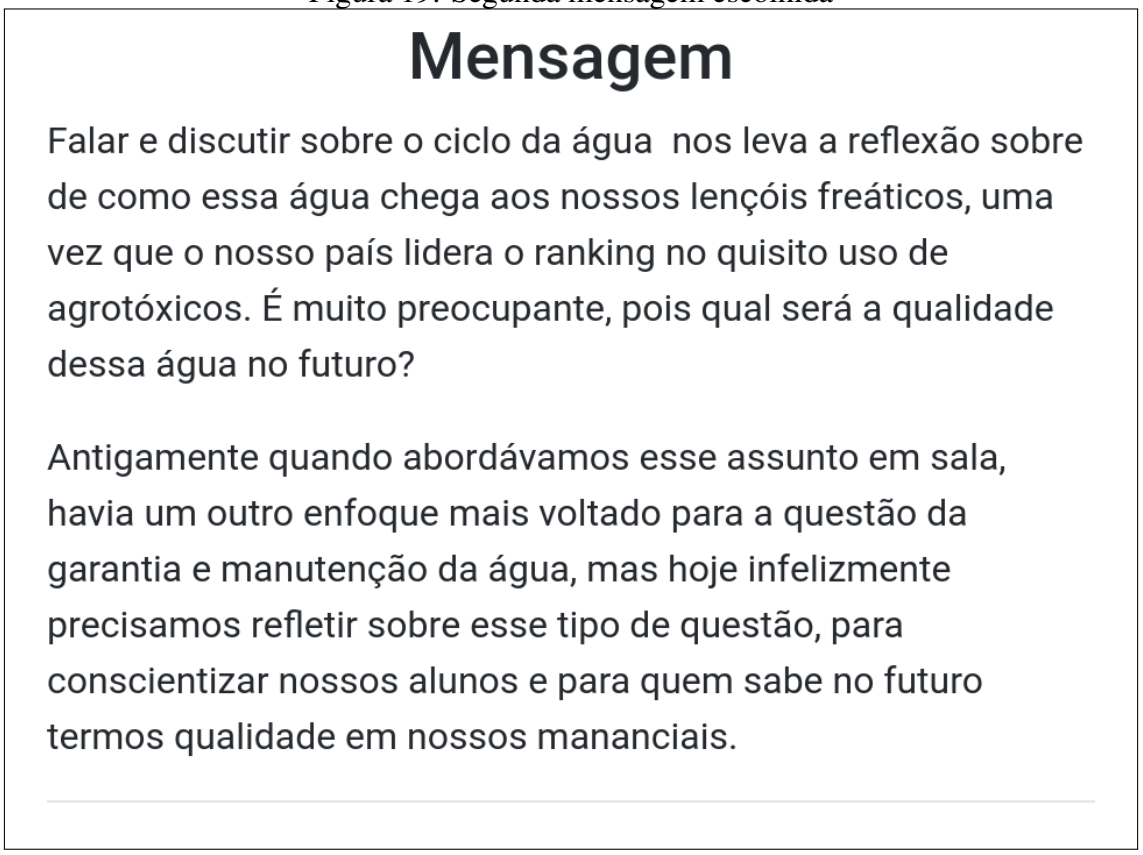
Este exemplo serve para ilustrar os casos em que as recomendações respondem a uma questão repetida. Na figura 25, vemos que a mensagem contém uma dúvida de um aluno que não consegue encontrar um determinado botão na página do curso. Na figura 26, nota-se que outra pessoa teve um problema similar, e a professora solucionou a dúvida. Já na figura 27, verifica-se mais uma pergunta bastante parecida.

