Análise do uso de feedback de relevância no Sistema de Integração Lattes-Qualis (SILQ)

Carlos Bonetti[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brazil

carlosbonetti.mail@gmail.com

Abstract. *SILQ emerged in 2015 with the purpose of matching the vehicles qualified by Qualis with vehicles of publications registered on the Lattes curriculum of researchers, in order to automate the process of generating quality indicators of scientific production from these curriculums. The purpose of this work is a set of modifications to the system to include the most updated Qualis data, a study of the system's accuracy and the inclusion of controls allowing the users to suggest matchings, a technique named relevance feedback, proposing two new algorithms and an experimental analysis to evaluate their efficiency at the system.*

Resumo. *O SILQ surgiu em 2015 com o objetivo de realizar o matching automático dos veículos qualificados no Qualis com os veículos de publicações cadastradas no currículo Lattes de pesquisadores, a fim de automatizar o processo de geração de indicadores das produções contidas nesses currículos. Este trabalho propõe um conjunto de alterações no sistema para inclusão dos dados Qualis mais recentes, um estudo da taxa de acerto do sistema e a inclusão de controles que permitam ao usuário sugerir matching, técnica denominada feedback de relevância, propondo dois novos algoritmos e uma avaliação experimental para análise de sua eficácia no sistema.*

1. Introdução

A Plataforma Lattes, criada e mantida pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), é um sistema de informação responsável pela integração da base de dados de currículos, grupos de pesquisa e instituições. O Currículo Lattes se tornou o padrão nacional no registro da vida científica de estudantes, professores e pesquisadores e é hoje adotada por institutos e universidades de todo o país [CNPQ 2015b]. No Currículo Lattes pode-se inserir dados gerais do pesquisador, produção bibliográfica, orientações, citações, participações em eventos científicos entre outros dados. No módulo Produção Bibliográfica, por exemplo, é possível a inserção de artigos publicados ou aceitos para publicação em periódicos indexados pelo ISSN [CNPQ 2015a].

A qualidade da produção bibliográfica dos Programas de Pós-Graduação é classificada pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) através de um conjunto de procedimentos denominado Qualis. O Qualis define estratos de qualificação a partir da análise da qualidade dos veículos onde a produção científica de pesquisadores brasileiros foi divulgada, ou seja, periódicos e eventos científicos. Esta análise é realizada em um processo anual de atualização, sendo os veículos enquadrados

nos estratos indicativos de qualidade A1, A2, B1, B2, B3, B4, B5 e C, do maior para o menor peso, para cada área do conhecimento [CAPES 2015].

Apesar da Plataforma Lattes possuir um módulo de inclusão de publicações e permitir a definição do veículo onde este foi publicado, não há qualquer tipo de conexão entre os sistemas Lattes e Qualis, ou seja, o processo de avaliação de uma publicação (que é feita através da avaliação do veículo onde este foi publicado) deve ser realizado de forma manual.

O Sistema de Integração Lattes Qualis (SILQ) surgiu no ano de 2015, desenvolvido como Trabalho de Conclusão de Curso dos alunos Felipe Nedel Mendes de Aguiar e Maria Eloísa Costa do curso de Ciência da Computação da Universidade Federal de Santa Catarina (UFSC), orientados pela Prof. Carina F. Dorneles, da mesma instituição. O objetivo do sistema é a classificação automática da produção científica do currículo Lattes do pesquisador, através do *matching* por similaridade de dados extraídos do Qualis, usando uma interface web amigável [de Aguiar and Costa 2015].

A primeira versão do sistema foi finalizada em 2015 e desde então encontra-se disponível de forma pública e gratuita através do endereço `http://silq.inf.ufsc.br/`. No entanto, esta primeira versão apresenta alguns pontos que merecem melhorias.

O SILQ possui em seu banco de dados de eventos e periódicos extraídos do Qualis apenas dados referentes ao triênio 2010-2012, já que, até metade de 2015, o Qualis realizava classificações de forma trienal. Em 2016, no entanto, após a finalização do projeto SILQ, o Qualis alterou seu modo de operação para classificações anuais, já disponibilizando dados referentes aos anos 2013 e 2014, que ainda não estavam incluídos nesta primeira versão da base de dados SILQ. O primeiro ponto de melhoria é fazer com que o SILQ permita a atualização automática da qualificação a cada novo ano de lançamento do Qualis. Como o formato no qual a CAPES disponibiliza o Qualis não é interoperável, o SILQ deve oferecer esta funcionalidade.

O segundo ponto a ser trabalhado é a falta de alguns recursos que poderiam facilitar o processo de avaliação e acompanhamento de currículos, como gráficos de classificações dentro Programas de Pós-Graduação. Uma característica importante também seria a capacidade de integração com outros sistemas, facilitando o reuso do serviço disponibilizado pelo SILQ em trabalhos futuros.

O terceiro ponto de melhoria seria a avaliação experimental do algoritmo de classificação do SILQ para o estabelecimento de uma medida do grau de precisão da ferramenta. A partir disto, outras técnicas podem ser propostas e comparadas com as anteriores a fim de provar se elas resultaram em ganho de precisão.

Como o *matching* entre os dados do Qualis e do Lattes de pesquisadores é feito através de funções de similaridade, o resultado pode ser um falso positivo. Portanto, um ponto importante a ser trabalhado é permitir ao usuário informar ao sistema sugestões de resultados. Esta técnica, conhecida por *feedback* de relevância, exige alterar o algoritmo de classificação a fim de treiná-lo com as informações providas pelo usuário e melhorar os resultados de pesquisas similares subsequentemente realizadas. Também se faz necessária a mensuração da taxa de acerto do sistema após a inclusão desta técnica e avaliar se ela beneficia a precisão do algoritmo de classificação.

A proposta deste trabalho, portanto, é a criação de uma segunda versão do SILQ, desenvolvida a partir do código existente da primeira, a fim de realizar as mudanças sugeridas acima, junto com a atualização da base de dados e arquitetura do sistema para a inclusão das novas qualificações Qualis em um ritmo anual.

A hipótese de pesquisa a ser avaliada a partir desta ideia, portanto, é avaliar se o *feedback* de usuários pode melhorar a taxa de acerto do SILQ, ou seja, aumentar o número de trabalhos corretamente avaliados pelo sistema.

1.1. Objetivos

O objetivo geral deste trabalho é analisar o impacto que o uso de *feedback* de relevância tem na precisão dos resultados de avaliações realizadas pelo SILQ, efetuado sobre uma nova arquitetura da ferramenta que inclui a criação de API de integração com outros sistemas e a atualização da base de dados conforme as novas classificações Qualis.

Os objetivos específicos são os seguintes:

- Reestruturação da arquitetura e banco de dados do SILQ a fim de suportar classificações de eventos e periódicos disponibilizados em um ritmo anual;
- Atualização do banco de dados do sistema com as últimas classificações disponibilizadas pelo Qualis (anos 2013 e 2014);
- Criação de uma API pública de disponibilização dos serviços do SILQ, via camada de aplicação REST para integração com outros sistemas;
- Alterações na interface do sistema incluindo migração de *framework* de interface, inclusão de controles de *feedback*, novos gráficos de acompanhamento de grupos de pesquisa e melhorias gerais de usabilidade;
- Propor novos algoritmos de avaliação baseados em similaridade textual e *feedback* de relevância e verificar se a taxa de acerto do sistema foi melhorada com tal ação.

1.2. Procedimentos Metodológicos

A primeira etapa do trabalho envolveu a migração tecnológica da camada de interface e desacoplamento entre *server* e *client side*, originando um serviço *RESTful* de integração. Para isto, foram utilizadas práticas ágeis de desenvolvimento de software incluindo desenvolvimento orientado a testes e eventuais modelagens lógicas de arquitetura de sistemas usando práticas da Engenharia de Software.

Concomitantemente, ocorreu a alteração da base de dados para suporte aos novos registros disponibilizados pelo Qualis. Certos aspectos do banco foram alterados para suportar tais registros novos. Os dados também passaram por um processo de *data-cleaning* manual antes de serem inseridos no banco, conforme detalhado na Seção 3.1.

A segunda etapa do trabalho envolveu o processo de construção do módulo de captação de *feedback* de relevância, envolvendo a modelagem lógica, criação da camada de serviço, API e alterações na página de resultados de avaliação do sistema com a inclusão dos controles que permitem aos usuários "julgar" os resultados retornados. Após esta etapa o algoritmo de classificação baseado em similaridade textual do SILQ foi estendido para levar em conta os *feedbacks* dos usuários. Deste processo surgiram dois algoritmos que passaram por uma avaliação experimental, realizada usando um conjunto de testes contendo trabalhos extraídos de currículos Lattes de pesquisadores e manualmente avaliados por um especialista, conforme relatado na Seção 4.3.

