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

LEONARDO EICH

ACERPI-Link: Uma Alternativa de Resolução de Entidades em Portarias Institucionais Utilizando Linkagem de Registros

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

Orientador: Prof[a]. Dr[a]. Renata Galante
Co-orientador: Christian Schmitz

Porto Alegre
2021

AGRADECIMENTOS

Gostaria de começar agradecendo à minha mãe, Janea Gleci Soares da Silva, por ter feito do meu bem-estar uma das suas prioridades na sua vida. Agradeço ao meu pai, Julio Eich, por sempre me incentivar a estudar e me dedicar a alcançar meus objetivos. Agradeço também à minha irmã, Giulia Eich, pelo companheirismo ao longo de toda essa jornada. É com certeza um privilégio contar com essa estrutura que me apoia e permite que eu trabalhe com tranquilidade em busca dos meus objetivos.

Quero registrar também meu amor e gratidão à minha namorada, Tatielle Andrade, pelo amor e carinho proporcionados ao longo de tantos anos, pelo entendimento e motivação em momentos mais difíceis e por tantos momentos bons compartilhados.

Gostaria também de agradecer aos meus gatos, Barney e Maju, por estarem ao meu lado, e especialmente à minha cachorrinha Dilma, grande companheira da minha vida e que o amor que tem por mim ultrapassa qualquer barreira que limite o entendimento humano. Fica também a lembrança de outros cães e gatos que passaram em minha vida e que nenhum dia apagará as memória do tempo que passamos juntos.

Agradeço à todos os companheiros de curso que de alguma forma se fizeram presentes nessa jornada. Em especial aos amigos Felipe Comerlato, Amanda Goveia, Lucas Flores, Victor Santos e Grégori Barros. Se eu consegui chegar a esta etapa com certeza foi graças ao companheirismo de vocês em algum ponto. Fica registrado também meu abraço aos amigos de fora da faculdade que foram também imprescindíveis em diversos momentos que, indiretamente, me ajudaram também a finalizar essa etapa da minha vida. Em especial ao meu amigo Renan Fraga, meu melhor amigo.

Por fim, mas não menos importante, agradeço à grande maioria de professores do Instituto de Informática e da Universidade Federal do Rio Grande do Sul como um todo pela paixão por ensinar, apesar de dificuldades que todos sabemos da existência. Agradeço especialmente aos professores Marcelo Walter, pelas palavras e carinho no início do curso, a professora Renata de Matos Galante, orientadora deste trabalho e com uma empatia enorme demonstrada para com os alunos, bem como o professor Edimar Manica, co-orientador deste trabalho. Agradeço também ao colega de curso Christian Schmitz, pelo empenho despendido junto dos orientadores deste trabalho para que ele fosse desenvolvido da melhor forma possível.

Muito obrigado.

RESUMO

Portarias são documentos utilizados no âmbito de instituições federais para publicar informações acerca dos servidores que ali trabalham. Esses documentos são disponíveis ao público de cada instituição em repositórios que na sua grande maioria não permite buscas por parte dos usuários sobre o conteúdo dos documentos. A abordagem ACERPI-Link visa estender a abordagem ACERPI (**Abordagem** para **Coleta** de documentos, **Extração** de informação e **Resolução** de entidades em **Portarias Institucionais**), focando na etapa de Resolução de entidades, tarefa que consiste em identificar quais entidades citadas nas Portarias se referem à mesma entidade no mundo real através da linkagem de registros. Experimentos realizados utilizando funções de comparação de *strings* sobre bases de dados reais demonstraram 99,16% de precisão pela abordagem proposta, 86,29% de revocação e 92,28% de desempenho na medida F1.

Palavras-chave: Extração de informação. linkagem de registros. medidas de comparação de *strings*.

ACERPI-Link: An Alternative to Entity Resolution in federal institutions' documents from Brazil using record linkage

ABSTRACT

Portarias are documents used in federal institutions to issue information about the public employees working there. These documents are available to the public of each institution in repositories that for the most part do not allow users to search the contents of the documents. The ACERPI-Link approach aims to extend the original ACERPI approach, focusing on the Entity Resolution step, a task that consists, in the context of this proposal, of identifying which entities cited in the Portarias refer to the same entity in the real world. Experiments performed using string comparison methods on real databases demonstrated 99.16% accuracy by the proposed approach, 86.29% revocation, and 92.28% performance on the F1 measure.

Keywords: entity resolution, record linkage, string distance metrics.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface* - Interface de Programação de Aplicativos
CAPTCHA	*Completely Automated Public Turing test to tell Computers and Humans Apart* - Teste de Turing Público Completamente Automatizado para Diferenciar Computadores de Humanos
JSON	*JavaScript Object Notation* - Notação de Objeto Javascript
ER	*Entity Resolution* - Resolução de Entidades
HTML	*Hypertext Markup Language* - Linguagem de Marcação de Hipertexto
IA	Inteligência Artificial
NER	*Named Entity Recognition* - Reconhecimento de Entidades Nomeadas
PDF	*Portable Document Format* - Formato Portátil de Documento
PNL	Processamento de Linguagem Natural
SGDB	Sistema de Gerenciamento de Banco de Dados
URL	*Uniform Resource Locator* - Localizador Uniforme de Recursos
XML	*Extensible Markup Language* - Linguagem de Marcação Extensível

SUMÁRIO

1 INTRODUÇÃO

Portarias são documentos utilizados por instituições públicas com a finalidade de disseminar instruções acerca da aplicação de leis ou regulamentos, nomeações, demissões, recomendações de caráter geral e afins. Instituições federais fazem o uso de portarias para disseminar informações a respeito dos servidores que ali trabalham, fazendo uso delas para comunicar afastamentos, substituições, entre outros. Devido ao potencial interesse público sobre esses documentos, a Lei nº 12.527 (BRASIL, 2011) formaliza a divulgação dessas informações.

A abordagem ACERPI (**Abordagem** para **Coleta** de documentos, **Extração** de informação e **Resolução** de entidades em **Portarias Institucionais**) (SCHMITZ et al., 2021) se propôs a facilitar o acesso dessas Portarias no contexto de Instituições Federais pois, mesmo esses documentos sendo públicos, muitas vezes o acesso é dificultado por estarem, por exemplo, espalhados em repositórios individuais de cada instituição ou até mesmo de diferentes *Campi* de uma mesma instituição. Ainda, esses documentos estão disponíveis apenas no formato PDF em repositórios que, na maioria das vezes, oferecem pouco ou nenhum filtro para a realização de pesquisas simples, como a busca utilizando o nome de um servidor específico. A ACERPI tem como objetivo ser uma abordagem que vá desde a descoberta de arquivos, faça a obtenção e consiga armazenar a informação contida nos documentos em um banco de dados, permitindo ao usuário realizar diversos tipos de consulta sobre o conteúdo dessas Portarias como, por exemplo, o nome dos servidores mencionados, suas matrículas identificadoras (SIAPE) ou a data de publicação de uma Portaria.

Uma das etapas envolvidas na abordagem ACERPI é a que envolve resolução de entidades (*Entity Resolution, ER*, em Inglês). Essa tarefa consiste, essencialmente, em identificar as entidades nomeadas (extraídas na etapa anterior) que se referem à mesma entidade do mundo real. No contexto deste trabalho essas instâncias são, majoritariamente, servidores públicos como professores e técnicos administrativos, podendo ainda se tratar, em menores casos, de estudantes ou empresas. Na abordagem ACERPI, essa tarefa é realizada por meio de deduplicação, um processo que tem como entrada uma coleção de entidades e produz como saída o conjunto de pares de todas as entidades correspondentes.

O objetivo principal deste trabalho é propor uma alternativa à esta etapa, a partir uma abordagem que utiliza a linkagem de registros que consiste em mapear as entidades

nomeadas à entradas de um arquivo de autoridades (*authority file*). O resultado final é uma coleção de registros enriquecidos de uma conexão com uma única entrada que representa a autoridade presente em um dado registro.

O arquivo de autoridades é criado através do uso de uma API de serviços do Portal da Transparência do Governo Federal (Portal da Transparência do Governo Federal (2021)), para a coleta dos dados dos servidores e criação do *authority file*. Diversos experimentos envolvendo métodos de comparação de *strings* e demais dados das entidades, onde os resultados utilizando dados reais da Universidade Federal do Rio Grande do Sul (UFRGS) e do Portal da Transparência do Governo Federal, apresentaram uma eficácia na abordagem de mais de 92% na Medida F1, 99% de precisão e 86% de revocação.

O restante do texto está organizado da seguinte forma: o Capítulo 2 apresenta as técnicas e ferramentas já estabelecidas que são utilizadas ao longo do desenvolvimento deste trabalho. O Capítulo 3 descreve brevemente propostas similares à deste trabalho com foco em outras áreas de aplicação. O Capítulo 4 apresenta de forma resumida a abordagem ACERPI e as etapas envolvidas nessa proposta. O Capítulo 5 aprofunda as etapas introduzidas nesta proposta e como este trabalho visa se diferenciar da abordagem ACERPI na etapa de resolução de entidades a partir de diversos experimentos. O Capítulo 6 detalha os experimentos e casos de falha na tarefa de comparação de *strings* para a resolução de entidades. Por fim, o Capítulo 7 avalia os resultados e contribuições pelo trabalho realizado e sugere possibilidades de trabalhos para o futuro.

2 TÉCNICAS E TECNOLOGIAS UTILIZADAS

O objetivo deste Capítulo é apresentar as tecnologias que foram utilizadas ao longo do desenvolvimento do trabalho e explicar o porquê cada uma foi escolhida.

2.1 *Web Scraping*

Web Scraping é o processo que consiste em recuperar dados de forma estruturada de um *website* de maneira automatizada (ScrapingHub, 2020). Essa coleta pode ser feita tanto de maneira manual quanto automática.

Destacam-se como as principais técnicas de *Web Scraping*:

- Manual - cabe ao usuário navegar entre as páginas e manualmente copiar e colar os seus conteúdos;
- Caminhamento da estrutura de arquivos da página e *download* automatizado - consiste na análise da estrutura das páginas de interesse através de pequenos trechos de código e posterior coleta do conteúdo de maneira automatizada;
- Inferência de um padrão de navegação (Palmieri Lage et al., 2004) - consiste em navegar em endereços relevantes inferidos através de padrões e posterior *download* automatizado do conteúdo completo das páginas.

Neste trabalho, apenas a última abordagem é utilizada devido ao volume de arquivos a serem recuperados e repetição das estruturas dos dados acessados.

2.2 *NoSQL: Bancos de Dados Orientados a Documentos*

Segundo AWS (2020), bancos de dados NoSQL (*Not only SQL* - Não apenas SQL) seguem um modelo não relacional e se caracterizam por sua grande flexibilidade e desempenho em larga escala. MongoDB (2020b) destaca que existem quatro principais tipos de banco de dados NoSQL:

- Orientados a documentos - armazenam dados em documentos similares aos objetos JSON. Cada documento contém pares de campos e valores.
- Chave-valor - são um tipo mais simples de base de dados onde cada item contém chaves e valores.

- Orientados a colunas - armazena dados em tabelas, linhas e colunas dinâmicas.
- Orientados a grafos - armazenam dados em nós e arestas. Os nós normalmente armazenam informações sobre pessoas, lugares e coisas, enquanto as arestas armazenam informações sobre as relações entre os nós.

No contexto de banco de dados, uma estrutura orientada a Documentos é similar a objetos JSON, onde cada documento é composto de pares chave e valor, os quais podem ser de tipos como *strings*, valores númericos, *arrays* e também outros documentos. Por sua vasta documentação e por se tratar de um banco de dados orientado a documentos, o banco de dados MongoDB (MONGODB, 2020a) foi uma escolha lógica para o desenvolvimento deste trabalho. Além disso, a abordagem ACERPI, estendida neste trabalho, adotou o banco de dados de MongoDB. A Listagem 2.1 exemplifica como alguns registros utilizados neste trabalho são armazenados no banco de dados:

Listagem 2.1 – Exemplo de documento do MongoDB

```
1 {
2     "id": 4630,
3     "name": "RENATA DE MATOS GALANTE",
4     "siape": ["1488770"],
5     "document": {
6         "name": "50216"
7     }
8 }
```

2.3 Funções de Similaridade Entre *Strings* Para Comparação de Nomes

Funções de distância mapeiam duas *strings* $s1$ e $s2$ para um número real r, onde um valor menor de r indica maior semelhança entre $s1$ e $s2$. As funções de semelhança são análogas, exceto que valores maiores indicam maior semelhança. A tarefa de comparação de nomes já foi explorada por diversas comunidades, incluindo estatísticas, banco de dados e inteligência artificial, onde cada uma formulou o problema de uma maneira diferente e propôs diferentes técnicas para a solução do problema (COHEN; RAVIKUMAR; FIENBERG, 2003).

Nesta seção, são apresentadas as funções de similaridade utilizadas no desenvol-

vimento deste trabalho e os principais conceitos por trás de cada uma delas, além de justificar os motivos que levaram a escolha dessas funções. Vale ressaltar que foram utilizadas as bibliotecas Py_stringmatching (anhaidgroup.ai, 2021) e Strsimpy (ZhouYang Luo, 2021), ambas de código fonte aberto focadas na implementação de funções de comparação de *strings* para a linguagem Python. Essas duas bibliotecas permitiram ganhar tempo de desenvolvimento, visto que não foi necessário implementar a partir do zero os algoritmos das funções de similaridade e, por serem utilizadas por uma grande comunidade, também garantem confiança nos resultados apresentados.

As quatro principais abordagens na tarefa de comparação de *strings* são (COHEN; RAVIKUMAR; FIENBERG, 2003):

- *Edit-distance like functions* - são uma maneira de quantificar o quão dissimilar duas *strings* são a partir do número mínimo de operações necessárias para transformar uma *string* na outra;
- *Token-based distance functions* - os algoritmos desta abordagem esperam conjuntos de *tokens* em vez de *strings* completas, onde a similaridade aumenta conforme aumenta o número de *tokens* em comum entre os dois conjuntos;
- *Hybrid distance functions* - como o nome sugere, os algoritmos que usam essa abordagem utilizam soluções híbridas, combinando, por exemplo, métodos que utilizam *tokens* com métodos que utilizam *strings* inteiras;
- *Blocking* ou *pruning methods* - para diminuir o gasto computacional, esses métodos utilizam ligação probabilística de registros, agrupando os registros através de variáveis conhecidas *a priori* para pares que possuem um *match*.

No contexto desse trabalho, foram utilizadas funções que se baseiam nas três primeiras abordagens. Cada uma dessas funções será introduzida e terá seu funcionamento explicado a seguir.

2.3.1 Jaro

A função de similaridade Jaro (JARO, 1989) é uma função que usa o conceito de *edit distance* e estabelece que, dadas duas *strings* s1 e s2, sua distância Jaro(s1,s2) é definida como:

$$Jaro(s1, s2) = \frac{1}{3} \times \left(\frac{|\theta|}{|s1|} + \frac{|\theta|}{|s2|} + \frac{|\theta| - 0.5t}{|\theta|} \right) \tag{2.1}$$

onde:

- $|\theta|$: número de correlações entre caracteres - um caractere c é parte de θ caso $c \in s1$ em uma posição i, $c \in s2$ em uma posição j e $|i - j| \leq 0.5 \times max(|s1|, |s2|) - 1$
- $|s1|, |s2|$: comprimento das *strings* s1 e s2, respectivamente
- t: número de transposições - uma transposição existe se, enquanto iterando sobre s1 e s2, o i-ésimo caractere de s1 é diferente do i-ésimo caractere de s2.

O exemplo a seguir mostra o valor de similaridade definido pela função Jaro para duas *strings* s1 e s2:

- sejam duas *strings* s1 = Spears, s2 = Spaers
- Todos os caracteres pertencem ao conjunto θ
  - para *S, p, r* e *s*, todas as posições são iguais
  - para *e* e *a* as posiçõs variam em 1, que é o limite inferior para transposições (computada como $0.5 * 6 - 1$)
  - portanto, $\theta = \{S, p, e, a, r, s\}$
- quando iterando sobre s1 e s2, podemos observar:
  - primeiro caractere em s1 (S) = Primeiro caractere em s2.
  - segundo caractere em s1 (p) = Segundo caractere em s2.
  - no terceiro caractere de s1 temos que (e) $\neq$ (a), o terceiro caractere de s2.
  - uma segunda transposição ocorre no quarto caractere de s1, (a) $\neq$ (e).
  - demais posições são idênticas.
  - portanto, $t = 2$.
- Jaro(s1,s2) = $\frac{1}{3} \times \left(\frac{6}{6} + \frac{6}{6} + \frac{6-0.5\times2}{6}\right) = 0.944$

A função tem uma complexidade O(m x n) e serve principalmente em situações em que o objetivo é detectar erros de ortografia e também para o casamento de metadados.