Figura 18: Outra recomendação



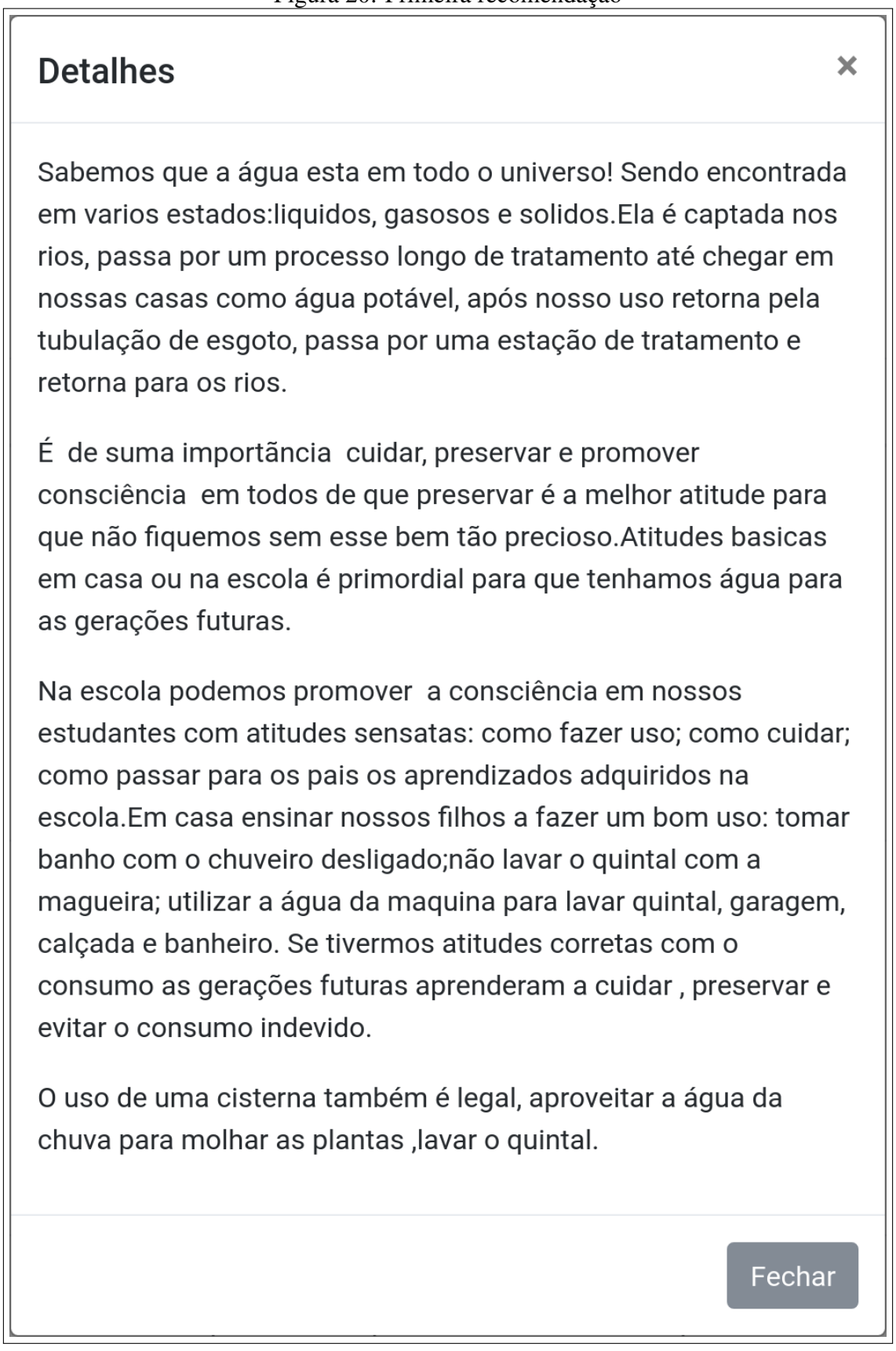
Fonte: própria autoria.