2. SILQ 2: Alterações estruturais

Como já mencionado, o presente trabalho é uma continuação dos esforços iniciados por [de Aguiar and Costa 2015] e que deu origem ao Sistema de Integração Lattes-Qualis (SILQ). Neste capítulo será apresentado o histórico do trabalho original, os motivos e justificativas que levaram ao desenvolvimento de um novo trabalho para sua continuação, comparações do que foi alterado nesta nova versão do sistema e por que estas alterações foram necessárias.

Para mais fácil compreensão e contextualização, durante o texto serão utilizados os termos "trabalho original", "SILQ 1" ou ainda "primeira versão do sistema" para referir-se ao trabalho de [de Aguiar and Costa 2015]. Os termos "SILQ 2" e "nova versão do sistema", em contrapartida, referem-se a este trabalho.

2.1. Histórico e Visão Geral do SILQ 1

Este trabalho é uma continuação de [de Aguiar and Costa 2015], um Trabalho de Conclusão de Curso de alunos do curso de Ciência da Computação da UFSC, orientados pela Professora Carina F. Dorneles. O objetivo desse trabalho de 2015 era a criação de um sistema que deveria ser capaz de qualificar produções científicas, nas categorias artigos e trabalhos apresentados em eventos, por busca por similaridade de dados com os dados extraídos do WebQualis [de Aguiar and Costa 2015, p. 26-27]. Este objetivo foi alcançado com a criação da primeira versão do Sistema de Integração Lattes-Qualis (SILQ), lançado no segundo semestre de 2015 e disponível no sítio `http://silq.inf.ufsc.br/`.

Apesar de estável e com sua função principal sendo desempenhada de forma satisfatória, o SILQ 1 deixou algumas lacunas e melhorias a serem desenvolvidas por trabalhos futuros. Segundo os próprios autores, "[...] o SILQ foi concebido para ser uma ferramenta de domínio público e vários projetos devem nascer a partir dele. A continuidade do projeto só tem a acrescentar ao mundo acadêmico [...]" [de Aguiar and Costa 2015, p. 79], o que motivou a criação do trabalho para a continuação da proposta original. O ponto de partida para a nova versão do SILQ, portanto, foi a Seção "Trabalhos futuros" do trabalho original.

2.2. Como o SILQ realiza o *matching* Qualis-Lattes

O *matching* da produção científica de um currículo Lattes com o Qualis é realizado pelo SILQ com base na similaridade textual entre o título do evento de cada trabalho presente no currículo e o título do evento Qualis[1]. O título do veículo onde o trabalho foi publicado, juntamente com seu ano e área de avaliação cadastrados no Lattes são dados como *query* para o sistema. O sistema busca sobre todo o conjunto de dados Qualis presentes na base de dados aquele com maior similaridade textual em relação à *query*. Se esta similaridade for maior do que um *threshold* pré-estabelecido, então o resultado é considerado um *match* válido e o estrato atribuído pelo Qualis a este evento é também atribuído ao trabalho do pesquisador.

Pode-se tomar o exemplo real de um trabalho qualquer extraído de um currículo Lattes cujo evento tenha sido especificado como *"2016 IEEE 7th Latin American Sympo-*

[1] O *matching* por similaridade é utilizado somente para eventos. Para periódicos, o *matching* é realizado através de comparação entre o ISSN informado no Lattes e nos registros Qualis.

sium on Circuits & Systems (LASCAS)". A Tabela 1 mostra os resultados retornados pelo SILQ junto com seus respectivos valores de similaridade textual.

Tabela 1. Resultados retornados pelo SILQ para a *query "2016 IEEE 7th Latin American Symposium on Circuits & Systems (LASCAS)"*

Evento	Estrato	Similaridade
IEEE Latin American Symposium on Circuits and Systems (LASCAS)	B5	0.87
IEEE International Symposium on Circuits and Systems (ISCAS)	A1	0.48
IEEE Latin American Robotics Symposium (LARS)	B4	0.45
IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)	B1	0.43
Symposium on Asynchronous Circuits and Systems (ASYNC)	A2	0.40

O primeiro resultado retornado é o escolhido, por possuir a maior similaridade em relação à *query*. Neste caso, o trabalho é avaliado com o conceito B5, já que este é conceito Qualis atribuído ao evento.

O algoritmo de avaliação da primeira versão do SILQ retorna os resultados da Tabela 1 ao ser configurado para utilizar um "nível de confiança" de 40%. Esse nível de confiança é o *threshold* utilizado pelo algoritmo de classificação. Qualquer resultado cujo nível de similaridade em relação à *query* seja inferior ao nível de confiança utilizado não é retornado.

O nível de confiança pode ser ajustado através das opções de avaliação, apresentadas quando o usuário requisita uma avaliação de currículo Lattes (Figura 1). Diminuir o nível de confiança (e em consequência o *threshold* do algoritmo de classificação) implica em obter mais resultados e classificar mais trabalhos, porém diminuir a precisão do algoritmo, já que resultados não relevantes serão retornados para as *queries* que não obtiveram bons resultados (resultados com nível de similaridade alto). A Seção 4.3 apresenta testes de validação do algoritmo que indicam o nível de precisão obtidos para cada valor de *threshold* utilizado, além de sugerir um nível de confiança ideal a ser utilizado para maximizar o número de trabalhos corretamente avaliados pelo sistema.

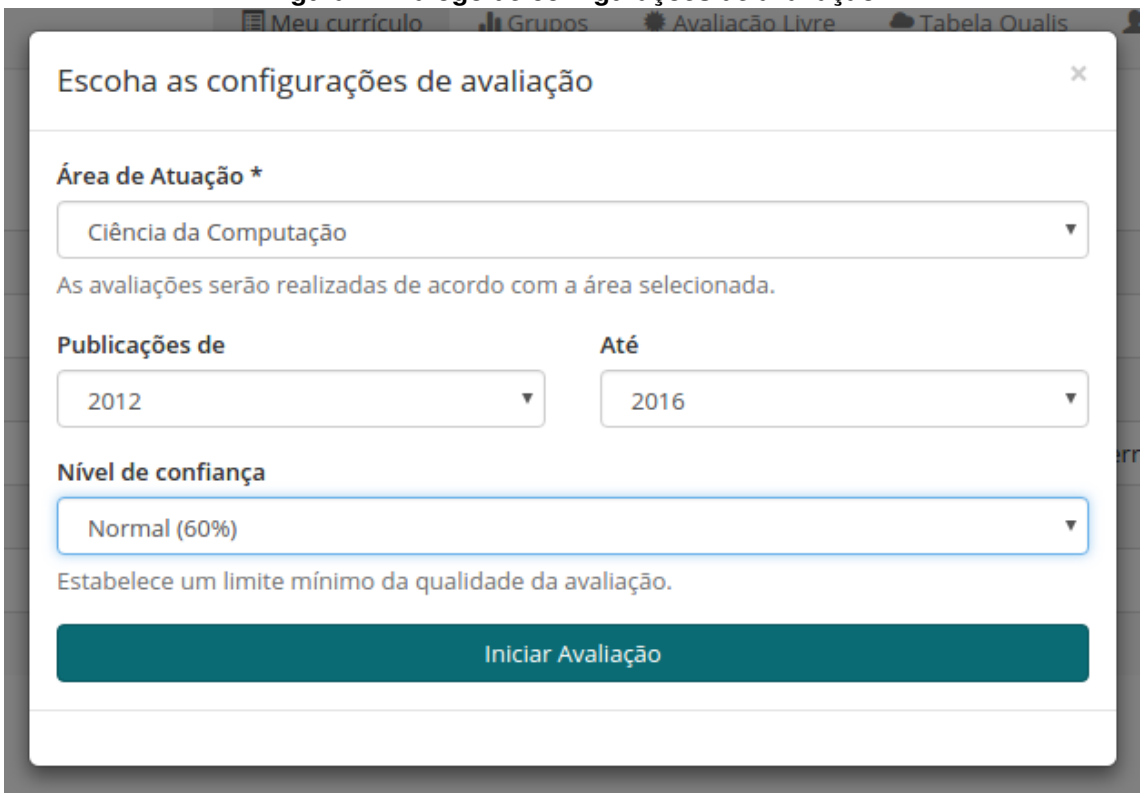
3. SILQ 2

3.1. Extração e inserção dos novos dados Qualis

Os dados Qualis de eventos e periódicos foram extraídos pelo trabalho original a partir de documentos em PDF e planilhas. Como relatado em [de Aguiar and Costa 2015, p. 50-52], os dados foram extraídos dos PDFs utilizando bibliotecas para manipulação desse formato e passaram por um processo de limpeza. A partir deste processo, aproximadamente 107 mil tuplas foram criadas representando os periódicos do triênio 2010-2012.

No primeiro semestre do ano de 2016, porém, a CAPES disponibilizou novos dados Qualis referentes ao ano de 2013 e 2014, alterando sua periodicidade de avaliação

Figura 1. Diálogo de configurações de avaliação



de trimestral para anual. Estes novos dados foram divulgados em formado CSV, junto com os antigos dados do triênio 2010-2012, mas desta vez separados por ano. Estes dados foram obtidos pela plataforma Sucupira (antigo WebQualis): `http://qualis.capes.gov.br/webqualis/`.

Como o conjunto de dados do SILQ 1 contava apenas com dados trimestrais, as tuplas originais não possuíam informação de ano (só podia-se deduzir que os dados faziam parte do triênio, mas não de qual ano). Desta forma, todas as planilhas divulgadas passaram por um novo processo de extração, incluindo os dados já inseridos no SILQ 1, mas agora considerando a nova informação de ano.

As cinco novas planilhas passaram por um processo de *data cleaning* manual:

1. Substituição de entidades HTML especiais por seu caractere correspondente. Ex.: "`&`" por "`&`";
2. Normalização do campo ISSN para o formato "9999-9999": alguns registros não possuíam o dígito separador ou omitiam os zeros à esquerda;
3. Correção de ISSNs errôneos: alguns registros possuíam o campo ISSN com dígitos faltando. Ex.: 0034-167 (Revista Brasileira de Enfermagem), cujo número correto seria 0034-**7**167. Estes casos foram tratados um a um e os números corretos identificados através de pesquisas na web.

As planilhas CSV foram então dadas como *input* à base de dados resultando na criação de 339.204 tuplas, referentes a cada um dos anos do período divulgado 2012-

2015. A Tabela 2 mostra o número de periódicos extraídos para cada ano de avaliação do Qualis.

Tabela 2. Número de periódicos extraídos dos dados Qualis

Ano	Nº de periódicos extraídos
2010	75.786
2011	66.171
2012	108.272
2013	44.437
2014	44.538
Total	**339.204**

Após esta etapa, verificou-se que todos os registros da base de dados antiga do SILQ 1 estavam contidos nesta nova versão. Desta forma, os dados do SILQ 1 foram mantidos, porém agora incluindo o ano de avaliação, além do nome do periódico, ISSN, estrato Qualis atribuído pela CAPES e área de avaliação.

A inclusão dos novos registros aumentou o número de publicações passíveis de serem avaliadas pelo sistema. Os dados Qualis de 2015 e 2016 ainda não foram divulgados no momento de escrita deste trabalho, mas podem ser incluídos na base de dados, assim que forem disponibilizados pela CAPES, passando por um processo semelhante ao descrito nesta Seção.

3.2. Criação do *Web Service*

A primeira versão do SILQ foi desenvolvida com base no *Play*[2] *Framework*, um *web application framework* escrito em Java e Scala que simplifica o processo de construção de uma aplicação web. Este mesmo framework provê APIs e bibliotecas para o desenvolvimento tanto do *back-end* quando do *front-end* de uma aplicação web.

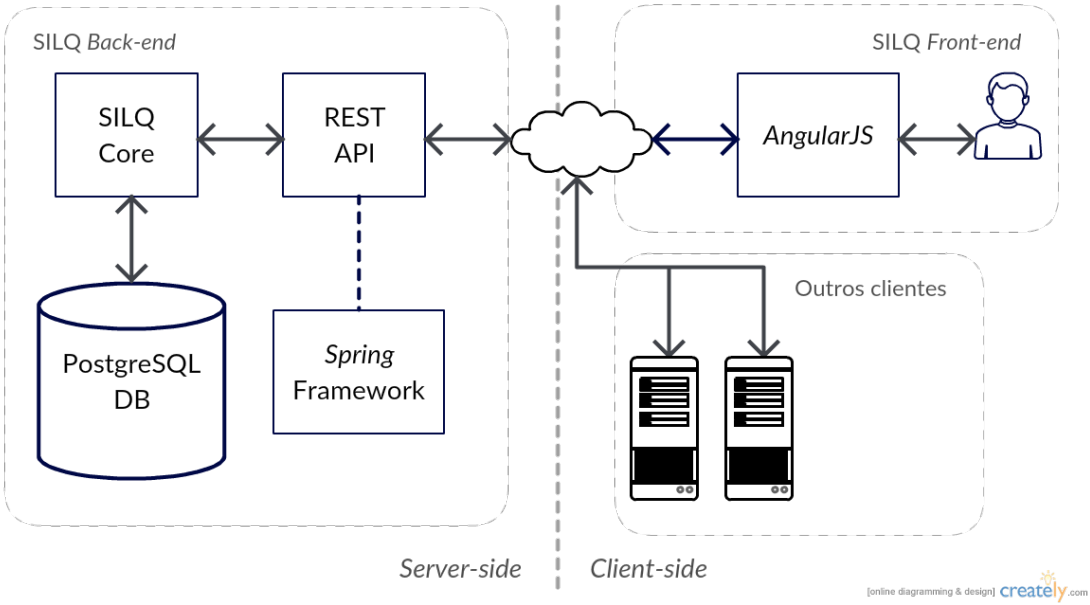
Para esta nova versão do SILQ, porém, um dos objetivos seria a criação de uma API de integração programática, tornando o SILQ não só um sistema para uso de usuários finais, mas também para integração com outras ferramentas, como um *web service*. Para tanto, uma alteração significativa na arquitetura da aplicação foi feita, separando o sistema em um *server-side* rodando Spring[3] e servindo seu conteúdo através de uma API REST; e um *client-side* remodelado utilizando AngularJS[4] e consumindo o serviço através de requisições HTTP.

Em um primeiro momento, o "núcleo" do SILQ, que inclui os algoritmos de avaliação e processamento dos currículos, não foi alterado, apenas a forma com que este serviço é disponibilizado e consumido. Desta forma, a linguagem utilizada no sistema, o Java, foi mantida. A escolha da alteração do framework back-end, de *Play!* para *Spring*, se deu ao fato da mudança no paradigma arquitetural da aplicação, com a introdução da camada REST. O Spring se trata de um framework Java *open-source* largamente utilizado pela comunidade que oferece suporte a vários aspectos de uma aplicação web, incluindo a construção de interfaces RESTful.

[2] `https://www.playframework.com/`
[3] `http://projects.spring.io/spring-framework/`
[4] `https://angularjs.org/`

A Figura 2 mostra um esquema da nova arquitetura do SILQ. A caixa “SILQ Core” representa a camada de serviço da aplicação, responsável pelo acesso ao banco de dados, avaliação de currículos e gerenciamento dos dados Qualis, cuja implementação foi realizada no trabalho passado (apesar de também sofrer algumas mudanças tecnológicas no trabalho atual).

Figura 2. Nova arquitetura do SILQ



3.3. Alterações no *front-end*

Na primeira versão do SILQ, as páginas HTML do sistema eram dinamicamente renderizadas no server-side pelo *Play!* Framework e servidas conforme as requisições dos clientes. Desta forma, além de executar toda a lógica de aplicação, o servidor também fica responsável pela lógica de apresentação do conteúdo.

Em arquiteturas REST, a lógica de apresentação de dados (ou visão), fica completamente separada do servidor, executando no *web browser* do cliente. O servidor só recebe requisições HTTP, as processa e retorna uma resposta em formato JSON. Esta resposta é recebida pelo cliente que então apresenta o conteúdo através da interface gráfica HTML renderizada pelo browser. Desta forma, a carga no servidor é diminuída, já que ele não está mais incumbido da tarefa de renderização das páginas HTML que serão servidas.

Utilizando esta ideia, no SILQ 2, a camada de visão foi totalmente reescrita em *Javascript*, utilizando o framework *AngularJS*. Trata-se de um framework *open-source* desenvolvido pela Google que auxilia na criação de interfaces web dinâmicas e integração com *webservices* via REST.

Desta forma, quando um cliente realiza a primeira requisição à URL do SILQ, o servidor serve uma página HTML (geralmente denominada de *index*) junto com os *assets* necessários para a construção de conteúdo dinâmico e estilização da página (imagens, arquivos CSS e Javascript). As requisições subsequentes, entretanto, utilizam somente requisições assíncronas ao servidor, utilizando Javascript: uma requisição conhecida por

Ajax (Asynchronous JavaScript and XML). Estas requisições utilizam a camada REST do servidor que retornará respostas em JSON e cujo conteúdo será processado no cliente e montará as páginas de forma dinâmica. Desta forma, parte da lógica da aplicação como um todo é transferida para o cliente, reduzindo a carga no servidor.

Figura 3. Página de resultados de avaliação do SILQ 1

Trabalhos

Título do trabalho	Ano de publicação	Nome do evento	Conceito
A Filtered-Page Ranking: An Approach for Previously Filtered HTML Documents Ranking	2016	International Conference on Internet and Web Applications and Services	B3 (1.0) ✔
Towards Automatic Document Classification by Exploiting only Knowledge Resources	2015	International Conference of the Chilean Computer Science Society	B3 (1.0) ✔
Implementação de um esquema de extração de dados tabulares da web	2015	XII Workshop de Trabalhos de Iniciação Científica (WTIC)	✖
Nazca: A Context-based Matching Method for Searching Heterogeneous Structures	2014	International Conference on Computer and Information Technology	B1 (1.0) ✔
Uso de Expressões Temporais em Busca na Web: Uma análise através das sugestões de consulta.	2014	Escola Regional de Banco de Dados	✖
SSUP - A URL-based method to entity-page discovery	2014	International Conference on Web Engineering	B1 (1.0) ✔
Descoberta de Domínio Conceitual de Páginas Web	2014	Workshop de Teses e Dissertações em Banco de Dados	✖
Proposta de um Framework para Visualização de Dados Agregados por Similaridade para Auxiliar Consultas durante a Navegação na Web	2013	Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais	B4 (1.0) ✔
Detectando similaridade entre diferentes representações de entidades usando Classificadores Bayesianos	2012	Escola Regional de Banco de Dados	✖
Pré-processamento de tabelas Web heterogêneas para execução do algoritmo de junção merge	2012	Escola Regional de Banco de Dados	✖
Usando informações do contexto para melhoria da precisão nas buscas por similaridade	2012	Escola Regional de Banco de Dados	✖

Figura 4. Página de resultados de avaliação do SILQ 2

Nome
Carina Friedrich Dorneles

Área do conhecimento
Ciência da Computação

Especialidade
Banco de Dados

Área utilizada na avaliação
Ciência da Computação

Período de avaliação
Todos

Totalizador Nível de confiança: Normal

Conceito	Qtde.
A1	4
A2	2
B1	5
B2	3
B3	6
B4	1
B5	4
C	1

Trabalhos 60 registros — Download .csv

2016 | A Filtered-Page Ranking: An Approach for Previously Filtered HTML Documents Ranking
International Conference on Internet and Web Applications and Services
B3 100% 2012 ICIW International Conference on Internet and Web Applications and Services
B5 75% 2012 ISWSA International Conference on Intelligent Semantic Web-Services and Applications
A1 70% 2012 MobiSys International Conference on Mobile Systems, Applications, and Services
A1 67% 2012 ICWS IEEE International Conference on Web Services
B3 63% 2012 ICNS International conference on Networking and Services
Ver menos resultados

2015 | Towards Automatic Document Classification by Exploiting only Knowledge Resources
International Conference of the Chilean Computer Science Society
B3 100% 2012 SCCC International Conference of the Chilean Computer Science Society
Ver mais resultados

2015 | Implementação de um esquema de extração de dados tabulares da web
XII Workshop de Trabalhos de Iniciação Científica (WTIC)
Nenhum conceito encontrado | Sugerir matching

2014 | Nazca: A Context-based Matching Method for Searching Heterogeneous Structures
International Conference on Computer and Information Technology
B1 100% 2012 CIT International Conference on Computer and Information Technology
Ver mais resultados

2014 | Uso de Expressões Temporais em Busca na Web: Uma análise através das sugestões de consulta.
Escola Regional de Banco de Dados
Nenhum conceito encontrado | Sugerir matching

Outra mudança significativa realizada no *front-end* da aplicação foi a atualização do *framework* CSS utilizado, inserida no processo de reescrita das páginas HTML. A

primeira versão do sistema utilizava a versão 2 do *framework open-source* Bootstrap[5], que foi atualizada para a versão 3 no SILQ 2. De forma paralela foram realizadas algumas mudanças na interface do sistema, principalmente na página de resultados de avaliação que passou a incluir controles permitindo ao usuário informar *feedback* de relevância sobre os resultados retornados. A Figura 3 mostra um exemplo da página de resultados de uma avaliação realizada pelo SILQ 1. A página foi remodelada para permitir a inclusão de controles de *feedback* e para apresentar múltiplos resultados candidatos para um mesmo artigo ou trabalho, conforme mostra a Figura 4.

3.4. Alterações no modelo lógico

O banco de dados relacional implementado em *PostgreSQL* também sofreu algumas modificações em relação ao trabalho anterior. A Figura 5 mostra o esquema lógico do banco de dados da versão atual do SILQ.

Algumas mudanças pequenas de nomenclatura foram realizadas. Uma mudança significativa, porém, foi a unificação de duas tabelas, *tb_dado_geral* e *tb_profissional*, para uma única tabela, *tb_curriculum_lattes*. A função destas duas tabelas antigas era guardar os currículos enviados de usuários e de pesquisadores de grupos, respectivamente. Nesta nova versão do SILQ, porém, em ambos os casos os currículos Lattes enviados pelo sistema são salvos na mesma tabela de currículos (*tb_curriculum_lattes*), inclusive tendo seus registros reutilizados em caso de, por exemplo, dois usuários enviarem o mesmo currículo. Neste caso, o sistema armazena somente uma vez o currículo e cria duas referências diferentes a ele, poupando-o do trabalho de duplicar os dados do currículo Lattes, cuja representação em XML contém em média algumas dezenas de *kilobytes*.

Outra alteração foi a adição da tabela *tb_feedback* para registro dos *feedbacks* de relevância informados pelos usuários, conforme relata a Seção 4.1.

3.5. Garantia da qualidade

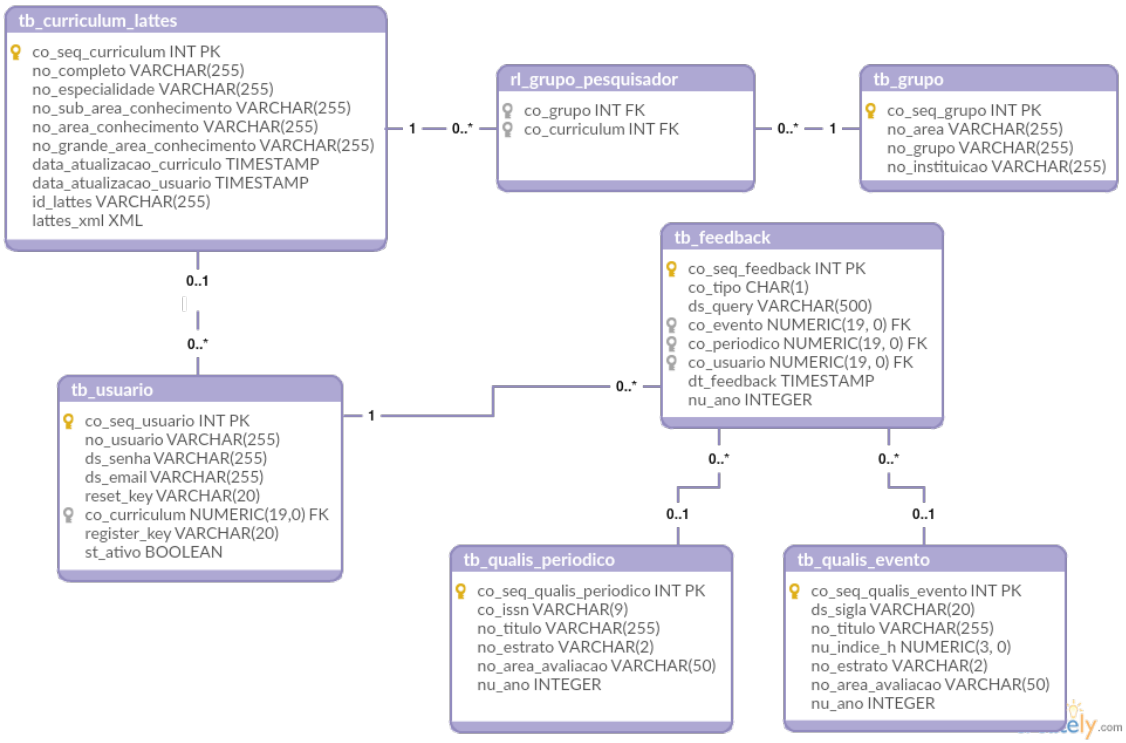
Para a garantia da qualidade desta nova versão do SILQ, foram introduzidas duas camadas de testes automatizados: a primeira, envolvendo *testes unitários* e *testes de integração* escritos em Java e com o objetivo de garantir a corretude do server-side da aplicação; e a segunda, *testes de sistema* escritos em Javascript e simulando casos de uso reais, garantindo a corretude do client-side e de sua integração com o servidor.

No SILQ, os testes unitários e de integração foram escritos em Java utilizando o framework de testes *jUnit*[6]. Eles testam cada função da camada de serviço da aplicação, por exemplo simulando o upload de um currículo qualquer e verificando se os dados extraídos e retornados são de fato aqueles contidos no currículo. Foram criados testes, inclusive, para as funcionalidades já existentes desde o SILQ 1, mas que ainda não eram cobertos com casos de teste automatizados, para assim aumentar a confiabilidade do sistema e garantir que mudanças futuras não ocasionem *bugs* nestes módulos antigos.

Este nível de teste, porém, não valida a interface de usuário e sua integração com a camada de serviço. Todo o client-side da aplicação ainda estaria "descoberto" de casos de testes. Para sanar este problema, foram incluídos no SILQ casos de teste de sistema escri-

[5] `http://getbootstrap.com/`
[6] `http://junit.org/`

Figura 5. Novo esquema do modelo lógico do SILQ



tos em Javascript e utilizando o framework de testes *Protractor*[7], criado especificamente para testes *end-to-end* de aplicações feitas com AngularJS. O Protractor simula as ações de um usuário real realizando cliques em botões, preenchendo formulários e navegando através de links da aplicação. Desta forma, são garantidos algum nível de corretude da interface do sistema e da integração entre o *client* e *server-side*.

4. Uso de *Feedback* de relevância

Como já mencionado, um dos itens de melhoria indicados por [de Aguiar and Costa 2015] após a realização da primeira versão do sistema seria "permitir que o usuário auxilie a ferramenta na qualificação". Em sistemas de IR, esta característica é denominada *Feedback de Relevância*.

A ideia por trás desta técnica é permitir que o usuário julgue resultados iniciais retornados pelo sistema, classificando-os como relevantes ou não para a *query* vigente. O sistema então é capaz de utilizar esta informação para melhorar seu algoritmo de classificação e retornar melhores resultados para novas *queries*.

O capítulo atual apresenta como esta técnica foi aplicada no SILQ, qual modelo lógico foi utilizado para o armazenamento dos *feedbacks* e qual técnica utilizada para sua obtenção. Por fim, são apresentadas medições realizadas no sistema que validam se a técnica foi relevante para o aumento da taxa de acerto do sistema.

[7] `http://www.protractortest.org/`

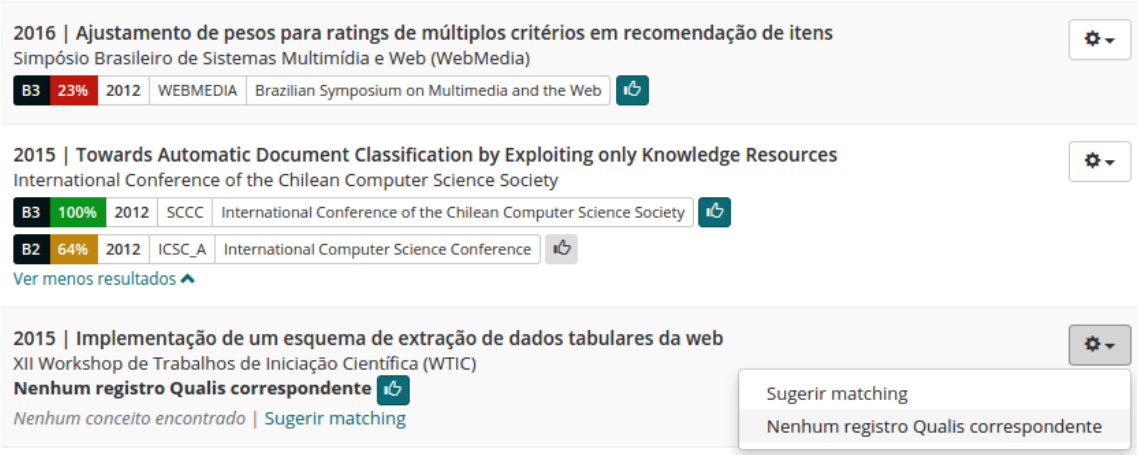
4.1. Obtenção de feedback

Em sistemas de IR que implementam *feedback* de relevância, tipicamente é permitido ao usuário julgar como relevante ou não relevante cada item retornado pelo sistema. Isso é natural pois o objetivo deste tipo de sistema é retornar a totalidade do conjunto de itens relevantes, cujo tamanho é variável dependendo da *query*. O SILQ, porém, é um caso específico em que existem somente 0 ou 1 item relevante para toda *query*. Isso acontece pois cada trabalho indicado no currículo Lattes de um pesquisador aparece qualificado no Qualis apenas uma única vez no ano em que o pesquisador o publicou. O objetivo do algoritmo de avaliação do SILQ é deduzir que registro Qualis é esse, caso exista na base de dados.

Para a implementação de *feedback* de relevância no SILQ, portanto, não é necessário o julgamento de cada item retornado pelo sistema em uma avaliação, mas apenas marcar *qual* dos itens da base de dados Qualis é um *match* correto para a *query* vigente. Desta forma, o usuário deve ser capaz de indicar o registro Qualis que deve ser considerado para cada trabalho avaliado. Também existe o caso especial em que não existe um registro Qualis correspondente ao trabalho, neste caso o usuário deve ser capaz de indicar que não existem *matches* corretos para o trabalho.

A Figura 6 mostra o exemplo de três trabalhos extraídos de um currículo Lattes e avaliados pelo SILQ que receberam *feedback* do usuário. O botão de "joinha" é utilizado para marcar o resultado que o usuário considera relevante para cada trabalho. Os botões grifados (com fundo azul) representam que o resultado já foi previamente marcado. Neste caso, o registro Qualis marcado como relevante é associado à *query* atual, que engloba o título, ano e área do trabalho avaliado, juntamente com o usuário que está realizando o julgamento. Também é possível marcar a opção "Nenhum registro Qualis correspondente" ou sugerir algum resultado não retornado previamente pelo sistema em "Sugerir matching" (mostrado no canto inferior direito da Figura exemplo).

Figura 6. Detalhe dos controles de feedback de relevância na página de avaliação de currículo Lattes



Os *feedbacks* de relevância dados pelos usuários são salvos na base de dados SILQ na tabela *tb_feedback* (Figura 5) para uso posterior pelo algoritmo de classificação, conforme descrito na Seção 4.2. Já que um trabalho qualquer pode ter no máximo um regis-

tro Qualis associado, é salvo somente um *feedback* por *query* por usuário. Desta forma, uma chave única é utilizada na tabela de *feedbacks*, dada pela dupla (título do trabalho, usuário).

4.2. Algoritmos de avaliação com feedback

Uma vez registrados os *feedbacks* dos usuários, passa-se a questionar de que forma utilizá-los para aumentar a taxa de acerto do sistema para futuras consultas. A Seção atual apresenta dois novos algoritmos que foram propostos e avaliados neste trabalho, e como foram implementados, enquanto a próxima Seção apresenta os resultados e comparações de exatidão dos mesmos.

4.2.1. Algoritmo `fb(t)`

Umas das abordagens mais simples que podem ser usadas neste caso é utilizar o resultado marcado pelo usuário sempre que uma *query* idêntica ao do *feedback* seja submetida ao sistema. O algoritmo `fb(1.0)`, portanto, foi desenvolvido com base nesta ideia. O valor "1.0" presente no nome do algoritmo apenas indica que o *feedback* é considerado em detrimento de qualquer outro resultado dado pelo sistema quando a *query* submetida for 100% similar (ou seja, idêntica) à *query* do *feedback*.

Pode-se dar o exemplo real de um nome de evento extraído de um currículo Lattes cadastrado pelo pesquisador como "Software Engineering Knowledge Engineering", no ano de 2009 e com área de avaliação Ciência da Computação. Ao avaliar tal trabalho, o sistema retorna a lista da Tabela 3.

Tabela 3. Resultados retornados pelo SILQ para a *query "Software Engineering Knowledge Engineering"*

#	Evento	Similaridade
1	Software Engineering and Data Engineering (SEDE)	0.53
2	International Conference on Software Engineering and Knowledge Engineering (SEKE)	0.49
3	Software Engineering and Applications (SEA_A)	0.45
...	...	...

Após analisar esta lista, o usuário submeteu um *feedback* ao sistema marcando o resultado #2 como o correto. Desta forma, utilizando o algoritmo `fb(1.0)`, toda *query* subsequente idêntica a "Software Engineering Knowledge Engineering" terá o resultado #2 retornado na primeira posição.

O `fb(1.0)` considera apenas *feedbacks* que sejam idênticos à *query* submetida, *queries* similares não são consideradas. O usuário do exemplo anterior possui um outro trabalho cadastrado em seu currículo Lattes cujo título de evento é "Software Engineering **and** Knowledge Engineering". Pode-se deduzir que o usuário quis se referir ao mesmo evento, porém o título não é idêntico ao exemplo anterior por causa do termo *"and"*.

Neste caso, o algoritmo `fb(1.0)` não é capaz de deduzir que os dois casos se referem ao mesmo evento, apesar da semelhança entre eles. Uma modificação que pode ser realizada no algoritmo é utilizar uma função de similaridade entre novas *queries* submetidas ao sistema com aquelas anteriormente submetidas e que possuem *feedback* do usuário. Se a similaridade entre a nova *query* e algum dos *feedbacks* for maior do que certo *threshold de feedback*, então é provável que a nova *query* se refira ao mesmo evento do *feedback* anteriormente fornecido.

O algoritmo `fb(t)` (para $0.0 \leq t \leq 1.0$) é uma generalização de `fb(1.0)` que considera *feedbacks* cuja similaridade textual em relação à *query* seja maior que o *treshold* t. Por exemplo, *fb(0.75)* irá considerar *feedbacks* cujo valor de similaridade textual em relação à *query* seja 0.75 ou superior. No exemplo anterior, ao submeter a nova *query* "Software Engineering and Knowledge Engineering" ao sistema, o algoritmo *fb(0.75)* calcula a similaridade entre ela e os *feedbacks* anteriores fornecidos pelo usuário e encontra o *feedback* da primeira *query* "Software Engineering Knowledge Engineering" por ser 88% similar à *query* atual. Neste caso, por ter uma similaridade maior do que o *threshold* de 0.75 estipulado, o algoritmo retorna o mesmo evento marcado no *feedback* para a *query* atual (o evento #2 da Tabela 3).

O algoritmo `fb(t)`, entretanto, leva a outros questionamentos, já que utiliza a mesma técnica de *data-matching* que foi proposta a melhorar. Qual o valor de t (*treshold de feedback*) ideal? Qual o algoritmo de similaridade textual ideal para este caso? A Seção 4.3.4 apresenta testes de validação do algoritmo `fb(t)` para diferentes valores de t.

O Algoritmo 1 é a representação em pseudocódigo de `fb(t)`. O parâmetro q representa a *query*, t é o valor de *threshold de feedback*, D é o conjunto de todos os documentos a serem pesquisados e F o conjunto de *feedbacks* fornecidos contendo as duplas (q_f, d), *query* do *feedback* e documento dado como *feedback*, respectivamente, tal que $d \in D$. A variável m é o registro provindo de *feedback* com maior probabilidade de ser um *match* correto para a *query*, caso exista. A saída R é uma lista 0-indexada contendo os resultados da consulta, ordenada por ordem decrescente da probabilidade do resultado ser um *match* correto para a *query*. A função `trigram_sim` calcula a similaridade textual entre duas *strings* utilizando o método *trigrams* e retornando um valor no intervalo $[0, 1]$. A função `trigram_rank` é o algoritmo de avaliação da primeira versão do SILQ, que cria o rank R de similaridade a partir da comparação entre q e cada um dos documentos de D. A função `insert_rank_top` insere um registro no topo do *rank*, removendo itens duplicados previamente inseridos.

4.2.2. Algoritmo `query_aliasing`

Uma adaptação de `fb(t)` que mostrou-se de mais fácil implementação e que não gera o questionamento de qual valor de t utilizar, foi considerar as *queries* de *feedbacks* anteriormente fornecidos pelo usuário, da mesma forma que o `fb(t)`, porém inseri-las no *rank* de resultados de novas *queries* submetidas com base em seus valores de similaridade textual em relação à nova *query*, junto com os resultados previamente selecionados. Assim, ao invés de escolhê-lo em detrimento dos demais, o evento marcado com *feedback* só é retornado se for mais bem ranqueado que os demais resultados.

Algoritmo 1: `fb(t)`

Input : q, t, D, F
Output: R

```
1  R ← trigram_rank(q, D)
2  s_m ← −1
3  for (q_f, d) ∈ F do
4  |   s ← trigram_sim(q, q_f)
5  |   if s ≥ t and s ≥ s_m then
6  |   |   m ← d
7  |   |   s_m ← s
8  |   end if
9  end for
10 if m then
11 |   insert_rank_top(R, m)
12 end if
13 return R
```

Considerando os mesmos exemplos dados na Seção anterior, em que o usuário submete a nova *query* "Software Engineering and Knowledge Engineering" ao sistema, o algoritmo de `query_aliasing` realiza comparação textual entre a nova *query* e as *queries* anteriores que possuam *feedback*, da mesma forma que o `fb(t)`, encontrando a *query* "Software Engineering Knowledge Engineering", com um valor de similaridade de 0.88. Ao contrário do $fb(t)$, o algoritmo de `query_aliasing` irá inserir o evento dado como *feedback* a esta *query* junto com a lista de resultados previamente encontrados apenas via similaridade textual, usando o valor de 0.88 para posicionamento no *ranking*. A Tabela 4 mostra o *ranking* retornado para este exemplo. Nota-se que o evento #2, marcado pelo usuário como correto, foi elevado no *ranking* por receber o novo valor de similaridade da comparação com o *feedback*.

Tabela 4. Resultados retornados pelo SILQ para a *query "Software Engineering and Knowledge Engineering"* utilizando `query_aliasing`

#	Evento	Similaridade
2	International Conference on Software Engineering and Knowledge Engineering (SEKE)	0.88
1	Software Engineering and Data Engineering (SEDE)	0.62
3	Software Engineering and Applications (SEA_A)	0.53
...	...	...

O valor de similaridade atribuído, porém, perde seu significado semântico pois não é mais a similaridade textual entre o título do evento do Lattes e do Qualis calculado através do *trigrams*, mas um valor adimensional usado apenas para ordenação relativa dentro do *ranking*.

Desta forma, ao processar uma *query* q qualquer, o sistema processa um *rank* de resultados primários com base no algoritmo *trigrams* inicial. O *rank* é ordenado do resultado mais similar à q ao menos similar. Após esta etapa, ele também compara q com cada uma das *queries* anteriormente submetidas pelo usuário e que possuem *feedback* de relevância utilizando o mesmo algoritmo de similaridade textual. O resultado mais similar é inserido no *ranking* de resultados. Assim, se q é idêntico a um *feedback* já submetido pelo usuário, o evento deste *feedback* será retornado e inserido no topo do *ranking* de resultados, por ser 100% similar à *query*. Outros resultados similares, porém não idênticos, serão inseridos no *ranking* conforme seu valor de similaridade e só serão escolhidos em detrimento de outros resultados primários se seus valores de similaridade forem superiores a eles.

É como se, ao dar um *feedback* de relevância qualquer, o usuário criasse um *alias* (um apelido) ao resultado que está sugerindo. Assim, o sistema deve avaliar novas *queries* não só comparando-as com o nome real dos documentos, mas também com os apelidos dados a eles pelo usuário. Por este motivo o algoritmo foi chamado de `query_aliasing`. A avaliação deste algoritmo foi realizada e comparada com os demais na Seção 4.3.4.

O Algoritmo 2 é a representação em pseudocódigo do método proposto. O significado semântico das variáveis é equivalente ao do Algoritmo 1. A saída deste algoritmo é o próprio *rank* R, possivelmente contendo novos resultados devido à comparação com os *feedbacks* de F. O método `insert_rank` da linha 4 insere o registro d na lista ordenada R com um valor de *rank* s, preservando a ordenação da lista, de forma que o elemento em $R[0]$ seja aquele com maior valor de *rank* e, assim, o registro com maior probabilidade de ser um *match* correto para a *query*. A função também elimina itens duplicados, preservando aquele com maior valor de *rank*.

Algoritmo 2: `query_aliasing`

Input : q, D, F
Output: R

```
1 R ← trigram_rank(q, D)
2 for (q_f, d) ∈ F do
3 |   s ← trigram_sim(q, q_f)
4 |   insert_rank(R, s, d)
5 end for
6 return R
```

4.3. Avaliação experimental

Esta Seção apresenta os procedimentos realizados para avaliar as alterações promovidas no algoritmo de avaliação da nova versão do SILQ e se elas contribuíram para o aumento da taxa de acerto do sistema.

4.3.1. Conjunto de testes

O conjunto de testes utilizado para a avaliação do sistema foi criado a partir dos currículos Lattes de 33 pesquisadores do programa de pós-graduação em Ciência da Computação da

Universidade Federal de Santa Catarina (UFSC).

Destes 33 currículos, 300 publicações foram selecionadas de forma aleatória e manualmente avaliadas: caso possuíssem um registro Qualis equivalente então a publicação juntamente com o Qualis associado eram salvos no conjunto de testes; caso não possuíssem registro Qualis equivalente, então eram marcados como tal e também adicionados ao conjunto de testes.

Nas avaliações descritas a seguir, foram dadas como *query* ao sistema cada uma das publicações da coleção de testes, porém sem expor os resultados manualmente avaliados. Cada resposta retornada pelo sistema foi comparada com a respectiva resposta manualmente selecionada. Em caso das respostas serem idênticas, então o sistema avaliou corretamente a publicação; em caso de não serem idênticas, avaliou incorretamente. O caso de não haver registro Qualis equivalente à publicação foi considerada uma resposta correta quando o sistema não retornou nenhum resultado, e uma resposta incorreta caso contrário.

4.3.2. Métricas utilizadas

Uma vez definido o conjunto de teste, é preciso definir as métricas utilizadas na avaliação. Através da comparação das métricas é possível concluir se houve melhora em certos aspectos do sistema. No caso do SILQ, deseja-se melhorar a taxa de acerto, ou seja, maximizar o número de trabalhos corretamente avaliados. As métricas clássicas de avaliação de sistemas IR discutidas na Seção **??** foram consideradas.

As métricas de precisão e revocação foram descartadas por não se encaixarem com a forma de avaliação do sistema, baseada em *rank*. Conforme já discutido, estas métricas não são indicadas para sistemas deste tipo. Medidas mais indicadas nesse caso são *Precision at k* (*P@k*) e *R-Precision*. O algoritmo de avaliação do SILQ, porém, considera apenas o primeiro registro Qualis retornado para realizar *match* com o trabalho sendo avaliado (apenas o mais similar). Neste caso, a avaliação usando estas duas métricas devem considerar apenas o primeiro resultado, ou seja, *P@1* e *R-Precision* com $|R| = 1$ (sendo R o número de registros relevantes para a *query*). Em ambos os casos, para cada *match* retornado pelo sistema, temos medidas com valor igual a 0, caso o sistema não tenha avaliado corretamente o trabalho, e 1 caso tenha avaliado corretamente. Têm-se, portanto, um simples valor *booleano* indicando se houve acerto ou não, para cada *query* submetida. Considerando todo o conjunto de testes, pode-se somar o número de acertos e dividir pelo tamanho do conjunto, resultado um valor que indica a *taxa de acerto* do sistema. Este valor também é conhecido como *exatidão*[8] e foi a medida base escolhida para a avaliação experimental do sistema.

Outra medida utilizada em um primeiro momento foi a Média de Rank Recíproco (MRR). Conforme discutido, ela é particularmente interessante para sistemas que produzem uma lista de resultados ordenados por probabilidade de corretude, e, ao contrário da exatidão, é capaz de modelar o quão bem o sistema classificou o resultado correto, mesmo quando ele não foi classificado em primeiro lugar.

[8]O termo utilizado na literatura é *accuracy*, cuja tradução usual é *precisão*. Optou-se pelo uso do termo *exatidão*, no entanto, para evitar confusões com a métrica de precisão.

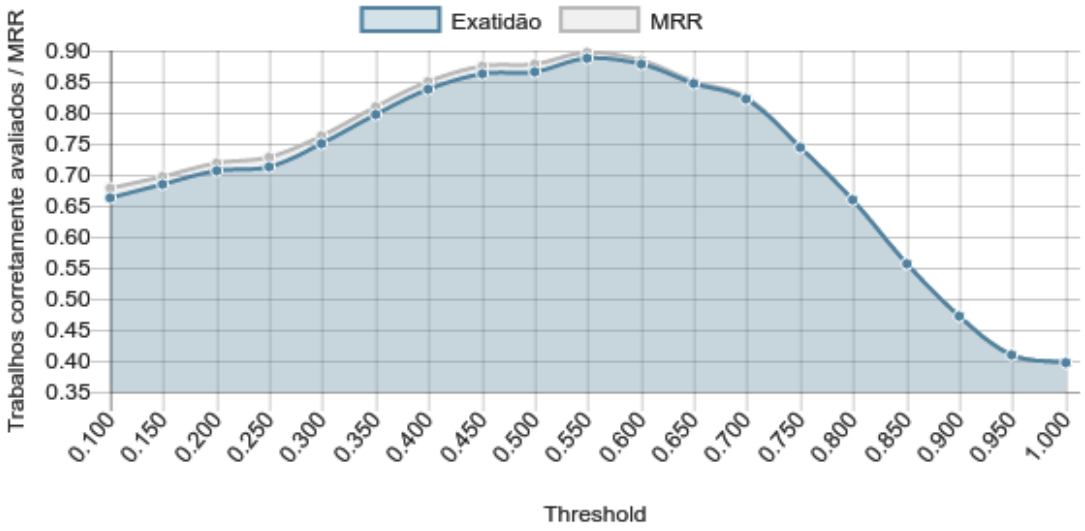
Por estas razões, as medidas de exatidão (ou taxa de acerto) e MRR foram escolhidas para as avaliações experimentais descritas nas próximas subseções.

4.3.3. Avaliação de *threshold* ideal

Um dos questionamentos levantados no início deste trabalho e que geralmente ocorre ao projetar sistemas de *data matching* baseados em similaridade, é o de qual *threshold* utilizar. Na primeira versão do sistema foi introduzido um controle de "nível de confiança" que permitia ao usuário controlar o *threshold* utilizado pelo algoritmo, conforme detalhado na Seção 2.2. O nível de confiança padrão, porém, foi fixado em 60% (equivalente ao *threshold* de valor 0.6). Este valor foi provavelmente escolhido de forma empírica pois observou-se que maximizava o número de resultados corretos, porém não foram realizados experimentos comprovando esta teoria.

Desta forma, para encontrar o valor de *threshold* ideal foi utilizado o conjunto de testes para avaliar o algoritmo inicial do SILQ 1, em um primeiro momento. O método utilizado foi o de avaliar via sistema cada uma das publicações do conjunto de testes e comparar com o resultado real, e repetir o processo variando o *threshold* a fim de observar as médias de exatidão e de rank recíproco (MRR). Os resultados foram agrupados no gráfico da Figura 7. A linha em azul claro representa a exatidão, ou seja, a taxa de trabalhos corretamente avaliados pelo sistema. A linha em cinza representa a média de rank recíproco (MRR), calculada conforme discutido na Seção **??**.

Figura 7. Taxa de trabalhos corretamente avaliados e Média de Rank Recíproco (MRR) para diferentes *thresholds*



O primeiro fenômeno que observamos ao avaliar o gráfico é o ponto de máximo por volta do valor 0.55 de *threshold*, que totaliza uma exatidão aproximada de 88%, e a tendência da exatidão baixar ao se afastar deste pico, para ambas as direções. Esse é um comportamento esperado pois valores de *threshold* baixos tendem a diminuir a exatidão do sistema por retornar resultados não relevantes para as *queries*, enquanto valores altos

tendem a diminuir a exatidão por deixar de retornar resultados relevantes. Este ponto máximo trata-se, portanto, do *threshold* ideal para o caso de testes em questão.

Outra característica observada é a tendência do valor de MRR acompanhar o da exatidão, sendo sempre igual ou apenas um pouco superior em magnitude. Isso acontece pela forma com que o MRR é calculado, atribuindo valor de $1/r$ a cada avaliação, sendo r a posição em que o resultado real foi avaliado pelo sistema. Se o resultado foi corretamente avaliado, portanto, o valor de $1/1 = 1$ é atribuído ao resultado, o mesmo valor que seria atribuído à exatidão, já que o conjunto de valores possíveis para esta métrica é $\{0, 1\}$ para cada resultado (0 representando um erro e 1 representando um acerto). A semelhança dos valores, portanto, indica que houveram poucos casos em que o algoritmo retornou o resultado real em posições inferiores à primeira no *rank* de avaliação. Esta característica do valor de MRR permaneceu constante nos demais testes realizados neste trabalho, portanto omitiu-se o valor de MRR nas demais avaliações.

4.3.4. Avaliação dos algoritmos

Os algoritmos descritos na Seção 4.2 foram avaliados utilizando o mesmo processo descrito na Seção anterior. O algoritmo *trigrams* trata-se do método inicial utilizado pelo SILQ 1 e cuja análise de *threshold* ideal foi realizada na Seção anterior. O algoritmo `fb(t)` foi testado variando t nos valores que obtiveram melhores resultados. Todas as análise foram realizadas com valor de *treshold* igual a 0.55. A Tabela 5 apresenta os resultados de cada teste.

Tabela 5. Comparação da exatidão dos diferentes algoritmos testados (utilizando *threshold* de 0.55)

Algoritmo	Exatidão
trigrams	88.667%
fb(1.00)	89.667%
fb(0.90)	90.667%
fb(0.80)	92.667%
fb(0.70)	92.667%
fb(0.60)	91.000%
query_aliasing	**93.333**%

A primeira tentativa de usar *feedback* de relevância na avaliação foi com o algoritmo *fb(1.00)*, que considera os resultados informados pelo usuário quando a *query* é idêntica a algum *feedback*. Houve uma melhora na taxa de acertos, porém de forma não tão significativa.

As variações que utilizam valores menores de t, porém, obtiveram melhores resultados, por serem capazes de identificar *queries* similares aos *feedbacks* já informados pelo usuário, mesmo quando este não julgou exatamente a *query* em questão. Um exemplo que demonstra este fato são os nomes de eventos *"IEEE International Symposium on Computer-Based Medical Systems"* e *"27th International Symposium on ComputerBased Medical Systems (CBMS)"*, extraídos de um mesmo currículo Lattes. É fácil notar que tratam-se do mesmo evento, porém o usuário informou a sigla e o número da edição no

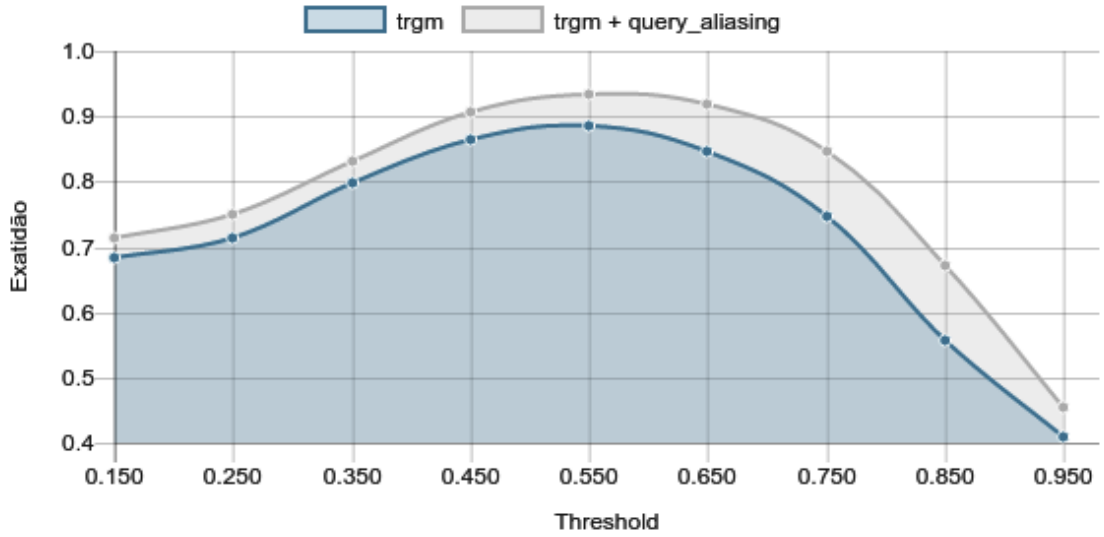
segundo, e nenhuma destas informações no primeiro (além de não utilizar o hífen em um dos casos). Caso o usuário tenha dado *feedback* para somente um dos casos, os algoritmos `fb(t)` com valores de t inferiores a 1.0 são capazes de utilizar o mesmo *feedback* para ambos, apesar das *queries* não serem idênticas.

Valores de t muito baixos, porém, deterioram rapidamente a taxa de acerto pois consideram *feedbacks* similares entre eventos que não tem relação. Desta forma, existe também um "valor ideal" de t, que gira em torno de 0.7 a 0.8 conforme os testes realizados.

O algoritmo `fb(t)`, porém, pode cometer "injustiças" pois considera os *feedbacks* como resultados corretos, independente dos resultados primários retornados, caso sejam superiores ao *threshold* t. O algoritmo de `query_aliasing` resolve este problema inserindo o *feedback* no *ranking* junto com os demais resultados. O melhor resultado (aquele mais similar à *query*) será utilizado, independente da técnica usada para obtê-lo. Nos testes realizados, o algoritmo de `query_aliasing` obteve a melhor taxa de acerto, com uma média de 93.3% de trabalhos corretamente avaliados.

Foi realizada uma última avaliação comparativa entre o algoritmo *trgm* inicial e o novo que utiliza `query_aliasing`. A Figura 8 mostra a taxa de acerto média para ambos os algoritmos variando o *threshold* utilizado. Nota-se que o algoritmo que utiliza *feedback* de relevância obteve melhores resultados, para qualquer *threshold* utilizado, aumentando em aproximadamente 6% a taxa de acerto. O *threshold* ideal, porém, se manteve constante em 0.55 pois é dependente da função de similaridade.

Figura 8. Comparação da taxa de acerto do algoritmo *trgm* e do *trgm + query_aliasing* para diferentes *thresholds*



5. Conclusões e trabalhos futuros

O SILQ é um esforço coletivo para a automatização e consequente melhoria na qualidade de gestão de grupos de pesquisa e principalmente de Programas de Pós-Graduação. A

atualização tecnológica e alimentação da base de dados da ferramenta são processos que devem ser constantemente realizados para mantê-la viável aos seus usuários.

Os objetivos iniciais de refatoração arquitetural com inclusão de melhorias de interface, novas funcionalidades e criação da camada de integração REST, foram alcançados. Tanto usuários como desenvolvedores de aplicações interessados no sistema se beneficiam dessa alteração. Também foram realizadas as inclusões de controles permitindo sugestões de resultados por parte dos usuários, e desenvolvidos dois algoritmos que utilizam tais informações para melhorar a taxa de acerto do sistema.

A hipótese de pesquisa levantada se mostrou verdadeira. De fato foi possível aumentar a precisão do sistema utilizando *feedback* de relevância dos usuários. Isso se mostrou verdade por meio dos experimentos realizados e relatados neste trabalho, e com o teste dos algoritmos propostos.

A avaliação de *threshold* mostrou que o valor ideal para o SILQ é de aproximadamente 0.55, muito próximo do "nível de confiança normal" estabelecido no trabalho anterior de [de Aguiar and Costa 2015], e que a taxa de acerto para este caso é de 88%.

Os algoritmos `fb(t)` e `query_aliasing`, baseados em *feedback* de relevância e similaridade textual, foram propostos e avaliados experimentalmente. Ambos resultaram em melhoria na taxa de acerto do sistema. O algoritmo `query_aliasing` foi preferido por resultar em uma melhor precisão e ser de mais fácil implementação, e está atualmente em uso na versão 2.3 do sistema. Através da alteração do *threshold* ideal e da inclusão do algoritmo baseado em *feedback* de relevância, a taxa de acerto média do sistema aumentou de 87% para 93.3%.

Esses resultados são relevantes não só por mostrarem que a taxa de acerto média do sistema aumentou, mas por estabelecerem uma medida base para trabalhos futuros. Os algoritmos utilizados neste trabalho são estratégias simples de uso de *feedback* de relevância construídas sobre a mesma função de similaridade textual do SILQ 1, o método *trigrams*, porém outros algoritmos e estratégias diferentes podem ser testados. Em especial, ambos algoritmos propostos consideram apenas o conjunto de *feedbacks* de um usuário específico, de forma a evitar a necessidade do tratamento de *feedbacks* conflitantes ou divergentes de diferentes usuários.

Como trabalhos futuros pode-se sugerir os seguintes itens:

- Propor e analisar outros métodos de similaridade textual ou estratégias diferentes que possam melhorar ainda mais a precisão do SILQ. Alguns exemplos seriam o uso de *machine learning* para treino do algoritmo de classificação, uma estratégia conhecida por *learning to rank*; e o algoritmo de *Rocchio*, muito utilizada em sistemas que implementam *feedback* de relevância e baseado no modelo de espaço vetorial;
- Considerar *feedbacks* provindos de terceiros, ou seja, levar em conta *feedbacks* informados por outros usuários, mesmo que o atual não tenha informado *feedback* a respeito de uma *query* específica;
- Considerar nomes de eventos traduzidos. Comumente são utilizados nomes de periódicos ou eventos estrangeiros em inglês no currículo Lattes, porém em português no Qualis, ou vice-versa. Esses casos dificilmente são avaliados corretamente pelo sistema, já que o algoritmo atual é baseado em comparação por simi-

laridade textual;

- Levantamento da produção por veículo de publicação. Considerando todos os currículos cadastrados no SILQ, seria possível levantar o número de publicações em cada periódico e evento cadastrado no Qualis;
- Gerar os valores de $I_{restrito}$ e I_{geral} automaticamente. Tratam-se de índices definidos pela CAPES e utilizados na avaliação de Programas de Pós-Graduação;
- Considerar, para fins de ponderação dos valores de estrato Qualis, os pesos considerados pela avaliação de Programas de Pós-Graduação realizada pela CAPES.

As mudanças relatadas neste trabalho já se encontram disponíveis na página oficial do SILQ: `http://silq.inf.ufsc.br/`, na versão 2.3 no momento de escrita deste trabalho, porém em constante evolução.

Referências

CAPES (2015). Classificação da produção intelectual. `http://www.capes.gov.br/avaliacao/instrumentos-de-apoio/classificacao-da-producao-intelectual`. Acesso em 17/11/2015.

CNPQ (2015a). Módulo produção bibliográfica. `http://ajuda.cnpq.br/index.php/Mdulo\_Produo\_Bibliogrfica`. Acesso em 17/11/2015.

CNPQ (2015b). Sobre a plataforma lattes. `http://www.cnpq.br/web/portal-lattes/sobre-a-plataforma`. Acesso em 17/11/2015.

de Aguiar, F. N. M. and Costa, M. E. (2015). *SILQ - Sistema de Integração Lattes Qualis*. Florianópolis: Universidade Federal de Santa Catarina, Biblioteca Universitária.