2.3.2 Jaro-Winkler

A função Jaro-Winkler (WINKLER, 1990) é um método que usa o conceito de *edit distance* como base e é também uma extensão da distância de Jaro. Esse método bonifica *strings* que tenham um prefixo em comum. A fórmula que define a distância

entre duas *strings* s1 e s2 é:

$$JaroWinkler(s1, s2) = Jaro(s1, s2) + |\rho| \times f \times (1 - Jaro(s1, s2)) \tag{2.2}$$

onde:

- $|\rho|$: é o tamanho do prefixo comum entre s1 e s2, limitado a 4 caracteres;
- f: é uma constante definida pelo usuário como um fator de correção que ajusta para cima a similaridade computada pela função Jaro, definida aqui como Jaro(s1,s2). Esse valor não deve ultrapassar 0.25, ou seja, $1/4$ do valor, sendo 4 o tamanho máximo de prefixo compartilhado entre s1 e s2, pois sem isso a função pode ultrapassar 1 como valor retornado. O valor definido em Winkler (1990) é $f = 0.1$;

O exemplo a seguir demonstra o cálculo da função Jaro-Winkler para duas strings s1 e s2:

- sejam duas *strings* s1 = Britney, s2 = Britney Spears
- Jaro(s1,s2) = $1/3 * (7/7 + 7/14 + 1) = 0.83$.
- para um valor de $f = 0.1$ temos:

$$JaroWinkler(s1, s2) = 0.83 + 7 * 0.1 * (1 - 0.83) = 0.898$$

Segundo Cohen, Ravikumar e Fienberg (2003), assim como a função Jaro, a função Jaro-Winkler tem um desempenho melhor na tarefa de comparação de *strings* menores, como nomes e sobrenomes.

2.3.3 *Cosine Similarity*

A função *Cosine Similarity* (Similaridade por Cosseno) (BILENKO et al., 2003) é uma função baseada em *tokens*, ou seja, as *strings* de entrada precisam ser transformadas em algo conhecido como *bag of words* ou em listas, e, dadas duas *strings* s1 e s2 em um domínino finito D, a similaridade por cosseno pode ser definida como:

$$Cosine(s1, s2) = \frac{V_{s1} * V_{s2}}{\|V_{s1}\| * \|V_{s2}\|} \tag{2.3}$$

onde:

- $\|V_{s1}\|$: $\sqrt{a^2 + b^2 + c^2 + ...}$ para um vetor $V_{s1} = [a, b, c, ...]$
- a dimensionalidade de um vetor é igual ao número de termos em D
- a i-ésima dimensão de um vetor representa o i-ésimo termo de D
- $V_{s[i]} = 0$ caso s não contenha o termo correspondente
- $V_{s[i]} = w(t_i)$, um peso associado ao i-ésimo termo de s, denotado como t_i.
- para comparação de *strings* diretamente, os termos (*tokens*) de duas entradas s1 e s2 podem ser utilizadas como o domínio D.

No contexto de recuperação de documentos, é utilizado *tf-idf* para calcular w. *Tf-idf*, uma abreviação do inglês para *term frequency-inverse document frequency*, significa a frequência do termo-inverso da frequência nos documentos, e é utilizada para calcular a importância de uma palavra em um conjunto de documentos, sendo que quanto menos a ocorrência de uma palavra, maior a importância dela para representar um dado documento (QAISER; ALI, 2018).

Os exemplos a seguir demonstram o funcionamento da função *Cosine Similarity* para duas *strings* s1 = *Aprender pode ser difícil* e s2 = *Aprender pode ser simples*, no contexto de comparação direta sem retenção de documentos.

- seja a representação vetorial dos textos, a partir da Tabela 2.1, V_{s1}: [1, 1, 1, 1, 0] e V_{s2}: [1, 1, 1, 0, 1]
- a similaridade por cosseno é calculada da seguinte forma:
  - é calculado o produto escalar entre V_{s1} e V_{s2}: $1*1+1*1+1*1+1*0+0*1 = 3$
  - é calculada a magnitude do vetor V_{s1}: $\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 0^2} = 2$
  - é calculada a magnitude do vetor V_{s2}: $\sqrt{1^2 + 1^2 + 1^2 + 0^2 + 1^2} = 2$
  - é calculada, enfim, a similaridade: $3/(2 * 2) = 0.75$

Tabela 2.1 – Representação vetorial dos textos

Palavra	V_{s1}	V_{s2}
Aprender	1	1
pode	1	1
ser	1	1
difícil	1	0
simples	0	1

Utilizando pesos *tf-idf* de uma coleção completa, podemos calcular a *Cosine Similarity* da seguinte maneira:

- São calculados os pesos *tf-idf* para os dados da Tabela 2.2 (representados na Tabela

2.3. O domínio D inclui todos os termos nos atibutos **título** e **gênero**.

- Sejam s1 e s2 os vetores dos CDs 4 e 5, é possível calcular a similaridade por cosseno como:
  $CosineSim(s1, s2) = \frac{0{,}134^2 \cdot 0{,}044^2}{\sqrt{0{,}134^2+0{,}253^2+0{,}253^2+0{,}253^2+0{,}044^2} \cdot \sqrt{0{,}134^2+0{,}253^2+0{,}253^2+0{,}044^2}}$
  $CosineSim(s1, s2) = 0,00021$

Tabela 2.2 – Fonte de dados descrevendo CDs

CdId	Título	Artista	Gênero
1	Britney	Britney	rock
2	ngap	britney	misc
3	Britney	Britney Spears	misc
4	Live from Las Vegas	Britney Spears	rock
5	Pink Floyd live	Pink Floyd	rock
6	Try This	Pink	rock
7	Try This	Pink	rock

Tabela 2.3 – Pesos tf-idf para as entradas selecionadas

Termo no domínio	Vetor para CD 4	Vetor para CD 5	Vetor para CD 1
Britney	0	0	0,134
ngap	0	0	0
Live	0,134	0,134	0
from	0,253	0	0
Las	0,253	0	0
Vegas	0,253	0	0
Pink	0	0,253	0
Floyd	0	0,253	0
Try	0	0	0
This	0	0	0
misc	0	0	0
rock	0,044	0,044	0,044

2.3.4 *Soft TF-IDF*

A função *Soft TF-IDF*, proposta em Cohen, Ravikumar e Fienberg (2003), é uma função híbrida que visa aproveitar os bons resultados de abordagens como Similaridade por Cosseno/TF-IDF (BILENKO et al., 2003) sem descartar automaticamente palavras que não são estritamente idênticas, utilizando, para isso, uma segunda função de similaridade. A definição formal define que: sendo $CLOSE(\theta, S, T)$ o conjunto de palavras $w \in S$ tal que existe um certo $v \in T$ tal que $dist'(w, v) > \theta$, onde $dist'$ é uma fun-

ção de similaridade secundária, e para $w \in CLOSE(\theta, S, T)$, seja $D(w, T) = max_{v \in T}$ $dist(w, v)$, se define

$$SoftTFIDF(S,T) = \sum_{w \in CLOSE(\theta,S,T)} V(w,S) \cdot V(w,T) \cdot D(w,T) \qquad (2.4)$$

2.4 Considerações Finais

Neste capítulo foram apresentadas as tecnologias e as técnicas que se mostraram fundamentais no desenvolvimento do trabalho apresentado nesta monografia. Elas foram de suma importância para atingir o objetivo, acelerando a evolução e gerando confiança nas decisões tomadas e nos resultados obtidos.

As técnicas vistas nesse capítulo foram utilizadas a exaustão na etapa de avaliação experimental, em especial as técnicas relacionadas à comparação de *strings* e nas que dizem respeito a coleta de dados, através de *Web Scraping*.

3 TRABALHOS RELACIONADOS

Neste capítulo, são apresentados os trabalhos relacionados à proposta desenvolvida. Inicialmente, é introduzido o trabalho de Schmitz et al. (2021), cuja abordagem proporcionou o desenvolvimento deste trabalho. Em seguida, são apresentados os trabalhos de Dozier et al. (2010), Papadakis et al. (2020) e Dalen-Oskam et al. (2014), que incluem abordagens de resolução de entidades utilizando *authority file* e deduplicação de registros. Depois, é apresentado o trabalho de Cohen, Ravikumar e Fienberg (2003), que faz uma comparação de funções de similaridade entre *strings* para tarefas de correspondência de nomes. Por fim, é realizada uma comparação entre as abordagens e suas implementações e a abordagem ACERPI-Link, desenvolvida neste trabalho.

3.1 Abordagem ACERPI

O abordagem ACERPI (SCHMITZ et al., 2021) tem como objetivo criar uma um modo genérico de obter, converter e estruturar Portarias, extrair informação e resolução de entidades de Portarias institucionais e oferecer ao usuário um banco de dados com informações referentes aos documentos, como os nomes dos servidores e suas matrículas identificadoras (SIAPE). Como este trabalho visa estender a abordagem original, propondo uma alternativa à etapa de Resolução de Entidades, o Capítulo 4 é dedicado a apresentar de forma mais detalhada da abordagem ACERPI.

3.2 Resolução de Entidades

O trabalho de Dozier et al. (2010) tem como foco métodos de reconhecimento de entidades nomeadas e resolução de entidades em textos legais. Em resumo, essas duas tarefas têm como objetivo, respectivamente, encontrar nomes de entidades dentro de textos e conectar esses nomes, no contexto deste trabalho, a uma entrada em uma base de dados pré-existente. No contexto de textos legais, as entidades envolvidas descrevem juízes, advogados, empresas, jurisdições e tribunais.

As entidades encontradas na etapa de de reconhecimento passam, então, por um processo de resolução de entidades que consiste em dois passos: extrair um nome identificado na etapa de reconhecimento de entidades nomeadas para uma respectiva classe

(como juízes ou advogados) a partir de informações associadas a este nome e, em seguida, armazenar esses dados em um registro. O segundo passo consiste na linkagem desse registro a uma entrada no arquivo de autoridades. O uso de informações das entidades é importante para a desambiguação no momento da resolução, algo semelhante ao desenvolvido neste trabalho, visto que é possível existir diversas entidades com o mesmo nome, o que diminuiria a precisão do algoritmo caso outros atributos não fossem utilizados. Um segundo ponto em comum entre os trabalhos foi o uso de um *gold standard*, que consiste em manualmente conectar uma pequena amostra das entidades às respectivas entradas no arquivo de autoridades para calcular a precisão e revocação do sistema desenvolvido.

O mecanismo de resolução de entidades desenvolvido por Dozier et al. (2010) consiste, na prática, no uso de blocagem para selecionar os principais candidatos de autoridades para uma entidade do texto e na utilização de vetores de atributos para a desambiguação, construídos a partir de todas as informações retiradas do texto para uma entidade.

No trabalho de Papadakis et al. (2020), o foco é na escalabilidade e otimização do processo de resolução de entidades, atacando o primeiro de dois passos, que é a escolha de candidatos que sejam válidos para a comparação com as entidades nomeadas no segundo passo desse processo, que é a comparação propriamente dita utilizando os candidatos selecionados para determinar se eles representam ou não a mesma entidade do mundo real. É destacado, ainda no trabalho de Papadakis et al. (2020), que a resolução de entidades pode se dar através de deduplicação, utilizada na abordagem ACERPI, onde uma coleção de de entidades como entrada e produz, como saída, o conjunto de pares de todas as entidades correspondentes. Outra opção apresentada, similar ao que é proposto neste trabalho, é a de *record linkage* (conexão de registros), que consiste em utilizar uma segunda coleção, livre de duplicatas, que retorna os pares de entidade entre as duas coleções.

Como a resolução de entidades é um problema inerentemente quadrático, Papadakis et al. (2020) sugere duas opções: a primeira é a utilização de Blocagem, agrupando em blocos comuns as entidades potencialmente coincidentes e realizando comparações apenas entre entidades que co-ocorrem em ao menos um bloco, trocando uma pequena perda de eficácia por um ganho significativo em eficiência. A segunda opção é a aplicação de Filtragem onde, dadas duas coleções de entidades, uma função de similaridade e um limiar, uma união das duas coleções identifica todos os pares em cada uma das coleções que tem uma similaridade maior ou igual ao limiar. Essa ténica pode ser considerada

como *record linkage*, enquanto a deduplicação aconteceria se essa técnica fosse passada considerando uma única coleção de entidades.

Por fim, o trabalho de Dalen-Oskam et al. (2014) tem como objetivo a criação de uma ferramenta que permita a busca por nomes de entidades em textos literários. Para isso, os autores definem que é necessário realizar as tarefa de reconhecimento de entidades nomeadas nesses tipos texto, ou seja, de personagens fictícios e, posteriormente, performar a resolução de entidades destes nomes para entradas na Wikipédia.

A tarefa de resolução de entidades foi realizada utilizando uma ferramenta que realiza uma tarefa conhecida como *wikification*, ou likagem de entidades, o que significa que essa ferramenta irá encontrar prováveis referências para as entidades (representadas por artigos da Wikipédia) em um processo que é realizado em dois passos. Primeiro, candidatos a menção dos nomes são identificados, sendo estes candidatos *n-gramas* contíguos aos nomes, além de também serem artigos na Wikipédia, títulos ou *links* para esses artigos. Em seguida, para cada candidato a menção, as entidades deste candidato são ranqueadas baseadas na probabilidade que uma dada menção se referir àquela entidade. O texto destaca que não é possível fazer uma avaliação do desempenho deste método, dado que os desenvolvedores não possuem um *gold standard* para realizar essa avaliação.

3.3 Comparação de Funções de Similaridade Entre *Strings*

O trabalho Cohen, Ravikumar e Fienberg (2003) tem como objetivo comparar diversos métodos desenvolvidos para a tarefa de comparação de *strings*. Primeiramente, são apresentadas as abordagens e os métodos existentes em cada uma delas. Alguns desses métodos são apresentados com mais detalhes na Seção 2.3. Em seguida, na seção de experimentos, os autores comparam essas diversos métodos de acordo com a finalidade para a qual foram desenvolvidos: *matching* ou blocagem.

Como é apresentado no Capítulo 6, algumas das funções de similaridade e resultados obtidos no trabalho de Cohen, Ravikumar e Fienberg (2003) foram aplicadas ao longo do desenvolvimento deste trabalho, especificamente para a tarefa de resolução de entidades.

3.4 Considerações Finais

Neste capítulo, foram apresentados cinco trabalhos relacionados à abordagem ACERPI-Link, desenvolvida neste trabalho. Primeiramente, a abordagem ACERPI, cuja proposta original consistia, dentre diversas etapas, a tarefa de resolução de entidades nomeadas utilizando a comparação exata entre *strings*. Em seguida, foram descritos os trabalhos de Dozier et al. (2010), Papadakis et al. (2020) e Dalen-Oskam et al. (2014), cujas abordagens para a tarefa de resolução de entidades nomeadas incluem soluções focadas tanto em eficácia quanto em eficiência. O último desses três trabalhos propõe, diferente dos demais, o uso de referências externas para a linkagem de entidades, enquanto os dois primeiros enfatizam o uso de um *authority file*. Por fim, foi descrito o trabalho de Cohen, Ravikumar e Fienberg (2003), onde os autores descrevem abordagens de comparação de funções de similaridade entre *strings*.

A Tabela 3.1 descreve, considerando as propostas relacionadas a tarefa de resolução de entidades, quais métodos são utilizados na abordagem ACERPI-Link e quais são utilizados nos trabalhos relacionados mencionados. Considerando o trabalho de Cohen, Ravikumar e Fienberg (2003), os métodos de comparação de *strings* utilizadas são detalhas na Seção 2.3, as quais não incluem métodos que utilizam blocagem.

Tabela 3.1 – Comparação entre abordagens propostas nos trabalhos relacionados

Abordagem	**Resolução de Entidades**	**Uso de Deduplicação**	**Uso de Blocagem**	**Uso de *Authority File***
DOZIER et al., 2010	Sim	Não	Sim	Sim
PAPADAKIS et al., 2020	Sim	Sim	Sim	Sim
DALEN-OSKAM et al., 2014	Sim	Não	Não	Não
SCHMITZ et al., 2021	Sim	Sim	Não	Não
ACERPI-Link	Sim	Não	Não	Sim

4 ACERPI

Este capítulo apresenta de forma resumida a abordagem ACERPI (SCHMITZ et al., 2021), um trabalho que tem como objetivo utilizar técnicas que vão desde descoberta até a estruturação de Portarias, extração de informação e resolução de entidades para criar um mecanismo de busca de informações profissionais de servidores públicos de uma maneira categorizada, filtrada e agrupada a partir de bases de dados de portarias federais.

4.1 Visão Geral da Abordagem ACERPI

A ACERPI propõe a criação de um banco de dados não relacional para consultas avançadas a respeito dos documentos relacionados a um servidor de uma Instituição, bem como quais servidores são referenciados em um dado documento publicado. O objetivo dessa abordagem é permitir uma pesquisa simples e fácil a respeito da jornada de servidores públicos através de documentos disponíveis abertamente em repositórios de órgãos federais, no contexto deste trabalho, portarias. O fluxo da abordagem ACERPI tem como entrada um conjunto de repositórios de documentos de Portarias como o repositório da (UFRGS, 2016), que contém documentos como o ilustrado na Figura 4.1 e gera como saída um banco de dados com as informações estruturadas dos servidores mencionados, bem como detalhes dos documentos e dos seus metadados.