Figura 19: Segunda mensagem escolhida



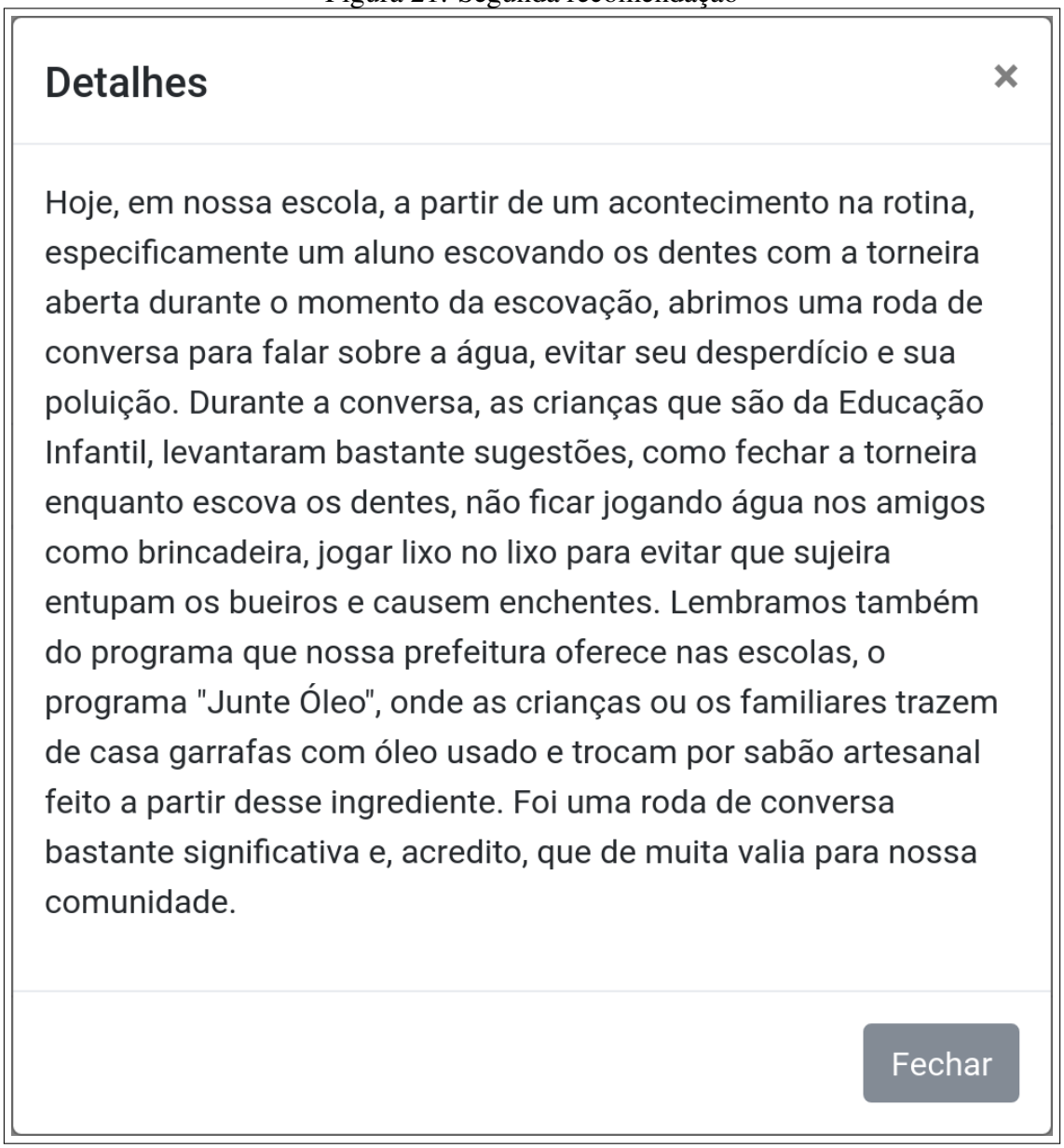
Fonte: própria autoria.

Figura 20: Primeira recomendação



Fonte: própria autoria.

Figura 21: Segunda recomendação

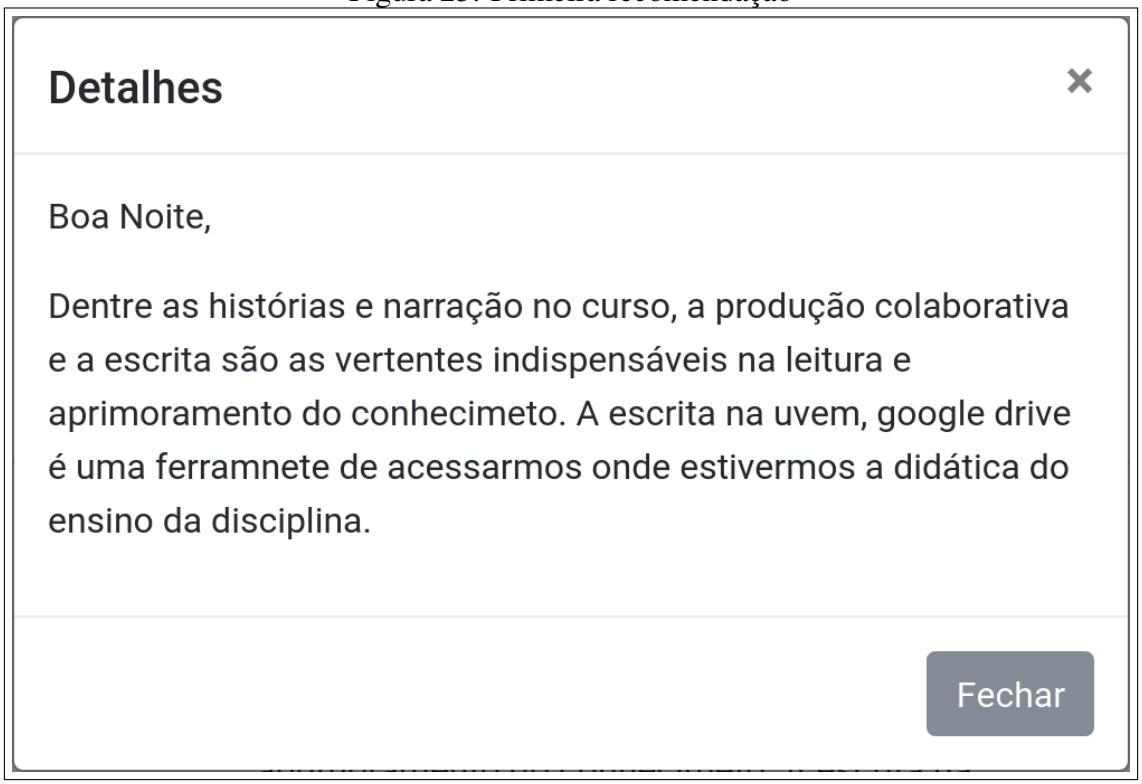


Fonte: própria autoria.

Figura 22: Terceira mensagem escolhida

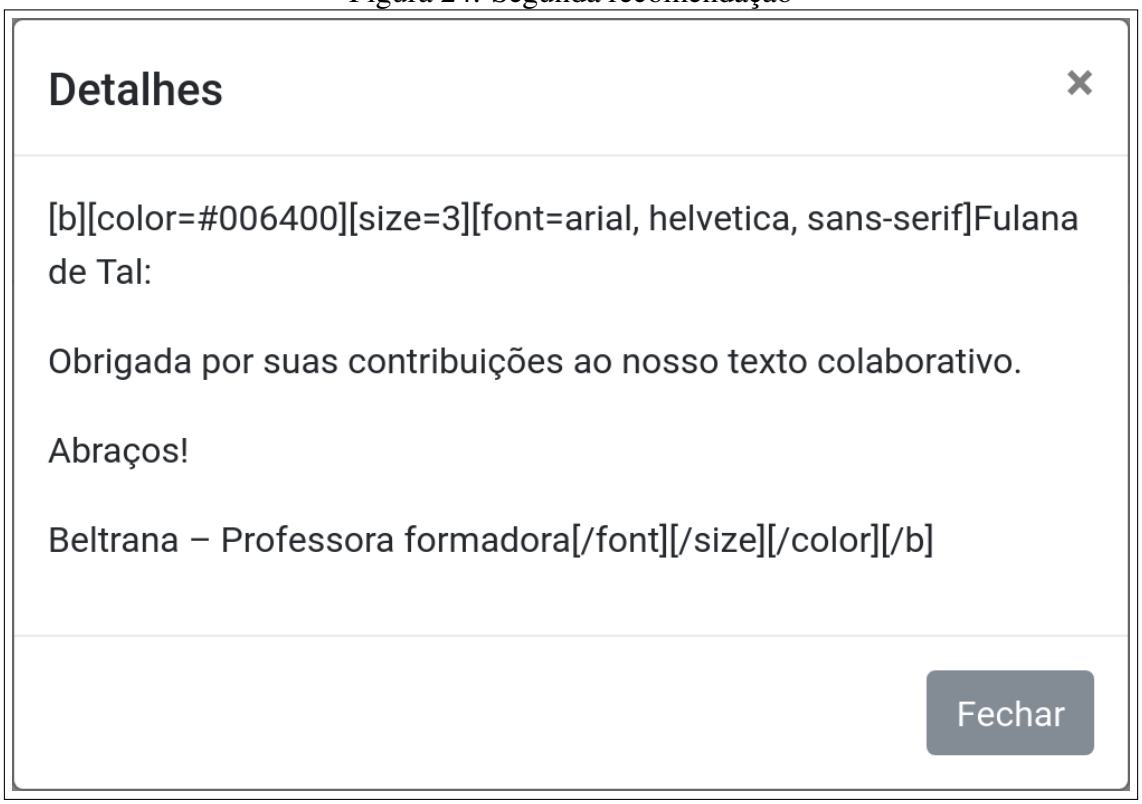
Fonte: própria autoria.