A Figura 4.2 ilustra o fluxo geral dos dados utilizados desde sua fonte até o armazenamento e pós-processamento. A etapa de coleta é dividida entre descoberta e obtenção dos arquivos, conversão para o formato de texto e estruturação dos arquivos de texto para arquivos XML. Em seguida, na etapa de extração são identificadas as referências aos servidores e demais metadados para, em seguida, armazená-los em um formato padrão. Por fim, na etapa de resolução de entidades, são identificados quais registros referenciam os mesmos servidores, utilizando técnicas de *Entity Resolution* (Data Community DC, 2013). Cada uma dessas etapas é apresentada com mais detalhes nas subseções abaixo.

4.1.1 Coleta

Essa etapa inclui a descoberta e obtenção dos arquivos, a conversão para formato textual e a estruturação dos documentos de texto para arquivos XML identificando as

Figura 4.1 – Portaria número 10403 de 13/11/2017, emitida pela Administração Central Universidade Federal do Rio Grande do Sul

SERVIÇO PÚBLICO FEDERAL

PORTARIA Nº 10403 de 13/11/2017

A PRÓ-REITORA DE GESTÃO DE PESSOAS DA UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL EM EXERCÍCIO, no uso de suas atribuições que lhe foram conferidas pela Portaria nº. 8117, de 10 de outubro de 2016, e conforme a Solicitação de Afastamento n°32907,

RESOLVE

Designar, temporariamente, nos termos da Lei nº. 8.112, de 11 de dezembro de 1990, com redação dada pela Lei nº.9.527, de 10 de dezembro de 1997, a ocupante do cargo de PROFESSOR DO MAGISTÉRIO SUPERIOR, do Quadro de Pessoal desta Universidade, **RENATA DE MATOS GALANTE** (Siape: 1488770), para substituir CARLA MARIA DAL SASSO FREITAS (Siape: 0351477), Diretor do Instituto de Informática, Código CD-3, em seu afastamento no país, no período de 14/11/2017 a 15/11/2017, com o decorrente pagamento das vantagens por 2 dias.

VÂNIA CRISTINA SANTOS PEREIRA
Pró-Reitora

Documento gerado sob autenticação Nº LQH.287.570.56V, disponível no endereço http://www.ufrgs.br/autenticacao

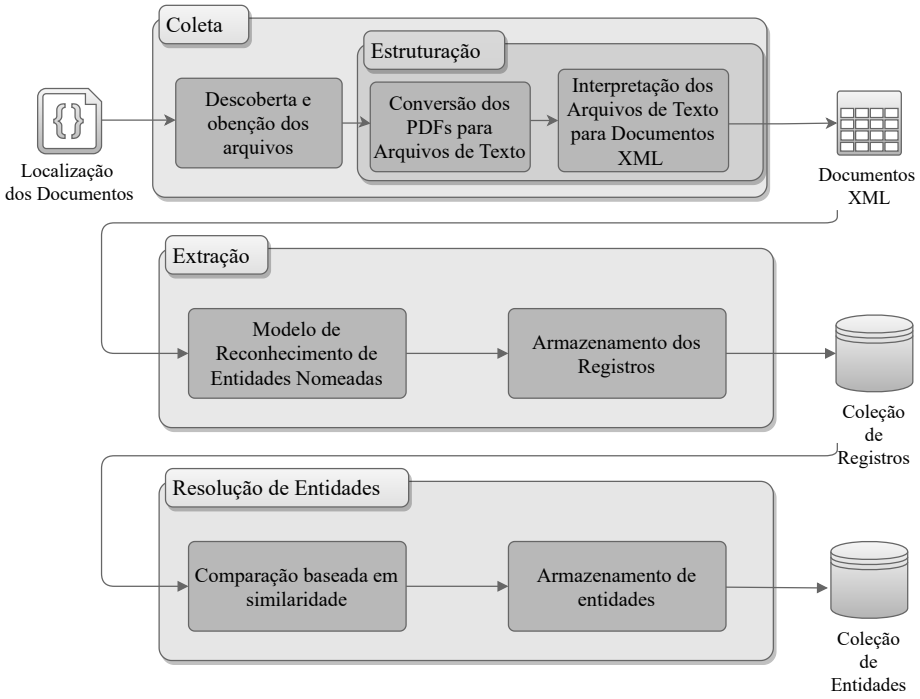
Documento certificado eletronicamente, conforme Portaria nº 3362/2016, que institui o Sistema de Documentos Eletrônicos da UFRGS.

1/1

portarias publicadas no documento em questão. Essas duas etapas acontecem da seguinte maneira:

- **Descoberta e Obtenção dos Arquivos** - Os arquivos PDF de Portarias são baixados das Instituições através de técnicas de *Web Scraping* (Seção 2.1) e, a depender de cada repositório e suas limitações, outras técnicas são aplicadas. Além disso, também são utilizadas técnicas de inferência de padrões de navegação (Palmieri Lage et al., 2004) e, ainda, de APIs que disponibilizam os dados de forma estruturada e

Figura 4.2 – Visão Geral do funcionamento da ACERPI



de fácil acesso.

- **Estruturação** - Essa etapa consiste em transformar os dados dos arquivos PDF para um formato de texto e, em seguida, esses textos são interpretados (utilizando expressões regulares) e deles são extraídas e convertidas para um formato XML as múltiplas Portarias que podem constar em cada arquivo e suas principais informações.

4.1.2 Extração

Essa etapa consiste em utilizar de técnicas de *Named Entity Recognition* (MARSHALL, 2019) e *Transfer Learning* para identificar referências aos servidores e os metadados relacionados a eles e, por fim, armazená-los em um formato padrão. Essa sequência de passos é resumida abaixo:

1. **Reconhecimento de Entidades Nomeadas** - Essa etapa consiste em extrair do conteúdo das Portarias os nomes dos servidores mencionados e suas respectivas matrículas SIAPE (um identificador único de cada servidor) utilizando expressões regulares, processamento de linguagem natural e um modelo de linguagem adaptado para o domínio de Portarias.
2. **Armazenamento dos Registros** - Nesta etapa os registros são armazenados no

banco de dados, o que permite a estruturação parcial dos dados, dando acesso às informações extraídas na etapa anterior, o que já permite a realização de consultas. O banco de dados escolhido foi o MongoDB (MONGODB, 2020a), um banco de dados NoSQL (Seção 2.2) orientado a documentos que permite a rápida evolução do esquema conforme a própria aplicação evolui (AWS, 2020). Cada documento, denominado registro, concentra as informações de um servidor identificado em uma dada portaria e possui a seguinte estrutura:

1. Um identificador único do registro.
2. O nome do servidor identificado na etapa de reconhecimento de entidades nomeadas.
3. Uma lista de SIAPEs identificados na respectiva Portaria. Na abordagem ACERPI, quando essa matrícula não é encontrada para um servidor em uma Portaria, são inseridos à lista todos os SIAPES encontrados nessa Portaria.
4. O identificador da portaria de onde esse registro foi encontrado.

A Listagem 4.1 ilustra como é armazenado um registro no banco de dados após a etapa de extração de informações da Portaria 4.1.

Listagem 4.1 – Registro gerado a partir da Portaria 4.1 para a servidora Renata de Matos Galante

```
{
    "id": 131072,
    "name": "RENATA DE MATOS GALANTE",
    "siape": ["1488770"],
    "document": {
        "name": "47048"
    }
}
```

4.1.3 Resolução de Entidades

Essa etapa consiste em analisar os registros e identificar, a partir dos dados dos servidores armazenados em cada documento (registro), quais registros referenciam as mesmas entidades no mundo real. É esperado, por exemplo, que ao final dessa etapa

tenha sido identificado que os registros representados nas Listagens 4.1 e 4.2, gerados a partir das Portarias representadas nas Figuras 4.1 e 4.3, referenciem a servidora Renata de Matos Galante e que essas duas referências sejam à mesma entidade do mundo real, a Professora Doutora Renata de Matos Galante, do Instituto de Informática da Universidade Federal do Rio Grande do Sul. Esse processo ocorre sobre todos os registros e ao final tem-se armazenada uma estrutura que representa todas as associações entre portarias e servidores.

Figura 4.3 – Portaria número 900 de 31/01/2018, emitida pela Administração Central Universidade Federal do Rio Grande do Sul

SERVIÇO PÚBLICO FEDERAL

PORTARIA Nº 900 de 31/01/2018

O PRÓ-REITOR DE GESTÃO DE PESSOAS DA UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, no uso de suas atribuições que lhe foram conferidas pela Portaria nº.7684, de 03 de outubro de 2016, do Magnífico Reitor, e conforme a Solicitação de Férias n°34471,

RESOLVE

Designar, temporariamente, nos termos da Lei nº. 8.112, de 11 de dezembro de 1990, com redação dada pela Lei nº.9.527, de 10 de dezembro de 1997, o ocupante do cargo de PROFESSOR DO MAGISTÉRIO SUPERIOR, do Quadro de Pessoal desta Universidade, **MARCELO WALTER** (Siape: 1550584), para substituir RENATA DE MATOS GALANTE (Siape: 1488770), Chefe do Depto de Informática Aplicada do Instituto de Informática, Código FG-1, em seu afastamento por motivo de férias, no período de 05/02/2018 a 14/02/2018, com o decorrente pagamento das vantagens por 10 dias.

MAURÍCIO VIÉGAS DA SILVA
Pró-Reitor de Gestão de Pessoas

Documento gerado sob autenticação Nº HJE.257.897.KMC, disponível no endereço http://www.ufrgs.br/autenticacao

Documento certificado eletronicamente, conforme Portaria nº 3362/2016, que institui o Sistema de Documentos Eletrônicos da UFRGS.

1/1

Listagem 4.2 – Registro gerado a partir da análise da Portaria da Figura 4.3 para a servidora Renata de Matos Galante

```
    {
        "id": 4630,
        "name": "RENATA DE MATOS GALANTE",
        "siape": ["1488770"],
        "document": {
            "name": "50216"
        }
    }
```

Na abordagem ACERPI, o método utilizado para identificar quais registros se referem às mesmas entidades é uma comparação baseada em similaridade que recebe como entrada os registros gerados a partir das Portarias e gera como saída o agrupamento dos registros que referem-se à mesma entidade do mundo real. A principal parte desse algoritmo é a função de *match*, que analisa primeiramente a matrícula SIAPE dos registros: quando únicas e idênticas é entendido que os registros se referem à mesma entidade do mundo real. Caso as matrículas SIAPE não sejam únicas e idênticas, é realizada uma comparação direta das entidades nomeadas nos registros onde um *match* ocorre caso esses nomes sejam idênticos após a remoção de espaços extras e descapitalização dos caracteres no momento da comparação entre os nomes. A Listagem 4.3 exemplifica a entidade gerada após a etapa de resolução de entidades para os registros das Listagens 4.1 e 4.2.

Listagem 4.3 – Entidade gerada a partir da resolução de entidades dos registros das Listagens 4.1 e 4.2

```
{
    "records": [47048, 50216],
    "names": ["RENATA DE MATOS GALANTE"],
    "siapes": ["1488770"]
}
```

4.2 Considerações Finais

Neste capítulo, foi apresentado um resumo da abordagem ACERPI. A Abordagem ACERPI constitui de três etapas principais, sendo elas a coleta dos arquivos dos repositórios das Instituições, a extração de informações e a resolução das entidades (DOZIER et al., 2010), foco de desenvolvimento deste trabalho conforme é mostrado com mais detalhes no Capítulo 5.

5 ACERPI-LINK: UMA ALTERNATIVA DE RESOLUÇÃO DE ENTIDADES EM PORTARIAS INSTITUCIONAIS UTILIZANDO LINKAGEM DE REGISTROS

Este capítulo apresenta uma alternativa à etapa de Resolução de Entidadas apresentada na abordagem ACERPI (SCHMITZ et al., 2021), uma abordagem que tem como objetivo utilizar técnicas que vão desde descoberta até a estruturação de Portarias, extração de informação e resolução de entidades para criar um mecanismo de busca de informações profissionais de servidores públicos de uma maneira categorizada, filtrada e agrupada a partir de bases de dados de portarias federais.

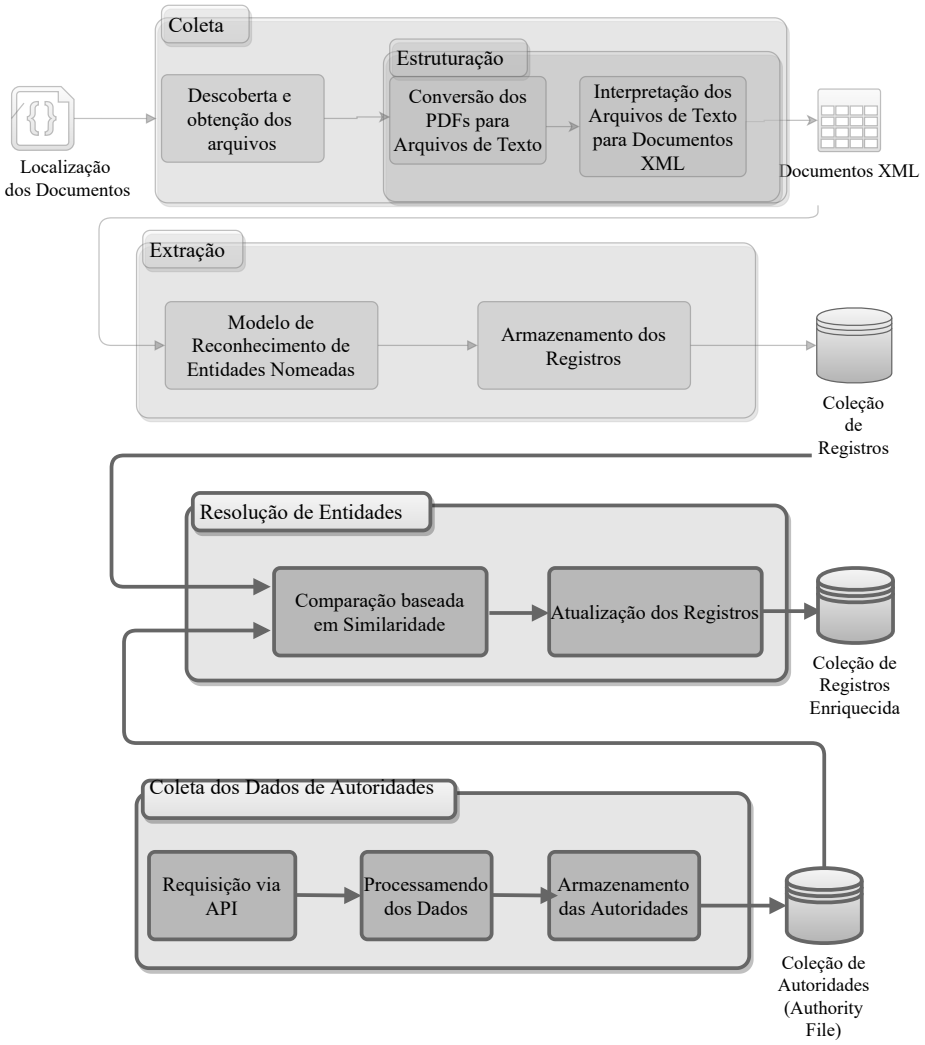
5.1 Uma Alternativa à Etapa Resolução de Entidades

Resolução de Entidades é a tarefa de desambiguar manifestações de entidades do mundo real em registros ou menções, através da criação de vínculos com uma base de dados pré-existente ou por agrupamento (Data Community DC, 2013). Por exemplo, a menção de uma servidora pública chamada *Renata de Matos Galante* pode ser resolvida para uma entrada específica de uma base de dados específicas de servidores públicos. Seguindo esse conceito, este trabalho se propõe a alterar o fluxo da abordagem ACERPI original, conforme ilustrado na Figura 5.1, que destaca as principais mudanças em relação à abordagem original: após a criação da coleção de registros, a etapa de resolução de entidades agora também recebe um *Authority File* (Seção 5.3). A partir disso, utilizando a comparação baseada por similaridade, em vez de gerar uma nova coleção representando as entidades, a saída desse novo método é a própria coleção de registros atualizada com um novo atributo que serve como *link* entre o registro e sua respectiva entidade no *Authority File*.

5.2 Coleção de Registros

A criação da coleção de registros acontece da mesma maneira que é apresentada na abordagem original ACERPI, e acontece após a etapa de de Extração e Armazenamento de Entidades, conforme apresentada na Seção 4.1.2. No Capítulo 6, esses registros são identificados como *records*.

Figura 5.1 – Fluxo de funcionamento da ACERPI-Link



5.3 Coleta dos Dados de Autoridades

Um *Authority File* (arquivo de autoridades) é um arquivo no qual cada registro identifica uma determinada pessoa ou entidade no mundo real (DOZIER et al., 2010). Exemplos de *authority file* incluem arquivos pertinentes ao domínio jurídico, por exemplo, como advogados, juízes, testemunhas, empresas, escritórios de advocacia e tribunais.

No contexto deste trabalho, uma autoridade é o nome dado a cada um dos registros de um *Authority File*. No banco de dados, cada entradada desse arquivo é representada em um documento com mais de 70 atributos. A Listagem 5.1 ilustra, resumidamente, os principais atributos de uma autoridade para o contexto deste trabalho. As autoridades são criadas a partir de uma API, conforme explicado abaixo, e são armazenados nos registros dados como a instituição na qual o servidor está em exercício e em qual o servidor está lotado. Um detalhe importante é que, como alguns dados são sensíveis, a API utilizada

formata alguns campos a fim de proteger os dados dos servidores, como é o caso do campo referente ao SIAPE do servidor (`codigoMatriculaFormatado`), onde apenas os três primeiros dígitos estão disponíveis.