Figura 23: Primeira recomendação



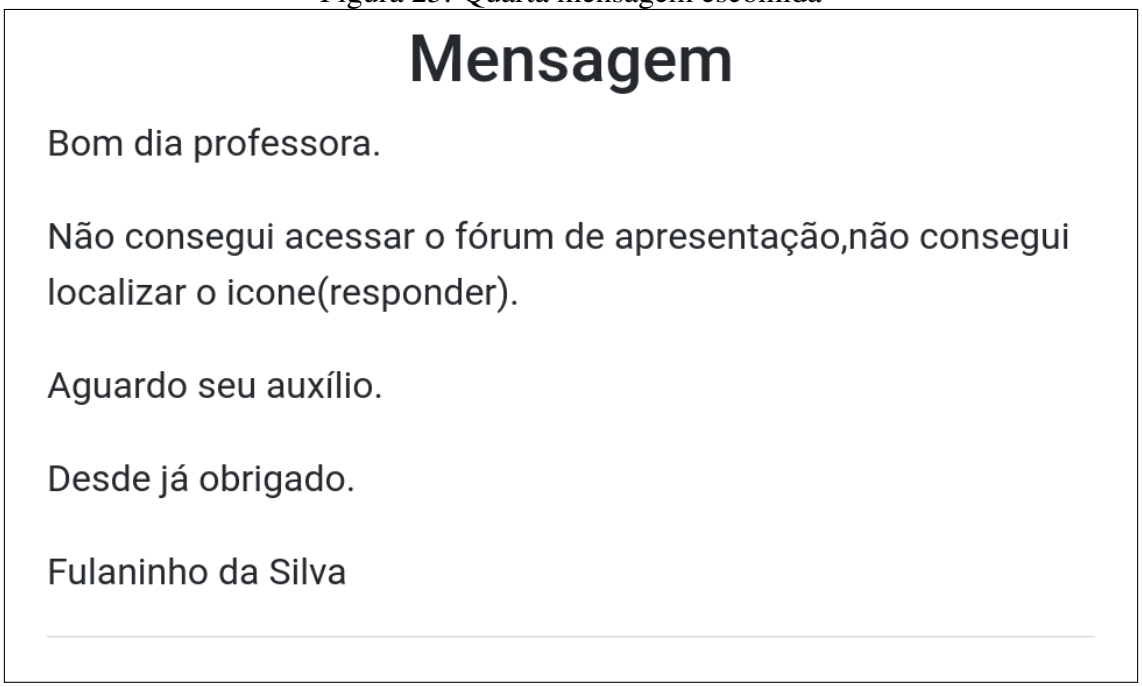
Fonte: própria autoria.