A base de dados utilizada no processo de criação do *Authority File* foi construída a partir da API de serviços do Portal da Transparência do Governo Federal (Portal da Transparência do Governo Federal, 2021). Essa API gratuita possibilita buscar por todos os servidores do Poder Executivo Federal a partir de diversos filtros (é necessária uma chave para acessar a API, podendo ser criada no endereço <http://www.portaldatransparencia.gov.br/api-de-dados/cadastrar-email>. Para este trabalho, a busca por servidores foi feita a partir dos órgãos de exercício, seguindo o padrão de navegação (Palmieri Lage et al., 2004) <http://api.portaldatransparencia.gov.br/api-de-dados/servidores?orgaoServidorExercicio={}&pagina={}>, onde o código do órgão de exercício do servidor foi substituído pelo valor 26244 (código da UFRGS), e também foi necessário usar um *script* para varrer todas as páginas em busca dos servidores, pois a API limita a 15 registros por página consultada. A mesma ideia foi aplicada para buscar servidores a partir do órgão em que o servidor está lotado mas, não necessariamente, em exercício. Para isso, basta trocar o endereço para <http://api.portaldatransparencia.gov.br/api-de-dados/servidores?orgaoServidorLotacao={}&pagina={}> e repetir o restante do processo. Como é ilustrado na Figura 5.1, a criação deste *authority file* é um processo independente das demais etapas, podendo ser feito a qualquer momento que anteceda a etapa de Resolução de Entidades.

Listagem 5.1 – Principais atributos de uma entrada no *Authority File*

```
{
    "_id": ObjectId("60e787b367f9712dece974c0"),
    "servidor": {
        "id": 56705627,
        "pessoa": {
            "id": 6105539,
            "nome": "RENATA DE MATOS GALANTE"
         },
        "orgaoServidorLotacao": {
            "codigo": "26244",
            "nome": "Universidade Federal do Rio Grande
               do Sul",
```

```
12              "codigoOrgaoVinculado": "15000"
13          },
14          "orgaoServidorExercicio": {
15              "codigo": "26244",
16              "nome": "Universidade Federal do Rio Grande
                    do Sul",
17              "codigoOrgaoVinculado": "15000"
18          },
19          "codigoMatriculaFormatado": "148**"
20      }
21  }
```

5.4 Comparação Baseada em Similaridade

O processo de resolução de entidades da ACERPI-Link se dá por meio da criação de um vínculo entre um registro e sua respectiva entrada em um *Authority File*. Esse processo acontece conforme descrito no Algortimo 1, que recebe como entrada o *Authority File* e a coleção de registros criada a partir das Portarias e gera, como saída, essa mesma coleção de registros enriquecida com um novo atributo que tem como valor, para cada registro, o identificador único de sua respectiva autoridade no *Authority File*.

Algorithm 1: Algoritmo para a resolução de entidades.

Entrada: conjunto de registros, conjunto de autoridades
Saída : cojunto de registros enriquecido

1 $registros \leftarrow$ conjunto de registros;
2 $autoridades \leftarrow$ conjunto de autoridades;
3 **foreach** *registro* r *from registros* **do**
4 $\quad id\_autoridade \leftarrow$ *findAuthority*$(r, autoridades)$;
5 $\quad r[authority\_ref] \leftarrow id\_autoridade$;
6 **end**
7 **return** registros;

O método *findAuthority*, descrito no Algoritmo 2, é a parte principal do algoritmo, visto que ele é responsável por encontrar uma autoridade para cada um dos registros. A função de *match* utiliza comparação de *strings* por similaridade (Seção 2.3) e pode, também, utilizar o SIAPE dos registros e das autoridades quando estes estiverem presentes,

como é visto com mais detalhes na Seção 6.3. Um *match* entre uma *record* e uma autoridade se dá no momento em que acontece um *match*, o que pode ocorrer de duas formas:

1. Quando é considerado apenas o valor de similaridade entre os atributos que representam o nome de uma entidade em uma *record* e o nome de um servidor contido em um registro no *Authority File*. Esse valor é calculado pelas funções de similaridade e pode variar de 0 até 1, sendo 0 quando as *strings* não possuem nenhuma similaridade e 1 quando a função considera que as *strings* são iguais. Nesse caso, se a similaridade retornada pela função atingir um valor mínimo de limiar, o algoritmo considera isso como um *match*.
2. Quando, além de ter que atingir um valor mínimo de similaridade, o algoritmo também leva em conta o atributo que representa o SIAPE das entidades. Caso o campo SIAPE não seja vazio para uma *record*, o algoritmo também vai comparar os três primeiros dígitos desse campo com os três primeiros dígitos do atributo que representa o SIAPE em uma autoridade, visto que apenas esses três primeiros dígitos estão disponíveis em todos os documentos no arquivo de autoridades. Caso esses valores sejam iguais, então finalmente o algoritmo considera isso como um *match*. Caso a *record* tenha o atributo SIAPE vazio a função volta ao caso base e considera apenas o valor calculado pela função de similaridade para os nomes das entidades.

Algorithm 2: Algoritmo para encontrar a Autoridade

Entrada: registro, conjunto de autoridades
Saída : identificador de uma autoridade

```
1  registro ← registro;
2  autoridades ← conjunto de autoridades;
3  id_autoridade ← null;
4  foreach autoridade aut from autoridades do
5      if match(registro, autoridade) then
6          id_autoridade ← aut[id];
7          break;
8      end
9  end
10 return id_autoridade;
```

Assim, o registro representado pela Listagem 4.1 pode ser conectado à entrada do *Authority File* correspondente à autoridade contida no próprio registro. Esse vínculo estabelecido gera uma atualização no registro conforme representada na Listagem 5.2. No exemplo, o registro é enriquecido com a criação de um novo atributo, `authorityID`,

contendo o valor do atributo `_id` na entrada do arquivo de autoridades retornada pelo método *findAuthority*.

Listagem 5.2 – Registro gerado a partir da Portaria 4.1 para a servidora Renata de Matos Galante enriquecido

```
{
    "id": 131072,
    "name": "RENATA DE MATOS GALANTE",
    "siape": ["1488770"],
    "document": {
        "name": "47048"
    },
    "authorityId": ObjectId("60e787b367f9712dece974c0")
}
```

5.5 Considerações Finais

Neste capítulo, foi apresentada abordagem ACERPI-Link, uma alternativa à Abordagem ACERPI. Essa alternativa consiste em alterar a etapa de resolução de entidades a fim de utilizar a linkagem de registros, onde cada registro gerado na etapa de reconhecimento de entidades é conectado à uma entrada de um *authoriy file* através da comparação dos nomes e outros dados dos registros e entradas do *authority file* utilizando funções de similaridade.

6 AVALIAÇÃO EXPERIMENTAL

Este capítulo descreve os experimentos desenvolvidos para avaliar diferentes métodos na tarefa de resolução de entidades. Primeiramente é apresentada uma visão geral sobre quais são os experimentos apresentados nesta Seção. Em seguida, são apresentadas as configurações gerais que sustentam todos os experimentos. Por fim, são descritos os experimentos realizados e os resultados obtidos.

6.1 Visão Geral

Foram realizados cinco experimentos durante o desenvolvimento deste trabalho. São eles:

1. **Experimento 1** - Encontrar o melhor limiar para cada uma das funções de comparação de *strings*;
2. **Experimento 2** - Avaliar qual a melhor função de similaridade para a resolução de entidades;
3. **Experimento 3** - Avaliar o impacto de records sem uma autoridade na eficiência da função de *match*;
4. **Experimento 4** - Verificar o tempo de execução das funções de similaridade avaliadas no experimento 1;
5. **Experimento 5** - Verificar o impacto no Tempo de Execução e na Medida F1 ao utilizar o primeiro *match* considerando o melhor limiar e ao utilizar o melhor *match* possível para cada *record*, independentemente do valor de similaridade.

6.2 Configurações Gerais dos Experimentos

As partes em comum a todos os experimentos são explicadas nesta seção, enquanto as particularidades são abordadas individualmente no detalhamento de cada experimento.

6.2.1 Ambiente de Configuração

Todos experimentos foram realizados em um computador com processador Intel® Core™ i7-3630QM CPU @ 2.40GHz, 15,6 GiB de Memória, Arquitetura 64-bit, 8 CPUs e Sistema Operacional Ubuntu 18.4.5 LTS.

6.2.2 Bases de Dados

Essa subseção tem como objetivo explicar os dados utilizados nos experimentos deste trabalho, as fontes das quais esses dados foram retirados e também como eles foram tratados para facilitar o desenvolvimento dos experimentos. Para todos os experimentos, duas bases de dados foram utilizadas, são elas:

- Coleçao de Registros (*records*) - Ao final da etapa de Extração e Armazenamento, a coleção registros conta com 194.000 registros armazenados no banco de dados. A base de dados utilizada foi a de Documentos públicos da Universidade Federal do Rio Grande do Sul. Para essa fonte, foram utilizados os mecanismos de extração de documentos desenvolvidos na abordagem ACERPI, os quais podem ser acessados através de endereços generalizados a partir do padrão de navegação (Palmieri Lage et al., 2004) `https://www1.ufrgs.br/sistemas/sde/gerencia-documentos/index.php/publico/ExibirPDF?documento=[0-9]{1,6}`. Em ambos os trabalhos, a expressão regular foi substituída pelos valores de 18000 a 105995. Cada arquivo dessa fonte contém, no máximo, uma portaria emitida pela Universidade.
- *Authority File* - Coleção de autoridades apresentada na Seção 5.3. Ao final do processo de extração dos dados descritos na Seção 5.3, os dados de servidores lotados na UFRGS totalizam 12.348 registros, enquanto os que estão em exercício na UFRGS totalizam 6.623 registros. Para criar uma única coleção, foi realizada uma limpeza sobre os dados de servidores lotados na UFRGS, removendo os registros dos servidores que estão também em exercício na UFRGS, o que faz com que a união das duas coleções originais não contenha duplicatas. Após, foram removidos 6580 registros da coleção, restando 5.768 registros na coleção, o que significa que apenas 43 servidores em exercício na UFRGS estão lotados em outras instituições. Além disso, muitos servidores ainda apareciam mais de uma vez em ambas cole-

ções e, após inspeção manual e comparação de alguns dos atributos dos servidores que se repetem, percebeu-se que isso ocorre quando um servidor se aposenta e seu registro original não é removido, mas é gerado um novo registro alterando apenas a nova função deste servidor, agora aposentado. Após a união das duas coleções, dos 12.391 servidores, mais 1281 documentos foram removidos, restando 11.110 na coleção final de Autoridades utilizada neste trabalho.

A partir desses dados, foram criadas duas novas coleções de registros, aqui chamados de *records*, para funcionarem como um gabarito (*Gold Standard*) para a verificação dos resultados de cada experimento. Foi desenvolvido um trecho de código *(script)* que seleciona 1000 Portarias aleatoriamente e, para cada portaria aleatoriamente selecionada, adiciona a um dos gabaritos todas as *records* que foram geradas a partir dessa portaria. Ao final desse processo, esse gabarito contava com 1399 *records*. Dessas 1399 *records*, 1240 foram manualmente conectadas às suas respectivas entradas no *Authority File*. Foi verificado que, para a grande maioria das *records* que não foi possível encontrar uma autoridade, todas se tratavam de *records* geradas a partir de portarias que mencionavam estudantes ou até mesmo empresas, para as quais de fato não seria possível existir uma entrada no contexto de servidores públicos.

Dessa forma, são geradas as duas coleções que foram utilizadas em todos os experimentos. São elas:

- Coleção 1 - É a coleção com as 1240 *records* que foram conectadas com suas respectivas autoridades.
- Coleção 2 - É a coleção contendo as 1399 *records* contendo, inclusive, as *records* sem uma entidade no arquivo de autoridades.

6.2.3 Métricas

Para todos os experimentos, as seguintes métricas, baseadas nas definições de Dozier et al. (2010), são computadas e apresentadas através de gráficos e tabelas:

- Precisão - mede a frequência em que as autoridades atribuídas a cada *record* estão corretas. A fórmula da precisão pode ser descrita por $VerdadeirosPositivos/(VerdadeirosPositivos + FalsosPositivos)$.
- Revocação - é o percentual de *records* que de fato possuem um *match* com um registro no arquivo de Autoridades e que o modelo conse-

guiu identificar corretamente. A fórmula da revocação é descrita por $VerdadeirosPositivos/(VerdadeirosPositivos + FalsosNegativos)$. Para este trabalho, como cada *record* possui, no máximo, uma autoridade que realmente casa com o nome armazenado no registro, a soma $VerdadeirosPositivos + FalsosNegativos$ pode ser resumida ao número total de *records* que de fato possuem uma autoridade, ou seja, o número total de pares *record*-autoridade.

- Medida F1 - é uma média ponderada que utiliza Precisão (P) e Revocação (R) para, através de uma só métrica, medir a eficácia da abordagem. A fórmula da F1 pode ser descrita por $2 * P * R/(P + R)$.

Os valores que representam as métricas são definidos da seguinte manteira:

- Verdadeiros Positivos - quantidade de *records* que são associadas a uma autoridade e que se referem à mesma entidade no mundo real.
- Falsos Positivos - quantidade de *records* que foram associadas a uma autoridade incorretamente.

6.3 Experimento 1 - Escolha do Melhor Limiar

Este experimento tem como objetivo avaliar o desempenho das funções de similaridade apresentadas na Seção 2.3 sobre os dados apresentados na seção 6.2.2 e, a partir dos resultados obtidos neses diversos testes, identificar qual o o melhor limiar para cada função. Os testes seguem a seguinte estrutura: para cada função de similaridade, o objetivo é que todas as *records* encontrem um registro na coleção de autoridades que represente a mesma autoridade contida nas *records*. Como as funções de similaridade retornam valores entre 0 e 1, os limiares testados também variam nesse intervalo, sendo acrescido 0,1 a cada iteração em um primeiro momento e, após análise dos resultados, novos testes são rodados visando aprimorar a análise nos intervalos em que as funções mostram maior potencial. Nesse caso, é acrescido 0,02 ao valor do limiar a cada iteração. A métrica considerada para decidir qual limiar obteve o melhor desempenho é a Medida F1. Essa métrica sumariza as demais métricas mas, no caso de mais de um limiar atingir o valor máximo nessa métrica, a precisão obtida pela função é considerada para desempate.

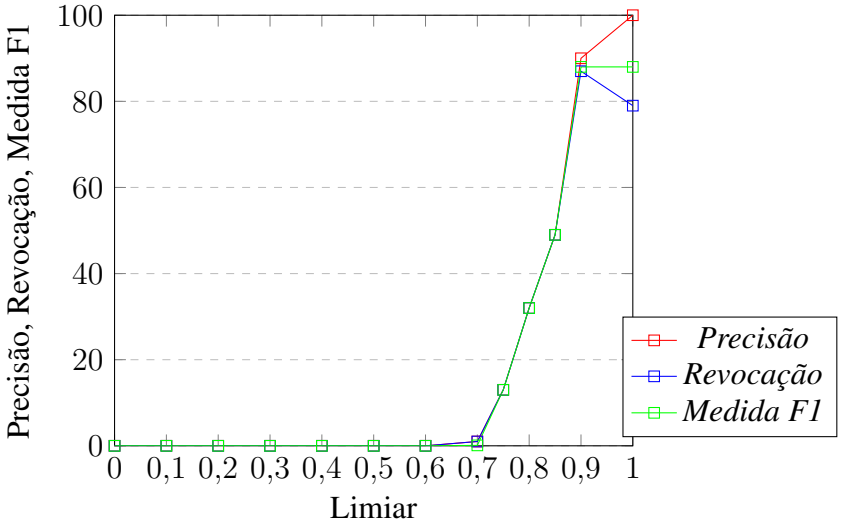
6.3.1 Jaro-Winkler

Nesta subseção, são detalhados os testes envolvendo a função de similaridade Jaro-Winkler visando buscar o melhor limiar para essa função.

6.3.1.1 Coleção 1, matrícula SIAPE sendo desconsiderada

Como mostra a Figura 6.1, essa função de similaridade tem um desempenho nulo para limiares de 0 até 0,7 e, a partir de 0,9, possui um desempenho satisfatório e equilibrado entre todas as medidas. A partir desses resultados, a fim de refinar os resultados, foram rodados novos testes variando o limiar entre 0,8 e 1 acrescentando 0,02 no valor a cada iteração. A Figura 6.2 mostra o desempenho da função para estes novos valores e, a partir dos resultados obtidos, levando em conta a Medida F1, 0,96 é o melhor valor de limiar para a função Jaro-Winkler nestas configurações sobre esses dados.

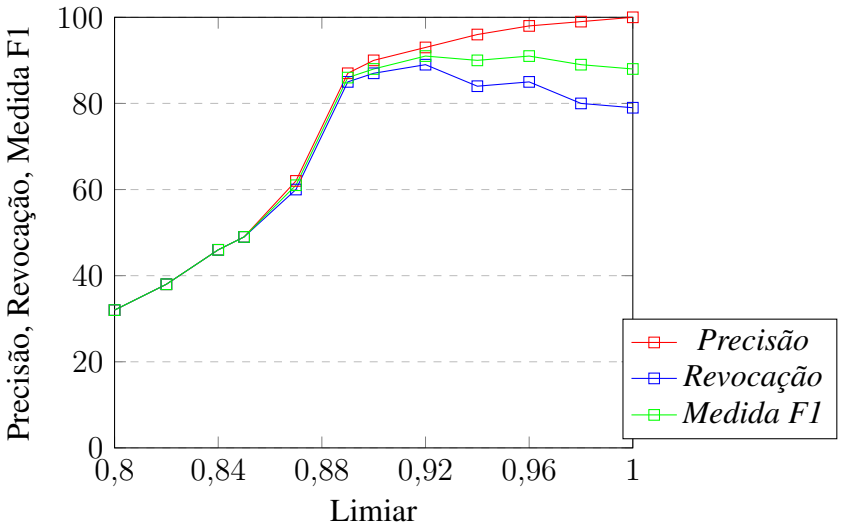
Figura 6.1 – Jaro-Winkler desconsiderando o campo SIAPE, coleção 1



6.3.1.2 Coleção 1, matrícula SIAPE sendo considerada

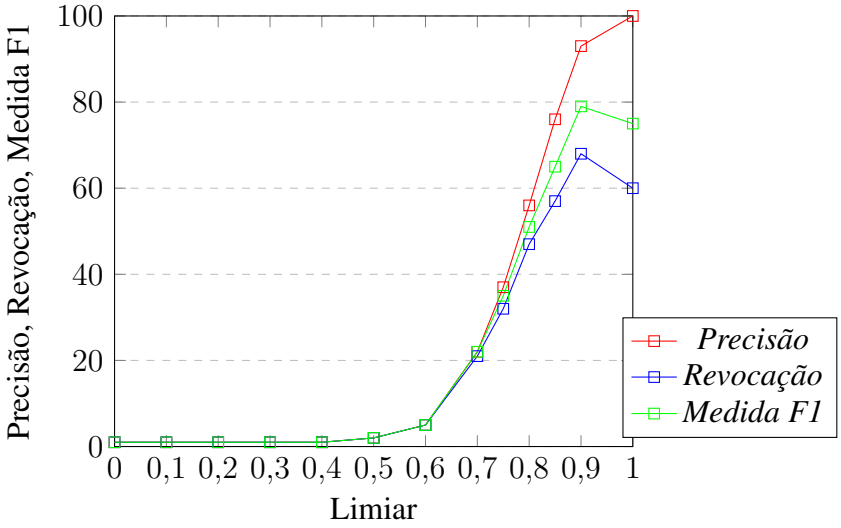
A Figura 6.3 mostra o desempenho da função Jaro-Winkler sobre a mesma coleção, agora considerando o atributo SIAPE das *records*. Como pode se observar, os resultados são muito baixos para limiares menores que 0,8. Para observar os resultados no intervalo onde a função obteve melhor desempenho, foram rodados novos testes com o limiar variando entre 0,8 e 1, onde a função apresenta maior potencial. A Figura 6.4 mostra o desempenho da função para este experimento, e tomando como base os resultados obtidos, levando em conta a Medida F1, a função Jaro-Winkler com limiar fixado em

Figura 6.2 – Jaro-Winkler desconsiderando o campo SIAPE, coleção 1 - Refinamento



0,92 obteve o melhor desempenho nessas configurações.

Figura 6.3 – Jaro-Winkler considerando o campo SIAPE, coleção 1



6.3.1.3 Coleção 2, matrícula SIAPE sendo desconsiderada

Como mostra a Figura 6.5, essa função de similaridade tem um desempenho nulo para limiares de 0 até 0,7 e, somente a partir do limiar 0,9 possui um desempenho satisfatório em todas as métricas. Por isso, visando refinar os resultados no intervalo onde a função teve um desempenho melhor, foram rodados novos testes variando o limiar entre 0,8 e 1. A Figura 6.6 mostra o desempenho da função para estes valores e, a partir dos resultados obtidos e tomando como base a Medida F1, com 91% de desempenho o melhor valor de limiar para a função Jaro-Winkler nestas configurações sobre esses dados é 0,96.

Figura 6.4 – Jaro-Winkler considerando o campo SIAPE, coleção 1 - Refinamento

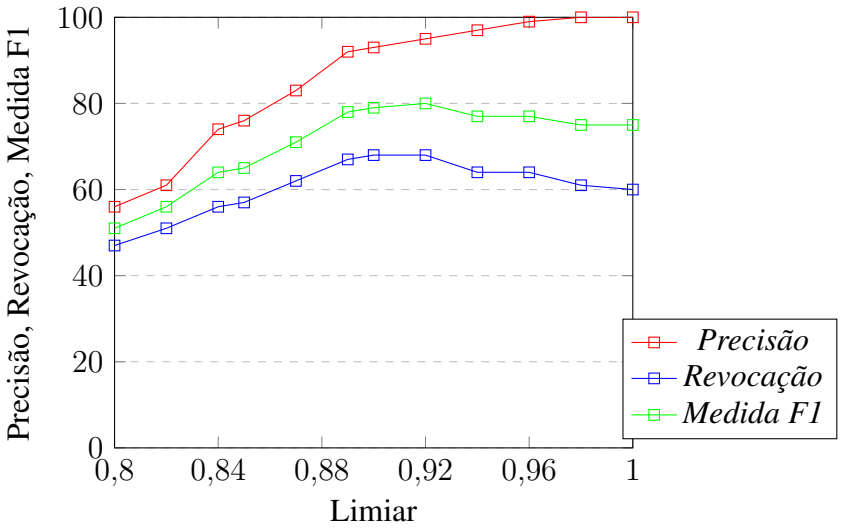
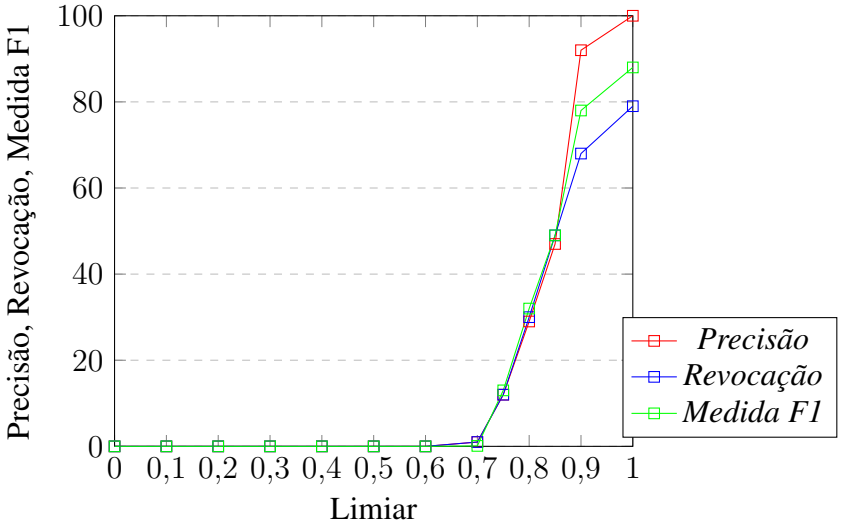


Figura 6.5 – Jaro-Winkler desconsiderando o campo SIAPE, coleção 2



6.3.1.4 Coleção 2, matrícula SIAPE sendo desconsiderada

A Figura 6.7 mostra o desempenho da função Jaro-Winkler nas métricas definidas para os experimentos sobre a coleção 2. Como pode se observar, essa função de similaridade tem um desempenho muito baixo para limiares de 0 até 0,75. Para refinar os resultados no intervalo onde a função teve um desempenho melhor, foram rodados novos testes variando o limiar entre 0,8 e 1. A Figura 6.8 mostra o desempenho da função para estes valores e, a partir dos resultados obtidos e observados, tomando como base a Medida F1, o limiar 1 com desempenho de 88% na Medida F1 é o melhor valor para a função Jaro-Winkler nestas configurações sobre esses dados.

Figura 6.6 – Jaro-Winkler desconsiderando o campo SIAPE, coleção 2 - Refinamento

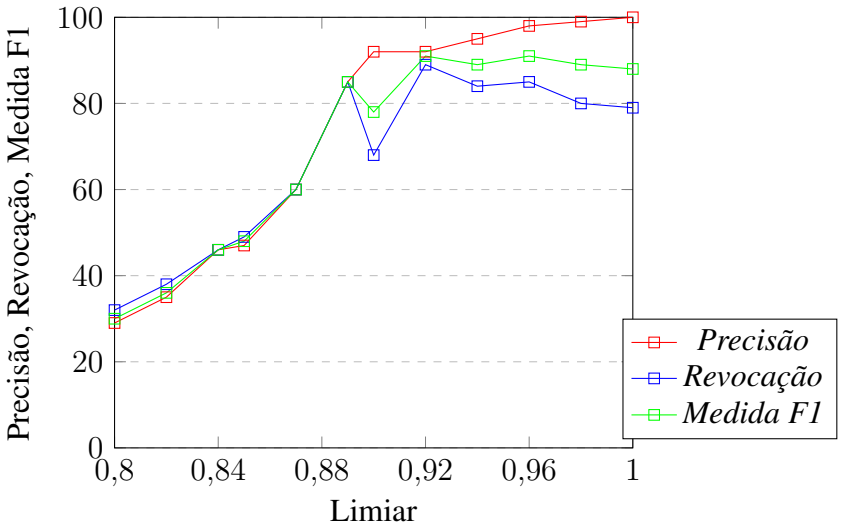
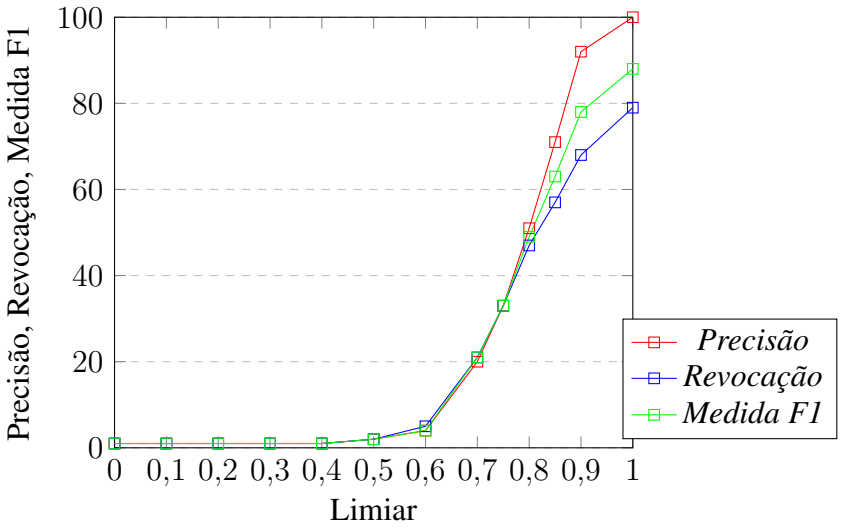


Figura 6.7 – Jaro-Winkler considerando o campo SIAPE, coleção 2

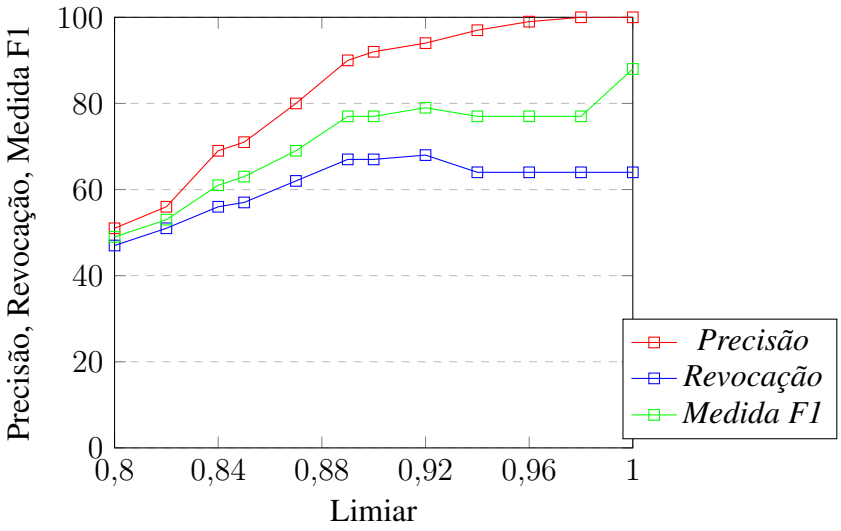


6.3.1.5 Discussão dos Resultados

Para a Coleção 1, no contexo em que o SIAPE é desconsiderado, o melhor limiar foi o de valor 0,96, que atribuiu corretamente uma autoridade para 1057 *records*, em um total de 1072 atribuições em uma coleção que contava com 1240 registros, totalizando uma precisão de 98% e 85% de revocação. Quando o SIAPE passou a ser considerado, o melhor limiar foi o de valor 0,92, que atribuiu corretamente 854 entidades em um total de 894 atribuições, totalizando uma precisão de 95% e 68% de revocação.

Para a Coleção 2, com o SIAPE não sendo considerado na função de *match*, o melhor limiar foi o de valor 0,96, que atribuiu corretamente 1057 entidades às *records*, em um total de 1074 atribuições, obtendo uma precisão de 98% e uma revocação de 85%. Quando o SIAPE foi considerado, o melhor limiar encontrado foi o de valor 1, que

Figura 6.8 – Jaro-Winkler considerando o campo SIAPE, coleção 2 - Refinamento



atribuiu corretamente 984 entidades com 100% de precisão e 79% de revocação. Vale ressaltar que para 159 das 1399 *records* desta coleção era esperado que de fato não fosse encontrada uma entidade no arquivo de autoridades, pois se tratavam de estudantes ou empresas.

6.3.1.6 Análise dos Casos de Falha

Em ambas as coleções, um motivo recorrente para as *records* não obterem um *match* foi por terem seus nomes extraídos de forma incompleta na etapa de Reconhecimento das Entidades Nomeadas. É o caso da servidora *Jane Fraga Tutikian*, que aparece em diversas *records* apenas como *Jane Fraga*, o que faz o valor de similaridade diminuir consideravelmente pela função Jaro-Winkler, mesmo compartilhando um prefixo até o limite considerado pela função. Outros casos envolvem o simples fato da função ter atingido o valor mínimo de similaridade para uma autoridade com o nome similar ao da entidade contida na *record*, como o servidor *Rafael Menezes Nunes* que foi conectado a uma autoridade com o nome *Rafael Menezes Luz*, visto que os nomes compartilham um prefixo significativo que aumentou a similaridade calculada pela função.

No contexto de considerar o SIAPE, além do problema envolvendo entidades com o nome sendo identificado de forma parcial, a ideia de adicionar todos os SIAPEs encontrados no documento para uma *record* cuja entidade não obteve um SIAPE identificado se mostrou um problema, visto que quando a função de similaridade obteve o valor mínimo para um limiar, como os SIAPEs eram diferentes o *match* acabou não acontecendo.

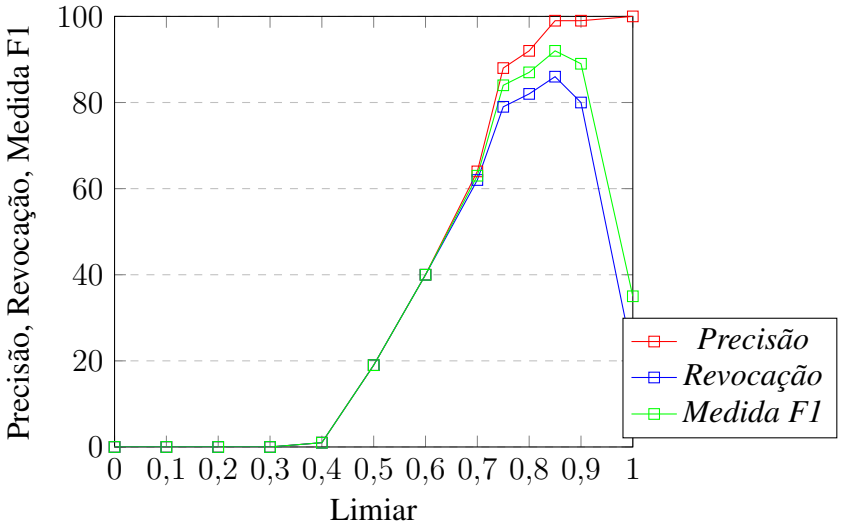
6.3.2 *Cosine Similarity*

Nesta seção, são detalhados os experimentos envolvendo a função *Cosine Similarity* com o objetivo de encontrar o limiar que otimiza os resultados para essa função sobre os dados apresentados na Seção 6.2.2 e as configurações especificadas para essa fase dos experimentos.

6.3.2.1 Coleção 1, matrícula SIAPE sendo desconsiderada

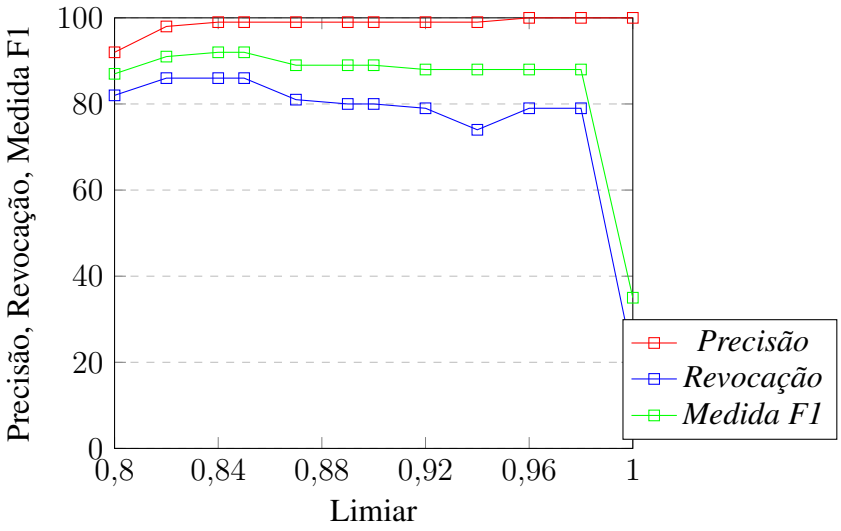
A Figura 6.9 mostra o desempenho da função *Cosine Similarity* de acordo com as métricas definidas para este experimento. A função apresenta um desempenho baixo para limiares entre 0 e 0,75 e, com o objetivo de refinar o desempenho da função no intervalo onde a função teve um melhor desempenho, novos testes foram rodados com o limiar variando entre 0,8 e 1, acrescendo 0,02 a cada iteração.

Figura 6.9 – Cosine Similarity desconsiderando o campo SIAPE, coleção 1



Como mostra a Figura 6.10, a precisão desse método é próxima de 100% a partir de limiares menores do que na função Jaro-Winkler e, por se tratar de uma função mais rigorosa na avaliação da similaridade, quando o limiar é colocado no valor máximo, a função obtém um desempenho muito inferior devido à grande quantidade de registros que não conseguem atingir esse valor para um *match*. Por fim, o valor de limiar fixado em 0,84 nessa função para esses dados garante o melhor desempenho, com 92% na Medida F1 e 99% de precisão.

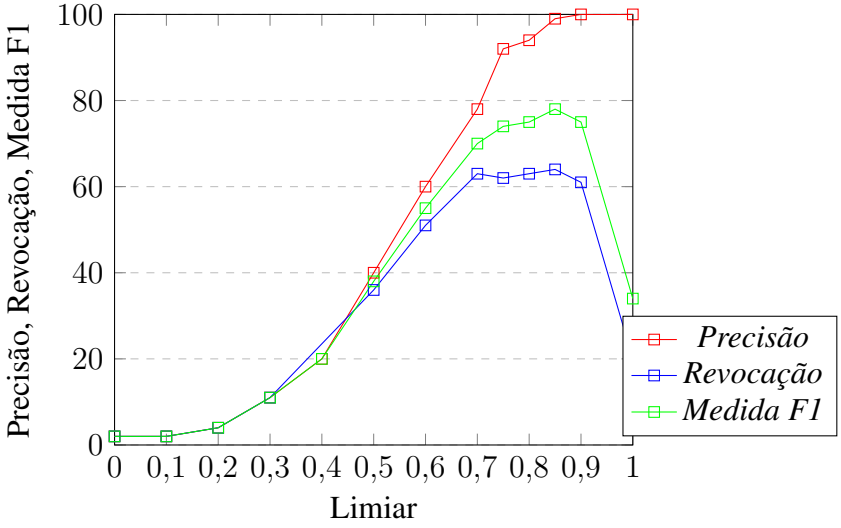
Figura 6.10 – Cosine Similarity desconsiderando o campo SIAPE, coleção 1 - Refinamento



6.3.2.2 Coleção 1, matrícula SIAPE sendo desconsiderada

A Figura 6.11 mostra o desempenho da função *Cosine Similarity* sobre a mesma coleção, agora considerando o atributo SIAPE das *records*. Como se pode observar, essa função de similaridade tem um desempenho baixo para limiares abaixo de 0,7 e apresenta maior potencial a partir de 0,8. Para refinar os resultados deste experimento, novos testes foram rodados com o limiar variando entre 0,8 e 1.

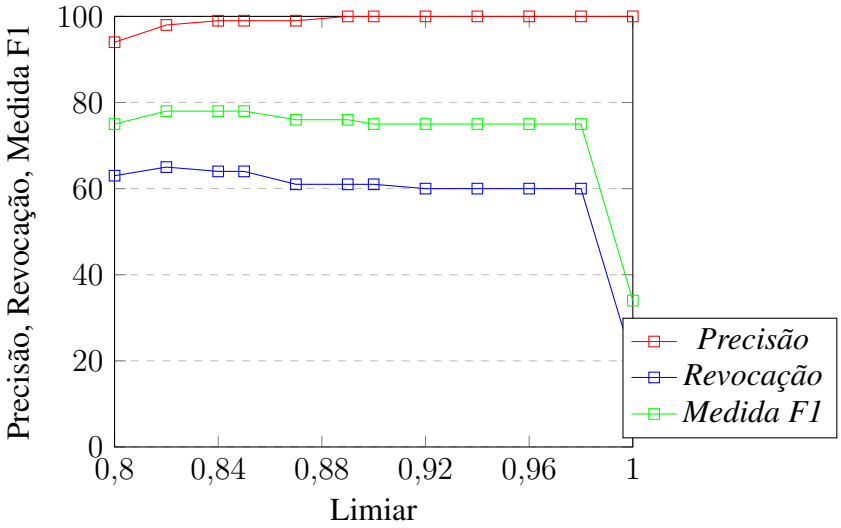
Figura 6.11 – Cosine Similarity considerando o campo SIAPE, coleção 1



Como mostram os resultados observados na Figura 6.12, essa função de similaridade tem uma alta precisão e valores mais baixos para revocação devido ao seu critério mais rigoroso para avaliar um *match*. Isso reflete nos resultados da Medida F1, em que os melhores valores de limiar flutuam entre 0,8 e 0,85. Levando em conta também a preci-

são, fica definido que o valor de 0,84 é o melhor limiar para essa configuração da função sobre esses dados.

Figura 6.12 – Cosine Similarity considerando o campo SIAPE, coleção 1 - Refinamento



6.3.2.3 Coleção 2, matrícula SIAPE sendo desconsiderada

A Figura 6.13 mostra o desempenho da função *Cosine Similarity* de acordo com as métricas definidas para este experimento, agora sobre a segunda coleção de dados e, como é possível observar, este desempenho é baixo para limiares entre 0 e 0,75. Visando refinar os resultados no intervalo onde a função teve um melhor desempenho, novos testes foram realizados com o limiar variando entre 0,8 e 1, acrescendo 0,02 a cada iteração. A Figura 6.14 mostra o resultado dessa nova avaliação e, de acordo com os resultados obtidos, novamente a precisão desse método é próxima de 100% a partir de limiares menores do que na função Jaro-Winkler. O limiar fixado em 0,84 tem o melhor desempenho considerando as três métricas.

6.3.2.4 Coleção 2, matrícula SIAPE sendo desconsiderada

A Figura 6.15 mostra o desempenho da função *Cosine Similarity* de acordo com as métricas definidas para este experimento. Como se pode observar, o desempenho é baixo para limiares abaixo de 0,7. Visando refinar os resultados no intervalo onde a função teve um melhor desempenho, novos testes foram realizados com o limiar variando entre 0,8 e 1.

Como mostra a Figura 6.16, a precisão desse método é próxima de 100% a partir de limiares muito inferiores em relação à função Jaro-Winkler, porém seu critério mais

Figura 6.13 – Cosine Similarity desconsiderando o campo SIAPE, coleção 2

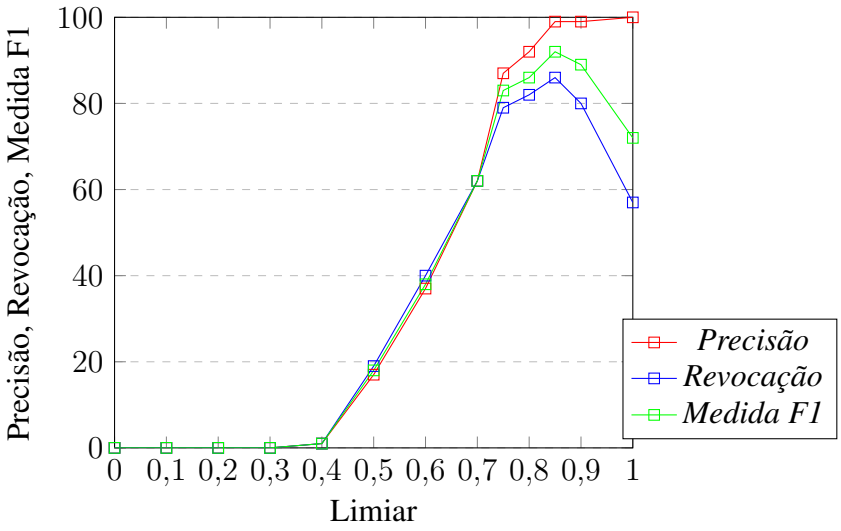
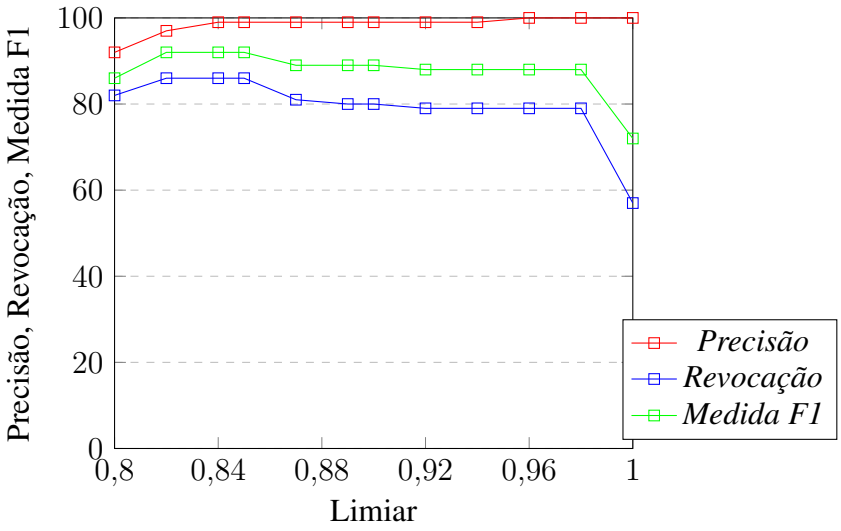


Figura 6.14 – Cosine Similarity desconsiderando o campo SIAPE, coleção 2 - Refinamento



rigoroso faz com que a revocação fique mais baixa, o que reflete nos resultados da Medida F1, que obtém o melhor resultado com o limiar fixado em 0,85 sobre esses dados nessas configurações considerando as três métricas de desempenho.

6.3.2.5 Discussão dos Resultados

Para a Coleção 1, com o atributo SIAPE sendo desconsiderado, o melhor limiar foi o de valor 0,84, que atribuiu corretamente uma autoridade para 1070 *records* em um total de 1079 atribuições em uma coleção que contava com 1240 registros, totalizando uma precisão de 99% e 86% de revocação. Quando o SIAPE passou a ser considerado, o melhor limiar foi o de valor 0,84, que atribuiu corretamente 803 entidades em um total de

Figura 6.15 – Cosine Similarity considerando o campo SIAPE, coleção 2

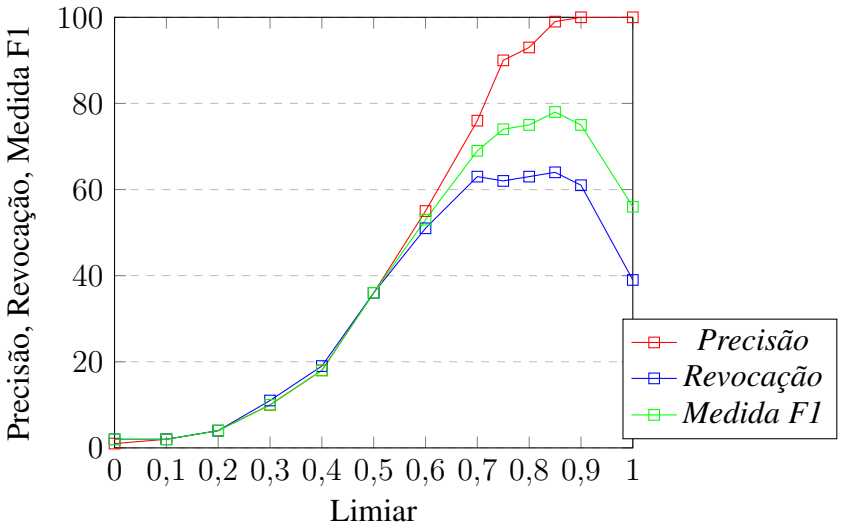
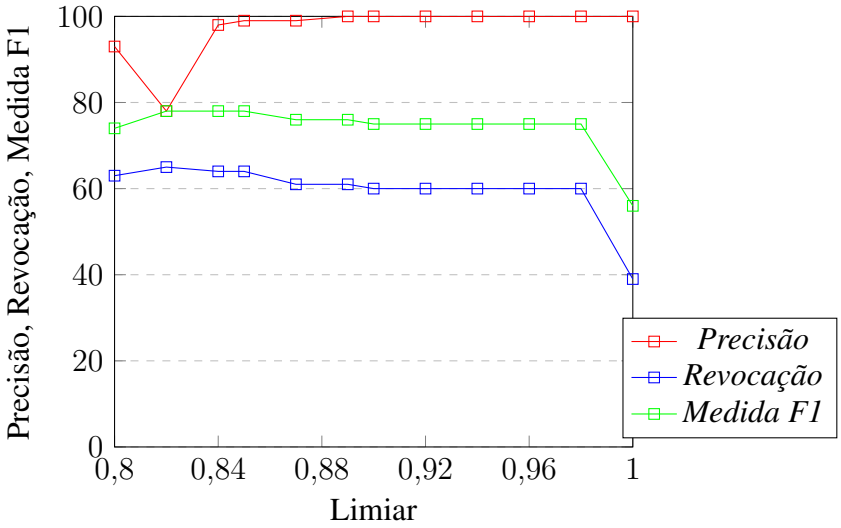


Figura 6.16 – Cosine Similarity considerando o campo SIAPE, coleção 2 - Refinamento



809 atribuições, totalizando uma precisão de 99% e 64% de revocação.

Para a Coleção 2, com o SIAPE não sendo considerado na função de *match*, o melhor limiar foi o de valor 0,82, que atribuiu corretamente 1070 entidades às *records*, em um total de 1092 atribuições, obtendo uma precisão de 97% e uma revocação de 86%. Quando o SIAPE foi considerado, o melhor limiar encontrado foi o de valor 0,84, que atribuiu corretamente 803 entidades em um total de 812 atribuições, atingindo 98% de precisão e apenas 65% de revocação.

6.3.2.6 Análise dos Casos de Falha

Novamente falhas no Reconhecimento das Entidades Nomeadas foi fator determinante para a grande maioria dos casos em que a função não obteve um **match** entre

record e sua respectiva autoridade. Casos como o da servidora *Ida Vanessa Doreder-lein Schwartz*, reconhecida apenas como *Ida* em uma das records foi comum dentre os casos de falha. Além disso, casos em que uma *record* encontrou uma autoridade com o nome suficientemente similar também aconteceram, como a *record* do servidor *Marcelo Utz Asconavieta* que acabou sendo conectada a autoridade *Marcio Utz Asconavieta*.

Quando o SIAPE foi considerado, esse problema logicamente impediu o *match* de acontecer antes mesmo da comparação entre SIAPEs. Além disso, o mesmo problema envolvendo SIAPEs que não pertenciam à entidade e foram atribuídos a ela em uma *record* acabou prejudicando o *match* quando este poderia ter acontecido.

6.3.3 *Soft TF-IDF*

Nesta seção, são detalhados os testes envolvendo a função *Soft TF-IDF* (Seção 2.3.4) com o objetivo de encontrar o limiar que otimiza os resultados para essa função sobre os dados apresentados na Seção 6.2.2 e as configuraçõs especificadas para essa fase dos experimentos. É importante ressaltar que para essa função foram seguidas as seguintes configurações baseadas na proposta original desse método (COHEN; RAVIKUMAR; FIENBERG, 2003): o valor de θ foi fixado em 0,9 para a função de similaridade secundária necessária para o funcionamento deste método, sendo essa função secundária a função Jaro (JARO, 1989).

6.3.3.1 Coleção 1, matrícula SIAPE sendo desconsiderada

Como mostra a Figura 6.17, essa função de similaridade tem um desempenho baixo para limiares de 0 até 0,3 e entra em uma crescente a partir de 0,4, atingindo o melhor valor para a Medida F1 com o limiar fixado em 0,6. Foram rodados novos testes para limiares variando de 0,5 e até 0,7, intervalo onde a função demonstrou maior potencial, acrescentando 0,02 no valor a cada iteração. A Figura 6.18 mostra o desempenho da função para esses valores e, a partir dos resultados obtidos, levando em conta a Medida F1, o melhor valor de limiar para a função softTFIDF nestas configurações é o de valor 0,64.

Figura 6.17 – SoftTFIDF desconsiderando o campo SIAPE, coleção 1

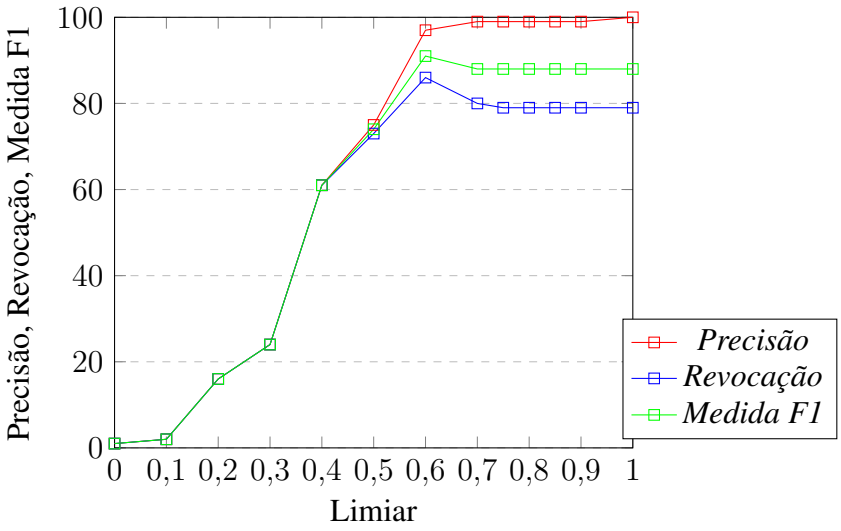
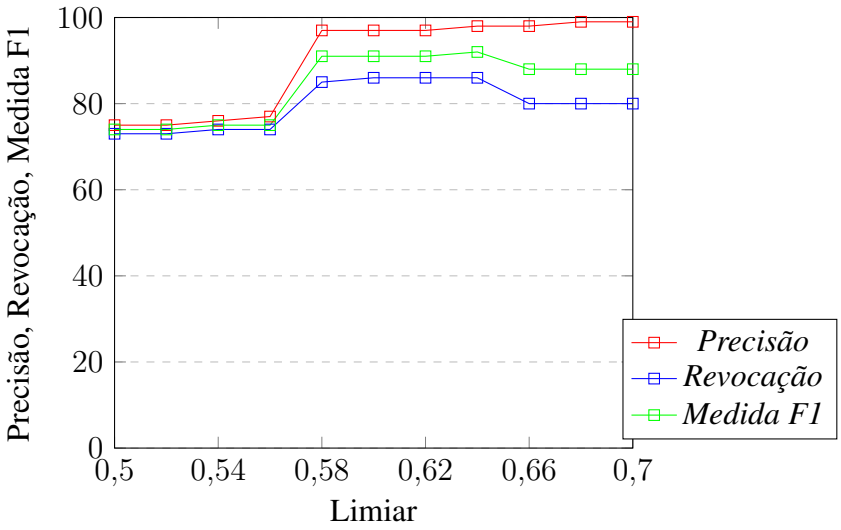


Figura 6.18 – SoftTFIDF desconsiderando o campo SIAPE, coleção 1 - Refinamento



6.3.3.2 Coleção 1, matrícula SIAPE sendo desconsiderada

A Figura 6.19 mostra o desempenho da função SoftTFIDF sobre primeira coleção de dados, agora considerando o atributo SIAPE das *records*. Como é mostrado na Figura 6.19, essa função de similaridade tem um desempenho baixo para limiares de 0 a 0,4 e mostra uma melhora expressiva a partir de 0,5.

Foram rodados novos testes variando o limiar entre 0,5 e 0,7, intervalo onde a função mostrou maior potencial, acrescentando 0,02 no valor a cada iteração. A Figura 6.20 mostra o desempenho da função para estes testes e, como muitos limiares empataram com o maior valor de Medida F1 (78%), levando em conta a precisão de 92% o melhor limiar para essa função é o de valor 0,56.

Figura 6.19 – SoftTFIDF considerando o campo SIAPE, coleção 1

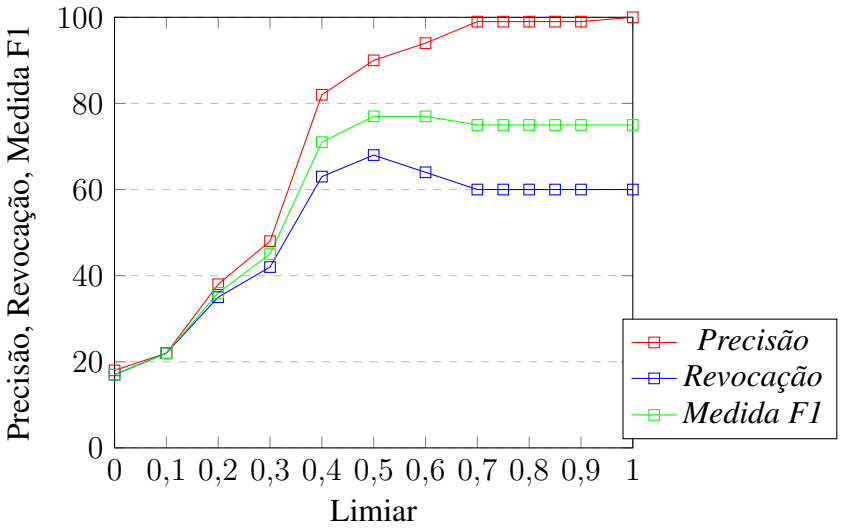
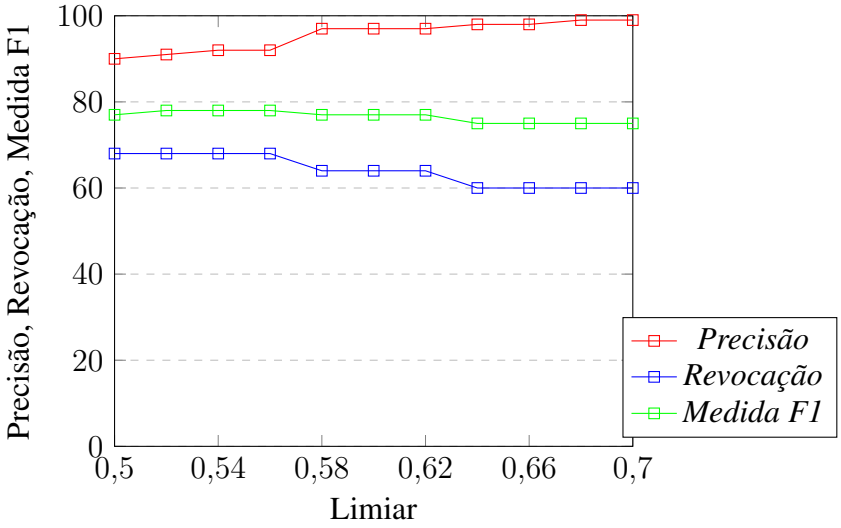


Figura 6.20 – SoftTFIDF considerando o campo SIAPE, coleção 1 - Refinamento



6.3.3.3 Coleção 2, matrícula SIAPE sendo desconsiderada

A Figura 6.21 mostra o desempenho da função SoftTFIDF sobre a segunda coleção de dados. Como mostra a Figura 6.21, essa função de similaridade tem um desempenho baixo para limiares de 0 a 0,4 e mostra uma melhora a partir de 0,5, obtendo um desempenho de 91% na Medida F1 com o limiar fixado em 0,6.

Foram rodados novos testes variando o limiar entre 0,5 e 0,7, intervalo onde a função mostrou maior potencial. A Figura 6.24 mostra o desempenho da função para esses experimentos que acabaram confirmando que o limiar fixado em 0,6 é o melhor valor para essa função.

Figura 6.21 – SoftTFIDF desconsiderando o campo SIAPE, coleção 2

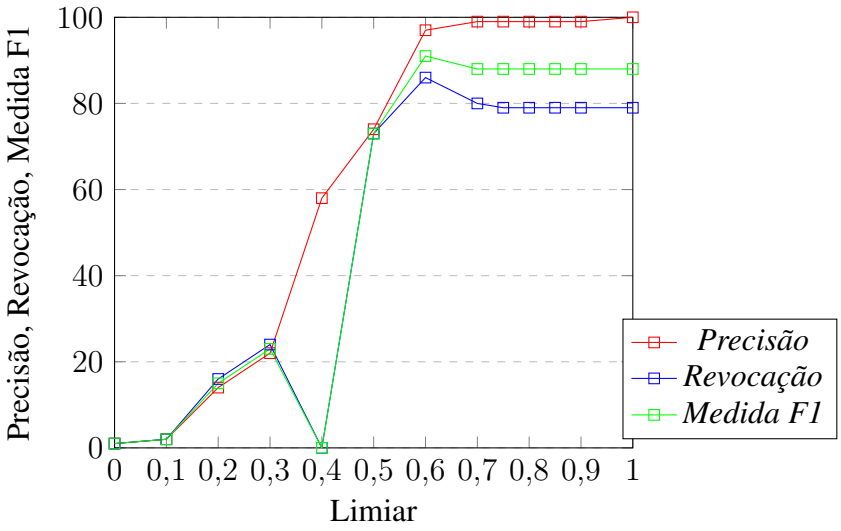
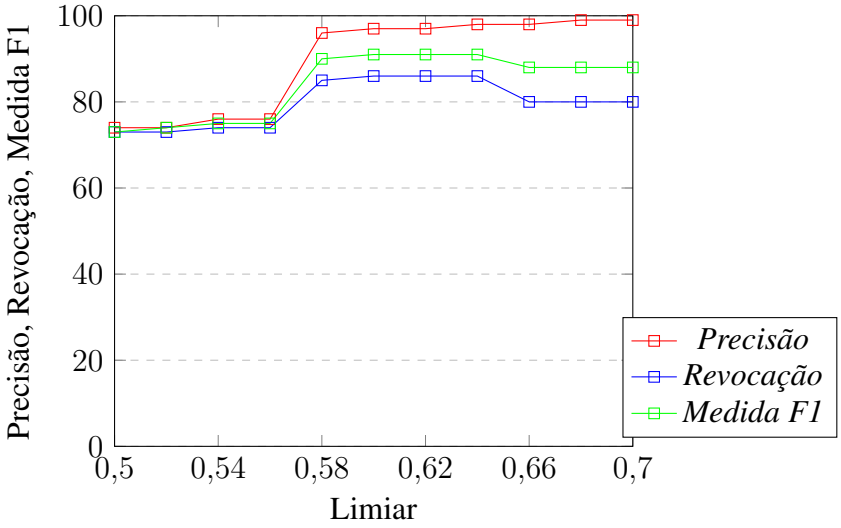


Figura 6.22 – SoftTFIDF desconsiderando o campo SIAPE, coleção 2 - Refinamento



6.3.3.4 Coleção 2, matrícula SIAPE sendo desconsiderada

A Figura 6.23 mostra o desempenho da função SoftTFIDF sobre a segunda coleção de dados, agora considerando o atributo SIAPE das *records*. Como mostra a Figura 6.23, essa função de similaridade tem um desempenho baixo para limiares de 0 a 0,3 e mostra melhor desempenho para valores a partir de 0,5.

Foram rodados novos testes variando o limiar entre 0,5 e 0,7, intervalo de melhor desempenho da função. A Figura 6.24 mostra o desempenho da função para esses testes e, de acordo com os resultados apresentados, com 78% de desempenho na Medida F1, o limiar 0,64 se mostra como o melhor valor para essa função.

Figura 6.23 – SoftTFIDF considerando o campo SIAPE, coleção 2

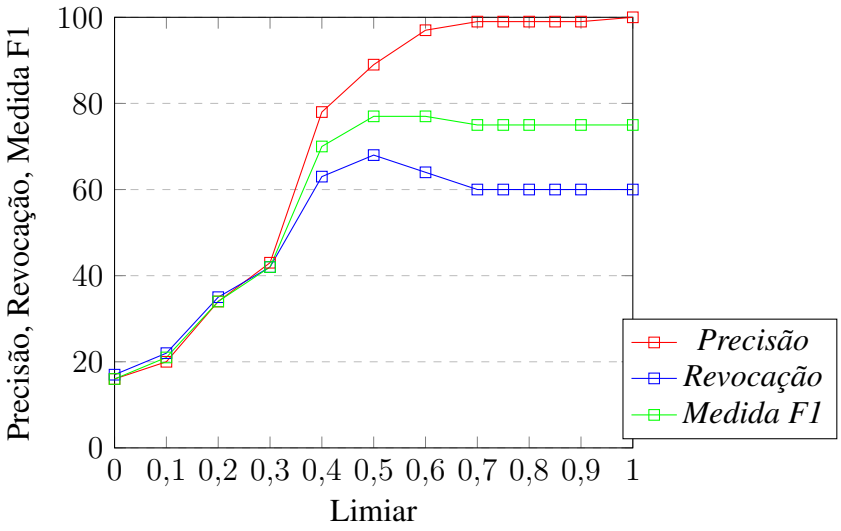
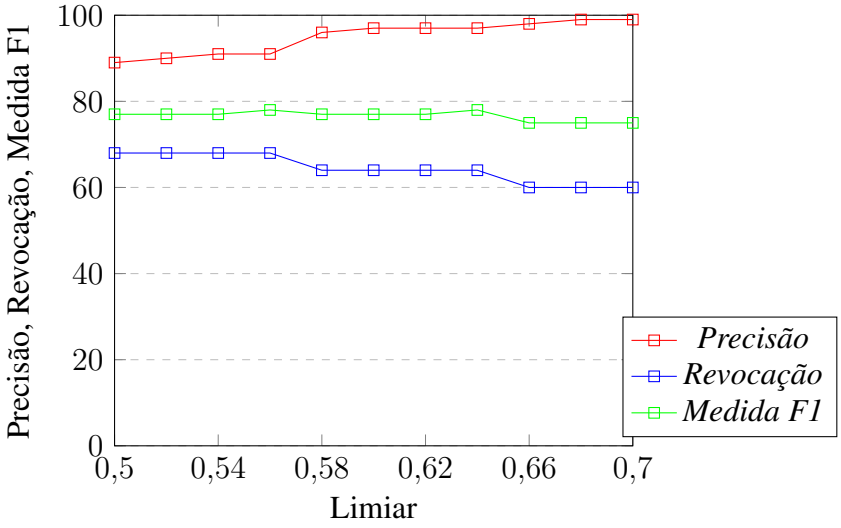


Figura 6.24 – SoftTFIDF considerando o campo SIAPE, coleção 2 - Refinamento



6.3.3.5 Discussão dos Resultados

Para a Coleção 1, com o atributo SIAPE sendo desconsiderado, o melhor limiar foi o de valor 0,64, que atribuiu corretamente uma autoridade para 1071 *records* em um total de 108 atribuições em uma coleção que contava com 1240 registros, totalizando uma precisão de 98% e 86% de revocação. Quando o SIAPE passou a ser considerado, o melhor limiar foi o de valor 0,56, que atribuiu corretamente 849 entidades em um total de 918 atribuições, totalizando uma precisão de 92% e 68% de revocação.

Para a Coleção 2, com o SIAPE não sendo considerado na função de *match*, o melhor limiar foi o de valor 0,60, que atribuiu corretamente 1067 entidades às *records*, em um total de 1095 atribuições, obtendo uma precisão de 97% e uma revocação de 86%. Quando o SIAPE foi considerado, o melhor limiar encontrado foi o de valor 0,64, que

atribuiu corretamente 805 entidades em um total de 822 atribuições, atingindo 97% de precisão e apenas 64% de revocação.

6.3.3.6 Análise dos Casos de Falha

Os problemas apresentados nos casos de falha das seções anteriores se repetiram e novos casos envolvendo problemas similares aos casos já apresentados surgiram. São casos híbridos dos apresentados nas funções anteriores, onde partes do nome das entidades é similar mas não necessariamente na mesma ordem. É o caso do servidor *Felipe Caron* que, apesar de existir uma autoridade com o nome idêntico ao identificado na *record*, foi conectado à autoridade *Felipe Raskin Cardon*, cujo primeiro nome é idêntico e possui grande similaridade no último sobrenome mas claramente não se trata da mesma pessoa.

Além disso, os problemas relacionados ao uso do SIAPE na função de *match*, apresentados nas seções anteriores, também acabaram se repetindo.

6.4 Experimento 2 - Escolha da Melhor Função de Similaridade

O objetivo deste experimento é definir qual a melhor função de similaridade para a tarefa de Resolução de Entidades no contexto deste trabalho, utilizando a base de dados definida na Seção 6.2.2.

A Tabela 6.1 resume os resultados obtidos na Seção 6.3 para cada uma das funções de similaridade. Os dados mostram que, independentemente da função de similaridade utilizada, considerar o atributo SIAPE das *records* na função de *match* diminui consideravelmente a revocação, visto que para muitas *records* foi atribuído um valor de SIAPE na etapa de Extração e Armazenamento de Entidades (Seção 4.1.2) que não condiz com a realidade, o que por consequência acaba diminuindo também o valor calculado para a Medida F1.

Analisando os dados apresentados na Tabela 6.1, é possível observar que todos os métodos obtiveram um desempenho similar considerando os melhores limiares nos contextos em que o SIAPE não é utilizado pela função de *match*. Apesar das funções apresentarem resultados muito próximos nas métricas avaliadas nos experimentos, o método *Cosine Similarity* se mostrou levemente superior às demais nas duas coleções de dados utilizadas nos experimentos, obtendo o melhor valor na Medida F1 e quase 100% de precisão nos dados da Coleção 1. Uma das razões que podem ter levado a esse de-

Tabela 6.1 – Melhores limiares para as funções de similaridade

Função de Similaridade	Coleção de Dados	Melhor Limiar	Siape sendo Considerado	Medida F1 (em %)	Precisão (em %)	Revocação (em %)
Jaro-Winkler	1	0,96	Não	91,43	98,60	85,24
Jaro-Winkler	1	0,92	Sim	80	95,52	68,97
Jaro-Winkler	2	0,96	Não	91,35	98,41	85,24
Jaro-Winkler	2	1	Sim	88,48	100	79,35
Cosine Similarity	**1**	**0,84**	**Não**	**92,28**	**99,16**	**86,29**
Cosine Similarity	1	0,84	Sim	78,37	99,25	64,75
Cosine Similarity	**2**	**0,84**	**Não**	**92,16**	**98,89**	**86,29**
Cosine Similarity	2	0,85	Sim	78,34	99,13	64,75
Soft TF-IDF	1	0,64	Não	92,01	98,43	86,37
Soft TF-IDF	1	0,56	Sim	78,68	92,46	68,46
Soft TF-IDF	2	0,60	Não	91,39	97,44	86,04
Soft TF-IDF	2	0,64	Sim	78,01	97,93	64,91

sempenho superior em relação às outras funções é a característica da função de separar os termos de uma *string* em tokens que, somando ao fato de que na etapa de Reconhecimento de Entidades diversos nomes de servidores tiveram seus nomes divididos, gerando mais de um registro em diversas Portarias, essa função consegue avaliar melhor esses subtermos de cada nome de servidor. Por esse motivo, se caracteriza como a melhor função de similaridade para função de *match* para ser utilizada no Algoritmo 1.

6.5 Experimento 3 - Verificar o Impacto de Entidades Sem Uma Autoridade

O objetivo deste experimento é avaliar o impacto que *records* sem uma autoridade correspondente no *authority file* geram na eficácia do Algoritmo 1 desenvolvido para a tarefa de Resolução de Entidades. Como explicado na Seção 6.2.2, os experimentos conduzidos neste trabalho utilizam duas coleções de *recods*: uma primeira coleção contendo apenas registros que, após verificação manual, possuem garantidamente uma autoridade correspondente no *authority file*, e uma segunda coleção contendo, além das *records* presentes na primeira coleção, outras *records* que, após verificação manual, foi constatado que se tratavam de estudantes ou empresas e, por isso, garantidamente não possuem uma autoridade correspondente no *authority file*.

A Tabela 6.2 mostra como as *records* sem uma entidade impactam no desempenho do Algoritmo 1 considerando apenas a função *Cosine Similarity*, definida no Experimento 6.4 como a melhor função de similaridade para os experimentos deste trabalho. É inte-

ressante observar que a coluna referente às *records* que foram corretamente conectadas a uma autoridade possui o mesmo valor em ambas as coleções e que, subtraindo o total de *records* sem autoridade do total de *records* que não obtiveram um *match* na segunda coleção, o valor é muito próximo ao total de *records* sem *match* na primeira coleção, o que é um excelente indicativo de que o método utilizado conecta uma *record* à uma autoridade apenas quando o *match* de fato existe, sofrendo pouco impacto quando esse vínculo não deveria existir pois, como mostra a Tabela 6.2, na Coleção 1 são 9 *records* conectadas incorretamente contra 22 na Coleção 2, sendo que a segunda possui 159 *records* a mais que a primeira, todas sem uma autoridade.

Tabela 6.2 – Impacto de *records* sem autoridade no desempenho da função *Cosine Similarity*

Coleção de Dados	**Total de *Records***	***Records* Sem Autoridade**	***Records* Linkadas Corretamente**	***Records* Linkadas Incorretamente**	***Records* Não Linkadas**
1	1240	0	1070	9	161
2	1399	159	1070	22	307

6.6 Experimento 4 - Tempo de Execução das Funções

Este experimento tem como objetivo apresentar o tempo de execução das funções de similaridade, fazendo uma análise sobre o *trade-off* entre eficiência e desempenho. A Tabela 6.3 apresenta, além do desempenho obtido nas métricas de avaliação, o tempo de execução das funções de similaridade utilizadas no Experimento 6.3 para avaliar todas as *records* nas Coleções 1 e 2 no contexto do melhor limiar encontrado para cada uma sem considerar o atributo SIAPE.

Tabela 6.3 – Tempo de execução das funções de similaridade para os melhores limiares

Função de Similaridade	**Coleção de Dados**	**Melhor Limiar**	**Tempo de Execução (em segundos)**	**Medida F1 (em %)**	**Precisão (em %)**	**Revocação (em %)**
Jaro-Winkler	1	0,96	524	91,43	98,60	85,24
Jaro-Winkler	2	0,96	692	91,35	98,41	85,24
Cosine Similarity	1	0,84	431	92,28	99,16	86,29
Cosine Similarity	2	0,84	649	92,16	98,89	86,29
Soft TF-IDF	1	0,64	1140	92,01	98,43	86,37
Soft TF-IDF	2	0,60	1510	91,39	97,44	86,04

Como mostra a Tabela 6.3, a função *Cosine Similarity* teve o menor tempo de

execução entre as três funções de similaridade utilizadas nos experimentos sobre as duas coleções de dados. Como visto, mesmo levando o menor tempo de execução, essa função foi a que apresentou os melhores resultados, obtendo quase 100% de precisão na Coleção 1.

A Tabela 6.4 apresenta o desempenho da função *Cosine Similarity* para limiares entre 0,6 e 1 considerando apenas a primeira coleção de dados. Como é possível observar observar, para limiares mais baixos, o tempo de execução da função diminui mais do que pela metade, pois quanto menor o limiar, menor a similaridade necessária entre duas *strings* para acontecer um *match*, porém o desempenho nas métricas avaliadas cai drasticamente. Todavia, o desempenho apresentado pelos limiares no intervalo em que essa função de similaridade tem o seu melhor desempenho é considerado satisfatório e justifica a diferença de tempo.

Tabela 6.4 – Tempo de execução da função *Cosine Similarity* para uma variação de limiares

Coleção de Dados	**Limiar**	**Tempo de Execução (em segundos)**	**Medida F1 (em %)**	**Precisão (em %)**	**Revocação (em %)**
1	1	832	35,32	100	21,45
1	0,9	476	89	99,97	80,32
1	0,84	431	92,28	99,16	82,29
1	0,80	410	87,90	92,63	82,17
1	0,70	277	63,62	64,35	62,90
1	0,60	160	40	40,19	4

6.7 Experimento 5 - Melhor *Match* x Melhor Limiar

Este experimento tem como objetivo avaliar a diferença de desempenho entre, no momento da busca de uma *record* por uma autoridade, usar o primeiro *match* considerando o melhor limiar de cada função e usar o melhor *match* possível para cada *record*, desconsiderando qualquer limiar.

A Tabela 6.5 apresenta o desempenho das funções de similaridade nas Coleções 1 e 2 sem considerar o SIAPE, selecionando o melhor *match* para cada *record*. Os resultados apresentados mostram que a precisão diminui em todos os métodos sobre a Coleção 2, visto que essa coleção possui *records* que não deveriam obter um *match*, o que num cenário mais amplo em uma base de dados maior é algo mais comum de acontecer, o que diminuiria ainda mais a precisão. Por outro lado, todos os métodos desempenham com

uma Medida F1 de aproximadamente 95% na coleção 1. O maior problema acaba sendo no tempo de execução, pois todas as *records* precisam varrer completamente o *authority file* para garantir o melhor *match*.

Tabela 6.5 – Melhor *match* encontrado para cada *record*

Função de Similaridade	Coleção de Dados	Medida F1 (em %)	Precisão (em %)	Revocação (em %)	Tempo de Execução (em segundos)
Jaro-Winkler	1	94,19	94,19	94,19	1113
Jaro-Winkler	2	88,51	83,48	94,19	1156
Cosine Similarity	1	95,64	95,64	95,64	1086
Cosine Similarity	2	89,88	84,77	95,65	1113
Soft TF-IDF	1	95,64	95,64	95,64	3362
Soft TF-IDF	2	89,88	84,77	95,64	2728

A Tabela 6.6 mostra o desempenho das funções de similaridade, considerando a Medida F1 e o tempo execução, para o melhor limiar de cada uma e também para o melhor *match* possível. Na Coleção 1 o desempenho das funções utilizando o melhor *match* é em média 3% superior em relação ao melhor limiar, porém na Coleção 2 acontece o inverso, visto que a segunda coleção possui 159 *records* sem uma autoridade mas que acabam sendo associada à uma autoridade devido à característica do algoritmo de utilizar o melhor *match*, independente de qual seja o valor da similaridade. O tempo de execução na busca pelo melhor *match* chega a ser próximo de 200% mais demorado relação ao tempo considerando o melhor limiar no método *Soft TF-IDF*, e 151% mais demorado na função *Cosine Similarity* na coleção 1. Levando em conta a precisão inferior da função utilizando o melhor *match* na Coleção 2, utilizar o melhor limiar é a melhor alternativa para o Algoritmo proposto para a Resolução de Entidades.

Tabela 6.6 – Melhor limiar x Melhor *match*. Entre parênteses, a diferença entre os métodos

Função de Similaridade	Coleção de Dados	Medida F1 do Melhor Limiar (em %)	Medida F1 do Melhor *Match* (em %)	Tempo de Execução Melhor Limiar (em segundos)	Tempo de Execução Melhor *Match* (em segundos)
Jaro-Winkler	1	91,43	94,19 (+2,76)	524	1113 (+112%)
Jaro-Winkler	2	91,35	88,51 (-2,84)	692	1156 (+67%)
Cosine Similarity	1	92,28	95,64 (+3,35)	431	1086 (+151%)
Cosine Similarity	2	91,76	89,88 (-1,88)	649	1113 (+71%)
Soft TF-IDF	1	92,01	95,64 (+3,36)	1140	3362 (+194%)
Soft TF-IDF	2	91,39	89,88 (-1,51)	1510	2728 (+80%)

6.8 Considerações Finais

Neste capítulo, foram apresentados os cinco experimentos realizados durante o desenvolvimento desta alternativa para a etapa de Resolução de Entidades da abordagem ACERPI. O primeiro, o mais extenso e que serviu de base para os demais, buscando o melhor desempenho para cada uma das funções de similaridade apresentadas na Seção 2.3. O segundo avaliando a melhor função, utilizando os resultados obtidos no primeiro experimento. O terceiro, avaliando o impacto de diferentes tipos de dados e expectativas sobre a melhor função avaliada no segundo experimento. O quarto, avaliando o tempo de execução das funções. E por fim, o quinto experimento, apresentando a diferença entre definir um limiar e utilizar o melhor *match* na tentativa de conectar uma *record* a uma autoridade.

7 CONCLUSÃO

Neste trabalho foi apresentada uma alternativa para a etapa de resolução de entidades da abordagem ACERPI (SCHMITZ et al., 2021) mas que mantém o objetivo final da proposta original: uma base de dados para a pesquisas a respeito de servidores e Portarias. Ao final, essa alternativa utiliza uma coleção de autoridades definida como *authority file* onde cada entrada representa uma entidade única no mundo real e pode ser conectada a diversos registros criados na etapa de reconhecimento de entidades nomeadas na abordagem ACERPI originalmente, onde cada registro contém, entre outros dados, o nome de uma entidade presente em uma dada Portaria. A identificação de qual entrada no *authority file* representa cada registro se deu por meio de comparação de *strings*, abordagem que foi avaliada em registros criados a partir da análise de Portarias da Universidade Federal do Rio Grande do Sul. Esses experimentos permitiram definir a melhor maneira de comparar os dados dos registros com os dados das autoridades.

Pontos de melhoria e evolução foram identificados, podendo ser desenvolvidos em possíveis trabalhos futuros. Destacam-se:

- O uso de técnicas de blocagem (PAPADAKIS et al., 2020) para o ganho de performance na etapa de resolução de entidades, visando diminuir o número de comparações feitas por cada registro sobre as entradas do *authority file*.
- O uso de paralelismo na etapa de resolução de entidades, visto que não existe mais interferência ou limitações nas relações entre *records* e autoridades.
- A identificação de outros dados dos servidores que possam ser armazenados nos registros e posteriormente utilizados na função de *match*.
- A construção de uma interface que utilize não só os dados dos registros mas também os dados presentes no *authority file*, permitindo ao usuário pesquisas sobre diversos atributos

Por fim, a principal contribuição deste trabalho foi a melhora na eficácia e eficiência na etapa de resolução de entidades da abordagem original, além dos experimentos realizados que ficam como referência para trabalhos que envolvam problemas similares.

REFERÊNCIAS

anhaidgroup.ai. User Manual for py_stringmatching. 2021. Disponível em: <https://anhaidgroup.github.io/py_stringmatching/v0.4.x/index.html>.

AWS. O que é NoSQL? 2020. Disponível em: <https://aws.amazon.com/pt/nosql/>.

BILENKO, M. et al. Adaptive name matching in information integration. **IEEE Intelligent Systems**, v. 18, n. 5, p. 16–23, Sep. 2003. ISSN 1941-1294.

BRASIL. Lei nº 12.527, de 18 de novembro de 2011. regula o acesso a informações previsto no inciso xxxiii do art. 5º , no inciso ii do § 3º do art. 37 e no § 2º do art. 216 da constituição federal; altera a lei nº 8.112, de 11 de dezembro de 1990; revoga a lei nº 11.111, de 5 de maio de 2005, e dispositivos da lei nº 8.159, de 8 de janeiro de 1991; e dá outras providências. **Diário Oficial da República Federativa do Brasil**, Brasília, DF, 2011. ISSN 1677-7042. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2011/lei/l12527.htm>.

COHEN, W.; RAVIKUMAR, P.; FIENBERG, S. A comparison of string distance metrics for name-matching tasks. **IIWeb**, v. 2003, 05 2003.

DALEN-OSKAM, K. et al. Named entity recognition and resolution for literary studies. **Computational Linguistics in the Netherlands Journal**, v. 4, p. 121–136, 12 2014.

Data Community DC. Entity Resolution for Big Data. August 2013. Disponível em: <http://www.datacommunitydc.org/blog/2013/08/entity-resolution-for-big-data>.

DOZIER, C. et al. Named entity recognition and resolution in legal text. In: ____. **Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language**. Berlin, Heidelberg: Springer-Verlag, 2010. p. 27–43. ISBN 364212836X.

JARO, M. A. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. **Journal of the American Statistical Association**, [American Statistical Association, Taylor Francis, Ltd.], v. 84, n. 406, p. 414–420, 1989. ISSN 01621459. Disponível em: <http://www.jstor.org/stable/2289924>.

MARSHALL, C. What is named entity recognition (NER) and how can I use it? December 2019. Disponível em: <https://medium.com/mysuperai/what-is-named-entity-recognition-ner-and-how-can-i-use-it-2b68cf6f545d>.

MONGODB. The database for modern applications. 2020. Disponível em: <https://www.mongodb.com/>.

MONGODB. What is NoSQL. 2020. Disponível em: <https://www.mongodb.com/nosql-explained>.

Palmieri Lage, J. et al. Automatic generation of agents for collecting hidden web pages for data extraction. **Data Knowledge Engineering**, v. 49, n. 2, p. 177 – 196, 2004. ISSN 0169-023X. Web Information and Data Management. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0169023X03001769>.

PAPADAKIS, G. et al. Blocking and filtering techniques for entity resolution: A survey. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 53, n. 2, mar. 2020. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3377455>.

Portal da Transparência do Governo Federal. Consulta a Servidores do Governo Federal. 2021. Disponível em: <http://api.portaldatransparencia.gov.br/swagger-ui.html>.

QAISER, S.; ALI, R. Text mining: Use of tf-idf to examine the relevance of words to documents. **International Journal of Computer Applications**, v. 181, 07 2018.

SCHMITZ, C. et al. Acerpi: An approach for ordinances collection, information extraction and entity resolution. In: **Anais do XXXV Simpósio Brasileiro de Bancos de Dados, SBBD 2021, online, October 4 - October 8, 2021**. [S.l.]: SBC, 2021.

ScrapingHub. What is web scraping? 2020. Disponível em: <https://www.scrapinghub.com/what-is-web-scraping/>.

UFRGS. Consulta a Portarias geradas pela Reitoria da UFRGS. 2016. Disponível em: <https://www1.ufrgs.br/sistemas/sde/gerencia-documentos/index.php/publico/consultar/>.

WINKLER, W. String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. **Proceedings of the Section on Survey Research Methods**, p. 354–359, 01 1990.

ZhouYang Luo. A library implementing different string similarity and distance measures. 2021. Disponível em: <https://pypi.org/project/strsimpy/>.