Figura 24: Segunda recomendação



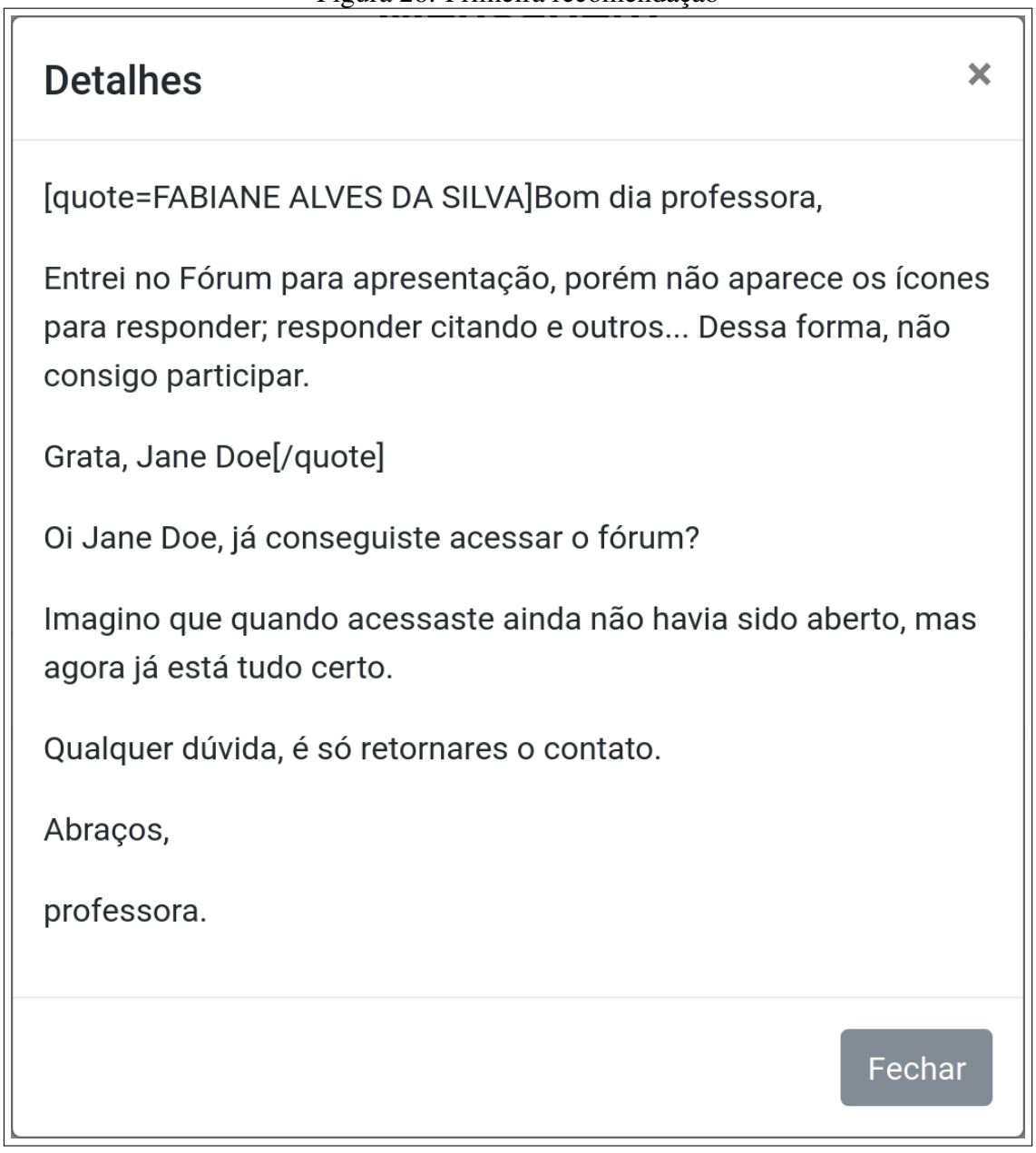
Fonte: própria autoria.

Figura 25: Quarta mensagem escolhida



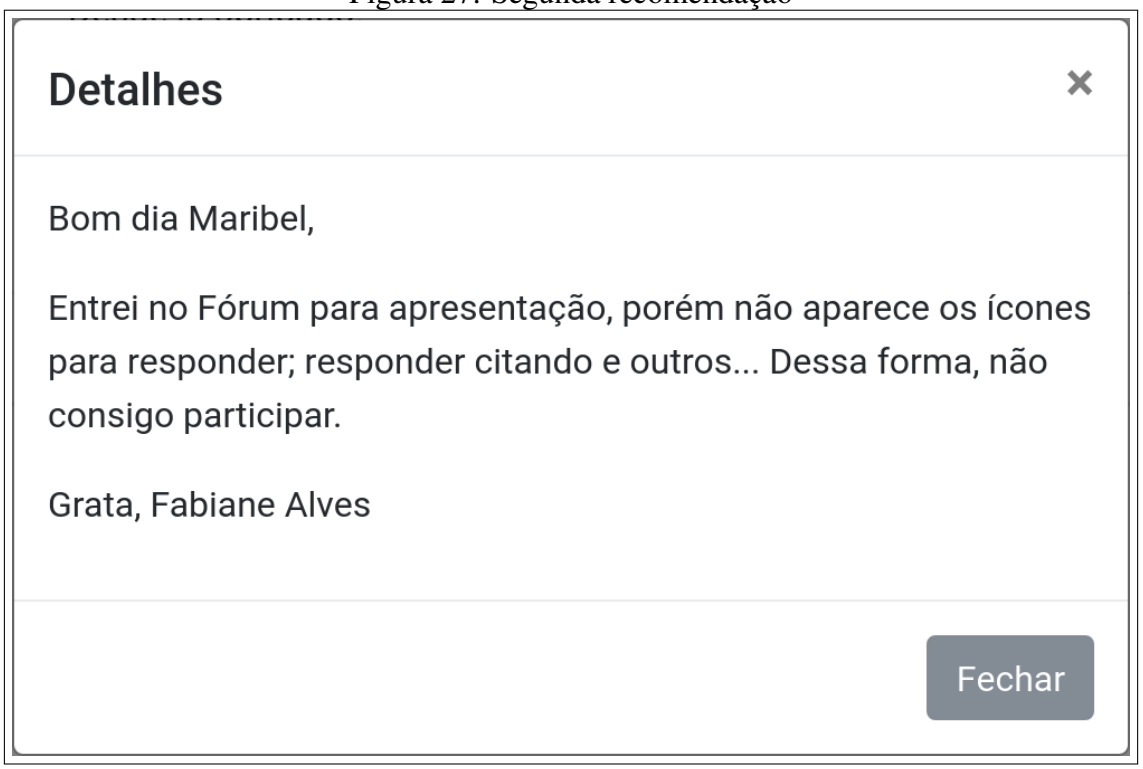
Fonte: própria autoria.

Figura 26: Primeira recomendação



Fonte: própria autoria.

Figura 27: Segunda recomendação



Fonte: própria autoria.

5 CONCLUSÃO

No início deste trabalho, foi visto que a educação a distância cresceu e continua crescendo muito no âmbito global. Tanto a quantidade de alunos quanto a oferta de cursos têm apresentado um aumento impressionante nos últimos anos.

Neste contexto, com cursos do tipo MOOC (*Massive Open Online Course* sendo o caso mais extremo), o número de alunos pode ser bastante alto, chegando aos milhares. Isso gera problemas no tocante à qualidade do atendimento prestado ao aprendiz, pois, tipicamente, a quantidade de professores é limitada.

Foi proposta, então, uma ferramenta que tenta auxiliar os alunos e professores no que diz respeito a interações via mensagens de texto em um ambiente de ensino online. Seu objetivo é evitar que sejam feitas tantas perguntas repetidas, uma vez que tal atitude sobrecarrega os tutores desnecessariamente. Isso pôde ser realizado através de um sistema de recomendação, com uso das técnicas de filtragem baseada em conteúdo, *term frequency–inverse document frequency*.

Ainda, a fim de tornar o software modular, ele foi divido em duas grandes partes. Uma é o serviço de recomendação propriamente dito, e outra é a interface web. Isso permitiu o uso de tecnologias que facilitam o tratamento de linguagem natural e cálculos avançados na construção do *back-end*, e, também para o *front-end*, o emprego de ferramentas que propiciam uma rápida prototipagem de aplicações web.

Considera-se que a contribuição deste estudo para a comunidade é um sistema que poderá servir como prova de conceito para outros mais complexos que façam proveito do conjunto de técnicas exploradas. Embora estas funcionem muito bem juntas, acredita-se que para um sistema real seja necessário explorar mais possibilidades que, possivelmente, retornem sugestões ainda mais interessantes. Entretanto, seria apropriado realizar testes com usuários reais (tanto alunos quanto professores), a fim de coletar dados sobre a qualidade geral das recomendações.

Devido às limitações apresentadas na seção 3.1, não foi possível integrar o sistema desenvolvido diretamente na plataforma ScolarTIC. Considerando isso, pensou-se que o SR criado poderia ser usado para disparar e-mails aos alunos com recomendações de mensagens semelhantes, para tentar responder suas questões através de dúvidas expostas no passado, e apresentar mensagens que falem de assuntos correlatos e, portanto, instiguem novas discussões.

Durante os testes, foi verificado um nível satisfatório de relevância das recomen-

dações. Em geral, com cinco recomendações, pelo menos uma era inteiramente relevante à mensagem de pesquisa, seja por abordar o mesmo assunto ou por responder à pergunta feita.

Desenvolvimentos futuros que provavelmente melhorariam a ferramenta incluem novas maneiras de calcular os pesos das *features* extraídas, novas funções de similaridade, bem como técnicas adicionais a serem empregadas em alguns pontos. Um exemplo seria, após testes reais com usuários, usar aprendizado de máquina sobre os resultados, para que o sistema consiga avaliar sozinho (com acurácia suficiente) quais recomendações são de fato relevantes. Ainda, talvez também fosse interessante comparar os resultados obtidos com este SR versus outros sistema de recuperação de informações existentes no mercado (como o Apache Lucene[20], por exemplo).

[20]Disponível em: <https://lucene.apache.org/core/>

REFERÊNCIAS

ADOMAVICIUS, G.; TUZHILIN, A. 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.

BALAVANOVIC, M.; SHOHAM, Y. Content-based, collaborative recommendation. **Communications of the ACM**, ACM, v. 40, n. 3, p. 66–72, 1997.

BARLA, M. Towards social-based user modeling and personalization. **Information Sciences and Technologies Bulletin of the ACM Slovakia**, v. 3, n. 1, p. 52–60, 2011.

BEZERRA, B. L. D. **Estudo de algoritmos de filtragem de informação baseados em conteúdo**. [S.l.]: Recife, 2002.

CLASS CENTRAL. **By The Numbers: MOOCS in 2017**. 2018. Disponível em: <https://www.class-central.com/report/mooc-stats-2017>. Acesso em: 18 jan. 2018.

ELAHI, M.; RICCI, F.; RUBENS, N. A survey of active learning in collaborative filtering recommender systems. **Computer Science Review**, Elsevier, v. 20, p. 29–50, 2016.

FISCHER, C. T. **Recomendação de Conteúdo em Fóruns Eletrônicos**. 2005. 82 p. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação), Instituto de Ciências Exatas e Tecnológicas – Centro Universitário Feevale, Novo Hamburgo, Brasil.

GOLDBERG, D. et al. Using collaborative filtering to weave an information tapestry. **Communications of the ACM**, ACM, v. 35, n. 12, p. 61–70, 1992.

MACMANUS, R. **A Guide to Recommender Systems**. 2009. Disponível em: <https://web.archive.org/web/20090220013113/http://www.readwriteweb.com/archives/recommender_systems.php>. Acesso em: 4 fev. 2018.

MOONEY, R. J. Collaborative filtering & content-based recommending. 2010.

OLIVEIRA, A. d. M.; CARNEIRO, M. L. F.; MUNHOZ, A. M. Análise do ambiente virtual moodle como tecnologia de apoio aos estudantes de biblioteconomia. 2009.

ORENGO, V. M.; HUYCK, C. A stemming algorithm for the portuguese language. In: IEEE. **String Processing and Information Retrieval, 2001. SPIRE 2001. Proceedings. Eighth International Symposium on**. [S.l.], 2001. p. 186–193.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

RESNICK, P.; VARIAN, H. R. Recommender systems. **Commun. ACM**, ACM, New York, NY, USA, v. 40, n. 3, p. 56–58, mar. 1997. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/245108.245121>.

RICH, E. User modeling via stereotypes. **Cognitive science**, Wiley Online Library, v. 3, n. 4, p. 329–354, 1979.

RIZZI, C. et al. Fazendo uso da categorização de textos em atividades empresariais. In: **International Symposium on Knowledge Management/Document Management (ISKM/DM 2000), III**. [S.l.: s.n.], 2000.

SARWAR, B. et al. Item-based collaborative filtering recommendation algorithms. In: **Proceedings of the 10th International Conference on World Wide Web**. New York, NY, USA: ACM, 2001. (WWW '01), p. 285–295. ISBN 1-58113-348-0. Disponível em: <http://doi.acm.org/10.1145/371920.372071>. Acesso em: 13 mar. 2018.

SINGHAL, A. et al. Modern information retrieval: A brief overview. **IEEE Data Eng. Bull.**, v. 24, n. 4, p. 35–43, 2001.

Torres Júnior, R. D. Combining collaborative and content-based filtering to recommend research papers. 2004.

WIKIPEDIA. **Amazon (company)**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Amazon_(company)>. Acesso em: 4 fev. 2018.

WIKIPEDIA. **BBCode**. 2018. Disponível em: <https://en.wikipedia.org/wiki/BBCode>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **Comma-separated values**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Comma-separated_values>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **JavaScript**. 2018. Disponível em: <https://en.wikipedia.org/wiki/JavaScript>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **JSON**. 2018. Disponível em: <https://en.wikipedia.org/wiki/JSON>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **Representational state transfer**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Representational_state_transfer>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **Stop words**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Stop_words>. Acesso em: 15 mai. 2018.

WIKIPEDIA. **Web API**. 2018. Disponível em: <https://en.wikipedia.org/wiki/Web_API>. Acesso em: 15 mai. 2018.

WIVES, L. K. Utilizando conceitos como descritores de textos para o processo de identificação de conglomerados (clustering) de documentos. 2004.

ZAMBERLAN, D. M. **Recomendação de Mensagens no Fórum Moodle**. 2011. 57 p. Trabalho de Conclusão de Curso (Bacharelado em Ciência da Computação), Instituto de Informática – Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil.