Universidade de São Paulo - USP
Universidade Federal de São Carlos - UFSCar
Universidade Estadual Paulista – UNESP

Criação de um grande repositório público de Entidades Nomeadas Abreviadas extraídas de um Corpus Histórico do Português do Brasil

Kátia Tiemi Hirotsu
Rosely Sanches
Sandra Maria Aluísio

NILC-TR-08-15
Dezembro, 2008

Série de Relatórios do Núcleo Interinstitucional de Lingüística Computacional
NILC - ICMC-USP, Caixa Postal 668, 13560-970 São Carlos, SP, Brasil

Agradecimentos

A minha família pelo apoio durante a minha vida e principalmente durante os períodos mais difíceis.

Aos amigos que fiz durante a decorrer este curso, agradeço por toda ajuda durante essa etapa da minha vida.

E as minhas orientadoras que me ajudaram muito durante a realização deste trabalho.

Índice

Índice de Figuras

Resumo

Esta monografia trata de um tema importante na área de pesquisa de Lingüística Computacional ou Processamento de Língua Natural (PLN): **Entidades Nomeadas** (EN) que são definidas como nomes próprios de pessoas, organizações, locais, acontecimentos, coisas (objetos nomeados), obras (artefatos e construções humanas), conceitos abstratos, além de datas e valores. Informações sobre o contexto em que a entidade está presente são ignoradas. O objetivo do projeto de pesquisa é a criação de um repositório de acesso público com algumas informações sobre as EN abreviadas encontradas em um corpus de documentos históricos brasileiros do século XVI ao XIX. Os dados armazenados no repositório referem-se: a categoria da abreviatura, sua expansão, a oração da qual ela foi extraída, entre outros. Os textos do corpus são caracterizados por não apresentarem uma ortografia padrão, portanto existe uma grande diversidade na forma de escrita de uma mesma palavra. Existem outros obstáculos que tornam o trabalho com esse tipo de material mais complexo, porém são esses mesmos desafios que motivam este projeto: palavras abreviadas são consideradas um desafio para o processamento automático de corpora históricos, pois elas apresentam variação de grafia, são muito freqüentes, além de serem ambíguas, isto é, apresentarem vários significados possíveis como é o caso da abreviatura “a” que pode significar alteza, alvará, Amaro, Ana, anima, ano, anos, Antônio, arroba, arrobas, Assembléia, assinado, Atual, entre outros. A busca por EN abreviadas segue uma metodologia, que pode ser resumida como uma pesquisa por termos adjacentes à EN. Depois, os resultados obtidos são avaliados manualmente e apenas as EN abreviadas da categoria escolhida são eleitas para serem introduzidas em um banco de dados, cujas informações registradas são acessíveis a pesquisadores através de uma aplicação web. Este estudo está inserido no escopo do projeto *Dicionário Histórico do Português do Brasil* (DHPB), um projeto do programa Institutos do Milênio do CNPq, cujo principal objetivo é a criação de um dicionário de Português do Brasil referente aos mesmos documentos utilizados neste projeto.

1. Introdução

1.1 Contextualização e motivação

A principal área em que este trabalho está inserido é a Lingüística Computacional que, entre outros objetivos, visa auxiliar a área da Lingüística através do uso da computação. Mais especificamente, relaciona-se a subárea de Lingüística de Corpus, que trabalha com corpora eletrônicos de amostras da língua natural.

Este estudo pertence a um projeto mais amplo, denominado Dicionário Histórico do Português do Brasil (DHPB), cujo objetivo é a elaboração de um dicionário para auxiliar pesquisadores que trabalham com textos antigos brasileiros. O DHPB é um programa do Instituto Milênio, do CNPq, composto pelos grupos NILC[1] (Núcleo Interinstitucional de Lingüística Computacional – Instituto de Ciências Matemáticas e de Computação da USP de São Carlos), DL[2] (Departamento de Letras da Universidade Federal de São Carlos), entre outras instituições nacionais e internacionais.

O projeto DHPB manipula um corpus de aproximadamente 7,5 milhões de palavras de textos históricos escritos entre os séculos XVI e o começo do século XIX, compondo-se de cartas de jesuítas missionários, documentos de bandeirantes, reportagens de sertanistas e documentos da Inquisição. Todos esses documentos foram digitalizados, além de um dicionário de abreviaturas composto por abreviaturas dos séculos XVI ao século XIX [Flexor, 1991], para que o processamento de análise computacional pudesse ser realizado. Contudo, como descrito em Vale et al. [2008], o dicionário Flexor exibe muitas abreviações não encontradas no corpus do projeto DHPB.

Um dos desafios impostos às pessoas que trabalham com materiais semelhantes aos documentos do projeto DHPB é a ausência de uma ortografia para que as pessoas pudessem se basear durante a escrita. Dessa forma, as palavras não seguem um padrão e muitas vezes apresentam-se abreviadas, o que torna o trabalho de interpretação desses documentos mais complicado.

Além disso, a precariedade do sistema de comunicação entre as agências de imprensa no Brasil colonial, a presença de diferenças culturais entre as regiões e a grande distância entre as pessoas contribuíram para o princípio de uma diversidade na grafia, pois cada indivíduo escrevia de modo parecido com a pronúncia da palavra, e algumas vezes cada região possui um modo peculiar de pronunciar um vocábulo. Outro obstáculo encontrado é a presença de novos termos nos textos brasileiros, relacionados com a flora e a fauna brasileira, que não existiam na

[1] http://www.nilc.icmc.usp.br/nilc/index.html
[2] http://www.ufscar.br/~letras/index.php

Europa e por isso não se encontram no português europeu. Todos esses obstáculos são mais bem relatados em Aluísio et al. [2008].

Devido a essas complicações, a colaboração de pesquisadores de várias partes do Brasil e de Portugal é muito importante para a realização do projeto DHPB. Neste grupo estão lingüistas e cientistas da computação de onze universidades.

A expansão errada de uma abreviação pode prejudicar a compreensão de um texto e dificultar a criação de ferramentas de processamento de língua natural, já que elas tratam dos problemas relacionados com geração e compreensão automática de línguas naturais humanas.

A compreensão correta dos textos históricos é muito importante para que historiadores (ou outros pesquisadores que trabalham com esse tipo de documento) possam interpretar de maneira correta os acontecimentos do passado, além de identificarem corretamente as pessoas que participaram desse período da história do Brasil, por exemplo. Esse é um dos motivos pelos quais fazem da tarefa de expandir corretamente uma abreviação um trabalho muito importante e que requer muita atenção. É ainda mais importante quando a abreviação é uma Entidade Nomeada, ou seja, é um nome próprio de alguma pessoa, local, acontecimento, entre outros.

O resultado deste estudo visa ser uma ferramenta adicional a pesquisadores que se propõem a examinar material do passado do nosso país. A principal atividade para a construção dessa ferramenta é a formação de um banco de dados com informações sobre Entidades Nomeadas abreviadas do português histórico do Brasil, principalmente porque vai disponibilizar a expansão correta dessas abreviações. Outro propósito desse estudo é facilitar o trabalho dos pesquisadores disponibilizando as informações sobre EN através de uma aplicação web, que pode ser acessada pela internet através de um browser cliente, como por exemplo, a Internet Explorer ou o Firefox. Dessa maneira, as pessoas que manipulam esses documentos históricos poderão entender corretamente sobre quem ou o que os textos estão relacionados e assim a história do nosso país poderá ser mais bem esclarecida e quem sabe, conseqüentemente, se tornar um assunto de maior interesse da nossa população.

1.2 Objetivos do trabalho

Nesta monografia é proposta a criação de um banco de dados com algumas informações sobre Entidades Nomeada abreviadas recuperadas através de busca por expressões regulares em um conjunto de textos históricos brasileiros do século XVI ao início do século XIX, os mesmos do projeto DHPB. Este banco de dados deve ser gerado e disponibilizado pela internet através de uma aplicação web. Dessa forma, pesquisadores terão acesso a uma ferramenta que visa auxiliar a análise desses documentos.

1.3 Organização da monografia

Esta monografia está dividida em quatro capítulos. O primeiro indica os desafios para a interpretação das abreviações de Entidades Nomeadas do português histórico do Brasil e sua manipulação computacional que foram a motivação para este projeto de graduação. No segundo são apresentados os conceitos importantes para entender o conteúdo dessa monografia, além de expor estudos semelhantes. O Capítulo 3 esclarece quais são as etapas executadas, além dos resultados obtidos. No Capítulo 4 é estabelecida uma conclusão com base nos resultados obtidos durante o progresso do projeto.

2. Revisão bibliográfica

2.1 Conceitualização e terminologia

Reconhecimento de entidades nomeadas (REN) é uma atividade presente em várias áreas nas quais há informação armazenada. Os sistemas que fazem esse tipo de reconhecimento permitem a melhora do desempenho de motores de busca, sistemas de indexação de informação, entre outros benefícios.

Um corpus deve ser constituído por dados verdadeiros, transformados de algum modo para que o computador possa realizar a tarefa almejada e representar uma variedade da língua que se deseja estudar. Nesse projeto, a variedade desejada é o português que vigorava na época da colonização portuguesa no Brasil.

A área de Lingüística de Corpus trabalha com o processamento de uma grande quantidade de textos, por isso muitas ferramentas foram desenvolvidas para facilitar o trabalho em pesquisas dessa área. As mais comuns são: concordanciadores, etiquetadores, entre outras.

Para o desenvolvimento do repositório de Entidades Nomeadas Abreviadas, as ferramentas utilizadas foram os processadores de corpus UNITEX[3] e Philologic[4], e um dicionário de variantes de grafia do português histórico construído no âmbito do projeto DHPB [Giusti et. al., 2007].

O UNITEX é um software livre. Seu conceito surgiu no LADL (*Laboratoire d'Automatique Documentaire et Linguistique*), sob a direção de Maurice Gross. Com a sua utilização, a busca por entidades nomeadas no corpus do projeto se tornou simples, pois existe uma função que realiza essa tarefa de busca em corpus. Contudo, para que as necessidades do projeto DHPB fossem satisfeitas, houve a necessidade de personalizar este processador de corpus. O resultado dessa adaptação, realizado por um projeto de mestrado do ICMC [Candido,

[3] http://igm.univ-mlv.fr/~unitex/
[4] http://moodle.icmc.usp.br/philologic-milenio/

2008], gerou o UNITEX-MILENIO[5]. Através dessa modificação na ferramenta, todos os textos indispensáveis para por em prática o projeto estão no formato adequado para serem manipulados.

As variantes na grafia de Entidades Nomeadas foram obtidas pelas ferramentas Philologic e o dicionário de variantes citado acima. Este último foi criado com a metodologia SIACONF (Sistema de Apoio à Contagem de Freqüência em Corpus) proposta por Giusti et. al. [2007] que utiliza 43 regras de transformações para agrupar variantes sob uma mesma forma gráfica. A Figura 1 mostra o resultado de uma pesquisa por variações da palavra "cidade", via software Dicionário[6] que trabalha com o formalismo DELA usado pelo UNITEX para representar dicionários.

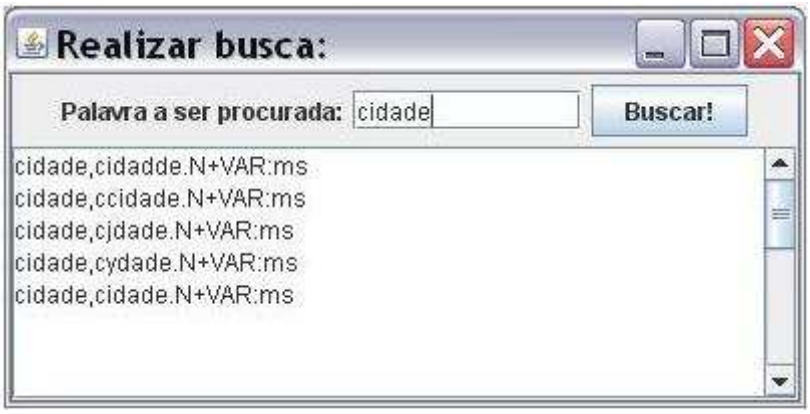


Figura 1: busca pela palavra "cidade" no Dicionário de Sinônimos do projeto DHPB

O primeiro campo do resultado obtido representa a palavra procurada, o segundo é uma provável variação dessa palavra encontrada em um corpus. A seguir estão as informações gramaticais (neste caso, "N" significa substantivo). Os códigos após a pontuação ":" é uma seqüência de informações flexionais, descrevem o gênero, o número, eventualmente o grau, o tempo e o modo das conjugações (":ms" significa masculino singular). Como este dicionário foi gerado automaticamente, todas as entradas ganharam a informação sobre flexões "ms".

A princípio, o uso de duas ferramentas que retornam variantes de vocábulos pode ser desnecessário. Entretanto, o Philologic, que usa um algoritmo de distância de edição, retorna uma grande quantidade de resultados cuja grafia é similar à palavra digitada pelo usuário, porém muitos deles não são variantes dessa palavra (por exemplo: a busca por "cidade" exibe entre seus resultados o termo "idade", que não é uma variação de "cidade"). Já o dicionário de variantes apresenta poucos resultados, contudo a vasta maioria (quase 100%) é alguma variação na grafia do vocábulo procurado. Dessa forma, a união desses dois instrumentos gera um benefício considerável, uma vez que retorna mais padrões para pesquisar.

[5] http://moodle.icmc.usp.br/milenio/
[6] http://www.nilc.icmc.usp.br/nilc/projects/unitex-pb/web/tools.html

Outro termo importante que posteriormente será citado é o HAREM[7], que é a avaliação conjunta de reconhecimento de Entidades Mencionadas. Chama-se "avaliação conjunta" porque é promovida uma comparação entre os sistemas dos participantes, com base em um conjunto de tarefas consensuais numa determinada área, usando um grupo de recursos em comum e uma métrica consensual.

O HAREM pretende medir o desempenho de sistemas de identificação e classificação de Entidades Mencionadas em textos, com base em coleções de documentos de Língua Portuguesa. Ele é realizado pela Linguateca[8] e este ano foi sua segunda edição.

Há um repositório público que contém exemplos de entidades nomeadas encontradas em textos contemporâneos, cujo nome é REPENTINO[9] (repositório de Entidades Nomeadas do português moderno) [Sarmento et. al., 2006]. Ele é dividido em várias categorias, cada qual tem as suas subdivisões. A construção desse repositório é coletiva e conta com colaboradores que fazem com que mais informações sejam armazenadas nele. Atualmente, existem 450.181 exemplos de entidades nomeadas nesse repositório. Sua principal função é auxiliar no desenvolvimento de sistemas de REN da língua portuguesa, pois contem exemplos que visam ajudar a construir regras para identificar as EN.

2.2 Trabalhos relacionados

Não foram encontrados sistemas de REN que trabalham com corpora de documentos históricos brasileiros, dessa forma esse estudo não teve muito material para se apoiar durante o seu desenvolvimento. Apesar disso, outros estudos relacionados ao REN foram pesquisados para dar um panorama de como é um sistema que reconhece uma entidade, mesmo que eles manipulem textos diferentes deste projeto. Os sistemas pesquisados mostram diferentes formas para resolver o problema de REN, portanto podem ser vistos como possíveis estratégias para melhorar este projeto, em um trabalho futuro. Nas próximas seções são apresentados alguns dos sistemas pesquisados.

2.2.1 Sistema Malinche

Inicialmente, o sistema Malinche [Solorio, 2007] foi desenvolvido para o Espanhol, mas provou-se que não seriam necessárias adaptações para que o mesmo sistema trabalhasse com REN em textos da língua portuguesa. Neste estudo, é empregada a estratégia de aprendizado de máquina para realizar a atividade de REN. Para isso, foi escolhido o algoritmo de aprendizagem

[7] http://www.linguateca.pt/HAREM/
[8] http://www.linguateca.pt
[9] http://www.linguateca.pt/repentino/

Support Vector Machine (SVM) (Vapnik, 1995; Stitson et al., 1996), mais especificamente a implementação do SVM inclusa no software livre WEKA[10].

O problema de REN é dividido em duas etapas, a primeira é chamada de delimitação das EN, e é responsável pela determinação de quais palavras podem ser consideradas EN. Já a segunda etapa denomina-se classificação de EN e encarrega-se da classificação da entidade em uma das seguintes classes: pessoa, organização, localização e variado.

Cada palavra possui cinco atributos para que o reconhecimento seja feito. Estes atributos são:

1. Informação ortográfica (se a palavra começa ou não com maiúscula);
2. Posição da palavra na sentença;
3. A própria palavra;
4. e 5. Correspondem ao contexto da palavra (conjunto de rótulos que indicam se a palavra está no começo de uma entidade, se pertence à entidade ou se não estão em nenhuma das duas opções anteriores, além de anotações PoS).

Estes atributos são mais bem descritos no trabalho de Solorio [2007].

Os resultados apresentados no artigo citado acima mostram que a utilização de algoritmos de aprendizado de máquina é uma boa estratégia para o reconhecimento de entidades. No HAREM de 2005, a medida P[11] para a tarefa de identificação é de 49,68% [Cardoso, 2007].

2.2.2 Sistema SIEMÊS

SIEMÊS[12] (Sistema de Identificação de Entidades Mencionadas com Estratégia Siamesa) [Sarmento, 2006] é um sistema híbrido apoiado por regras e por uma Base de dados com exemplos de entidades mencionadas já classificadas. É desenvolvido pela Linguateca e utiliza regras de forma e de semelhança para identificar e classificar as entidades mencionadas em texto livre. O cálculo de semelhanças do SIEMÊS é feito com base no REPENTINO. Sua medida P no HAREM de 2005 é 76,75% na tarefa de identificação [Cardoso, 2007].

2.2.3 Sistema Cortex[13]

Foi desenvolvido por Christian Nunes Aranha em sua tese de doutorado. O reconhecimento dos termos ocorre com ajuda de um autômato para identificar padrões de formação de entidades compostas com base num repertório de regras. São várias etapas até o término do reconhecimento das entidades. Quanto mais textos ele processa mais conhecimento lingüístico é acumulado, pois aprende a partir deles.

---

10 http://www.cs.waikato.ac.nz/ml/weka/

11 Mede a eficiência do sistema para delimitar corretamente as entidades.

12 http://poloclup.linguateca.pt/repentino/faq.html?#q14

13 www.cortex-intelligence.com

Ele tem quatro fontes de dados:

- Almanaque: lista de entidades de uma determinada categoria obtida de enciclopédica
- Metapalavras: lista de termos que aparecem nas vizinhanças das entidades
- Adivinhação: conjunto de termos que constituem as entidades mencionadas (por exemplo, Prof., Dr.)
- Léxico: armazena todo o conhecimento aprendido através de textos já processados pelo Cortex

No HAREM os valores das medidas foram: precisão (65,57%), abrangência (86,69%), medida f (0,7466) [Aranha, 2007].

3. Estado atual do trabalho

3.1 Projeto

A tarefa de construção de um repositório de EN abreviadas disponível pela internet pode ser divida em três grandes etapas: busca de EN abreviadas, formação do banco de dados que armazena as EN e implementação da aplicação web. A fase de pesquisa de EN é guiada pela mesma metodologia empregada na produção do REPENTINO, e a aplicação web segue o modelo do padrão MVC. Esses passos serão mais bem descritos no decorrer da monografia.

Para realizar a busca de Entidades Nomeadas foi usada a mesma metodologia aplicada na criação do Repositório REPENTINO. Ela consiste de seis etapas:

1. Escolher uma categoria para a qual se pretende pesquisar exemplos de entidades nomeadas.

2. Decidir uma estratégia apropriada para a pesquisa desses exemplos, que podem ser divididos em três formas:

    a) busca por palavras abreviadas colocadas ao lado esquerdo de certos tipos de Entidades Nomeadas (palavras que recebem o rótulo <INIT>).

    b) busca utilizando o contexto “local”, ou seja, palavras que indicam a presença de Entidades Nomeadas próximas a elas (exemplo: “localizado na XXX”, “próximo da XXX”, cuja probabilidade de que “XXX” indique uma Entidade Nomeada da categoria local).

    c) sufixos discriminatórios (exemplo: atualmente existem as partículas “Ltda.”, “S.A.”, indicando a presença de nomes de organizações próximos a essas partículas).

3. Construção de um padrão para ser usado em algum programa para a realização da busca no corpus.

4. Validação manual dos resultados obtidos. É necessário verificar se o resultado se enquadra na categoria pesquisada.

5. Inserção dos resultados validados na etapa anterior no repositório.

6. Criação de uma nova categoria ou subcategoria, aumentando o sistema de classificação taxonômico. Essa etapa é opcional.

A última etapa não é executada porque se tomou como modelo a taxonomia definida no HAREM. As categorias do HAREM de 2008[14] são:

- abstração (com 5 tipos): disciplina, estado, idéia, nome e outro;
- acontecimento (com 4 tipos): efeméride, evento, organizado e outro;
- coisa (com 5 tipos): classe, membro-classe, objeto, substancia e outro;
- local (com 4 tipos): físico, humano, virtual e outro;
    -físico (possui 7 subtipos): ilha, aguacurso, planeta, região, relevo, aguamassa e outro;
    - humano (possui 6 subtipos): rua, pais, divisão, região, construção e outro;
    - virtual (possui 4 subtipos): comsocial, sitio, obra e outro;
- obra (com 4 tipos): arte, plano, reproduzida e outro;
- organização (com 4 tipos): administração, empresa, instituição e outro;
- pessoa (com 8 tipos): cargo, grupocargo, grupoind, grupomembro, individual, membro, povo e outro;
- tempo (com 5 tipos): duração, freqüência, genérico, tempo_calend e outro;
    - tempo_calend (possui 4 subtipos): hora, intervalo, data e outro;
- valor (com 4 tipos): classificação, moeda, quantidade, outro; e
- outro.

O rótulo <INIT>, que foi citado anteriormente, é um rótulo para simbolizar um atributo da palavra. Entre os rótulos mais importantes neste projeto, estão:

- <ENT>: representa Entidade Nomeada;
- <INIT>: representa uma colocação presente ao lado esquerdo de alguns tipos de EN;
- <ABREV>: representa palavras abreviadas;

A referência Vale et al. [2008] possui uma explicação mais minuciosa sobre os atributos criados para o projeto DHPB.

A arquitetura da aplicação web que armazenará o resultado deste projeto é baseada no padrão MVC (Model-view-controller) porque separa a interface gráfica da navegação do seu comportamento. Este padrão apresenta três módulos implementados separadamente e que se comunicam para controlar os pedidos do usuário e suas respectivas respostas. Uma explicação específica sobre cada módulo é apresentada a seguir:

---

14 http://www.linguateca.pt/harem/

- Modelo: conteúdo da aplicação.

- Visão: funções da interface gráfica.

- Controlador: controla o fluxo de dados entre o modelo e a visão.

A comunicação entre esses módulos ocorre da seguinte maneira:

- Visão envia eventos para o controlador;

- Controlador modifica do estado do modelo;

- Modelo notifica a visão da mudança de estado;

- Visão busca dados atualizados no modelo.

A Figura 2 mostra a interação entre os três módulos do padrão MVC.

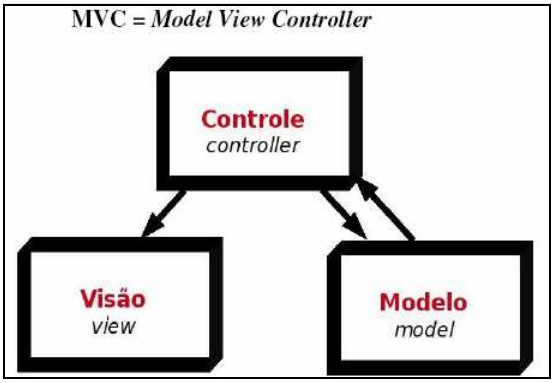


Figura 2: padrão MVC (modelo, visão, controle)

3.2 Descrição das atividades realizadas

3.2.1 Busca de Entidades Nomeadas Abreviadas

A categoria escolhida para este estudo foi “local”. Depois, escolheu-se qual a melhor estratégia para construir expressões regulares para REN. Esta monografia aborda as duas primeiras formas de construir padrões para pesquisa. A primeira maneira é a procura de EN adjacentes a termos com rótulo <INIT>, o que gerou 207 vocábulos. Este número é referente aos termos abreviados que começam com a letra “a”, “b” ou “c” do dicionário de Flexor [1991], pois o projeto DHPB está em andamento. O segundo método é mais abstrato, já que considera o contexto relacionado com a categoria desejada. Dessa forma foram consideradas palavras como: cidade, vila, vilarejo, porto, comarca, província, capitania, entre outros. No total são 188 padrões de busca (o apêndice A exibe uma lista de todas as expressões regulares buscadas no corpus).

É nessa etapa que a ferramenta UNITEX-MILENIO é muito útil, pois disponibiliza uma função de busca por expressões regulares no corpus do projeto, como mostra a ilustração da Figura 3.

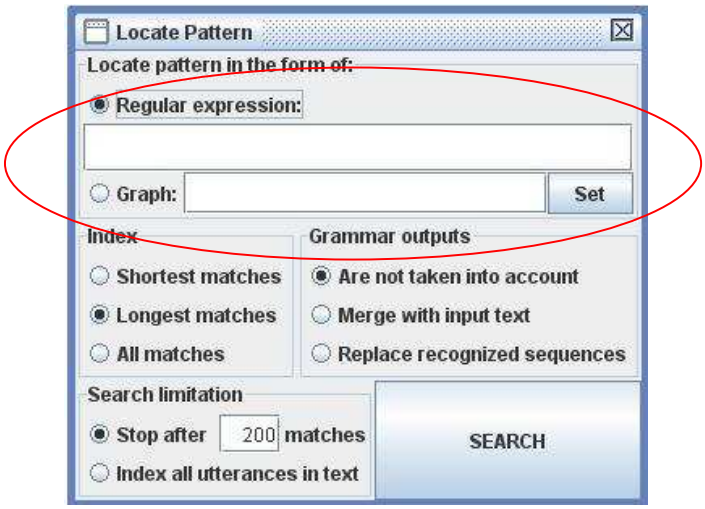


Figura 3: função de busca no corpus disponibilizada pelo UNITEX-MILENIO

Depois da realização da busca, muitos resultados negativos são retornados, ou seja, EN que não são abreviadas ou que não fazem parte da categoria escolhida. Para realizar a análise de quais dados devem ir ao repositório não há o emprego de ferramenta automática alguma, essa tarefa é manual, o que torna o processo demorado e minucioso. Por isso é necessário que alguém observe todos os resultados obtidos e verifique quais deles apresentam EN abreviadas da categoria desejada adjacentes ao termo procurado.

3.2.2 Construção do banco de dados

O gerenciador de banco de dados do projeto escolhido é o PostgreSQL[15], versão 8.3. A razão ocorre pelo fato de ser um software livre, um ótimo gerenciador de banco de dados e pela familiaridade adquirida em trabalhos anteriores de outras disciplinas do curso de graduação em Ciências da Computação.

Após a análise das características do projeto, concluiu-se que a criação de uma única tabela é suficiente para guardar as informações das EN. O nome dessa tabela é “dicionario" e compõe-se de quatro campos: “idAbreviacao”, “abreviacao”, “expansao”, “categoria” e “texto”. A primeira coluna é do tipo “integer”, pois se refere a uma identificação numérica da abreviação. Os demais campos são do tipo “character varying” (ou seja, cadeia de caracteres, cujo tamanho máximo é o número presente entre parênteses). A chave primária corresponde a coluna “idAbreviacao”. Cada tupla do banco de dados corresponde a um resultado obtido após a busca por um determinado padrão de EN. Para cada linha da tabela, o campo “abreviacao” diz respeito a Entidade Nomeada abreviada, “expansao” é a expansão dessa abreviatura, “categoria” refere-se a sua categoria e “texto” é uma frase retirada do documento em que a EN aparece.

15 http://www.postgresql.org

A Figura 4 foi extraída do banco de dados e visa demonstrar como os dados estão armazenados.

	idAbreviacao [PK] integer	abreviacao character varying	expansao character varying	categoria character varying	texto character varying(200)
1	1	N. Senhora da Conce	Nossa Senhora da Cc	local	"a do Beato Seraphim, na Ilha da Vargea. Aldea de N. Senhora da Conceição, na Ilha do Gambu. Ald"
2	2	s. Ant^o	Santo Antônio	local	"o q algãs se desfizeraõ mais sedo so a aldea de s. Ant^o jnda permaneçe mas desta gente toda n"
3	3	S. Sebastião	São Sebastião	local	"Arrayal de S. Sebastião, 20^o 19' 58". Arrayal de S. Caetano, 20^o 12' 58". Arrayal de Bom Jes"
4	4	S. Caetano	São Caetano	local	"Arrayal de S. Sebastião, 20^o 19' 58". Arrayal de S. Caetano, 20^o 12' 58". Arrayal de Bom Jes"
5	5	S. B.^to	São Bento	local	"E o tornej a entregar ao P.^e Prior do Conu^to de S. B.^to que mo apresentou E aqui asinou de"
6	6	Parn.^co	Pernambuco	local	"licadas reprezentaço'ens dos Off.^es da Cam.^a de Parn.^co, deprecando-lhe se dignasse aliviar"
7	7	Rio de Janr.^o	Rio de Janeiro	local	orda do Campo, Engenho, e Simão Pereyra cam.^o do Rio de Janr.^o Com outros de remotas parages
8	8	S. Vicente	São Vicente	local	"ontador proprietario da fazenda real da capitania de S. Vicente e S. Paulo. Em titulo de Freita"
9	9	S. Paulo	São Paulo	local	"ontador proprietario da fazenda real da capitania de S. Vicente e S. Paulo. Em titulo de Freita"
10	10	Sa. Bento	São Bento	local	"^ra foi dito que elle era procurador do mosteyro de Sa. Bento desta dita Cidade, e que a praja"
11	11	S. Paullo	São Paulo	local	"om companheiro, basta elle ser filho do bairo de S. Paullo do porvedor da junta q. D.^s tem o q"
12	12	S.^tos	Santos	local	"He a Carta escripta ao Juiz de Fora da V.^a de S.^tos de 23 de 7br.^o de 1806 registada no L.^ "
13	13	S. Paulo	São Paulo	local	"rdoso da Silveira, natural e cidadão da cidade de S. Paulo, filha de Salvador Cardoso de Almeid"
14	14	S. Paulo	São Paulo	local	"do na igreja do mosteiro de S. Bento da cidade de S. Paulo ao pé do altar de Nossa Senhora dos"
15	15	S. Paulo	São Paulo	local	"que se recitou no collegio de Jesus da cidade de S. Paulo deu o orador ao cadaver exposto no m"
16	16	S. Paulo	São Paulo	local	"mos Jesuitas, como consta do Archiv. da Cam. de S. Paulo L. n. 4, tit. 1658, fol. 3 et 24 vers."
17	17	S. Paulo	São Paulo	local	"Ruy Moschéra lhe havia feito Archiv. da Cam. de S. Paulo no liv. tit. 1585 até 1586, fol. 12 v."
18	18	S. Paulo	São Paulo	local	"iscentos e setenta e sete — Príncipe — (Cam. de S. Paulo liv. de Reg. tt.^o 1675 fl. 27 v.). Da"
19	19	Parn.^co	Pernambuco	local	"licadas reprezentaço'ens dos Off.^es da Cam.^a de Parn.^co, deprecando-lhe se dignasse aliviar"
20	20	Rio de Janr.^o	Rio de Janeiro	local	"orda do Campo, Engenho, e Simão Pereyra cam.^o do Rio de Janr.^o Com outros de remotas parages,"
21	21	S. Paulo	São Paulo	local	"ranampiacaba sobre a Vila de Ubatuba da Cap^a de S. Paulo, coiza de 2 leguas distante da Marinh"
22	22	S. Vicente	São Vicente	local	"3 da Comp^a de Jezus, q' p^la Costa da Cap^a de S. Vicente se achava este Metal em alguns Ribe"
23	23	são v^te	São Vicente	local	"o do dito an[o] nesta vila de são paulo capit^a de são v^te parte do brasil E nesta dita vila n"

Figura 4: exibição dos dados armazenados no banco de dados

O código sql utilizado para criar a tabela “dicionario” é exibido a seguir, e o banco de dados que armazena essa tabela chama-se “projGrad”:

```
CREATE TABLE dicionario
(
	"idAbreviacao" integer NOT NULL,
	abreviacao character varying(50) NOT NULL,
	expansao character varying(60) NOT NULL,
	categoria character varying(10) NOT NULL,
	texto character varying(200),
	CONSTRAINT dicionario_pkey PRIMARY KEY ("idAbreviacao")
)
WITH (OIDS=FALSE);
ALTER TABLE dicionario OWNER TO "admin";
```

3.2.3. Construção da aplicação web

A implementação da aplicação web é guiada pelo padrão MVC, e é por isso que o projeto apresenta uma classe Servlet (“DicionarioServlerController”), uma página JSP (“dicionario_lista”) e a classe JavaBean (“DicionarioBean”). Essa foi uma maneira encontrada para implementar os três módulos básicos do padrão.

O Servlet corresponde ao papel do controlador no padrão MVC, pois é ele quem decide qual deve ser a reação correspondente a uma determinada ação do usuário. A classe “DicionarioServlerController” (neste trabalho é a implementação do Servlet) possui os métodos “doGet” e “doPost”, que são os métodos de serviço. A requisição ocorre conforme a Figura 5.

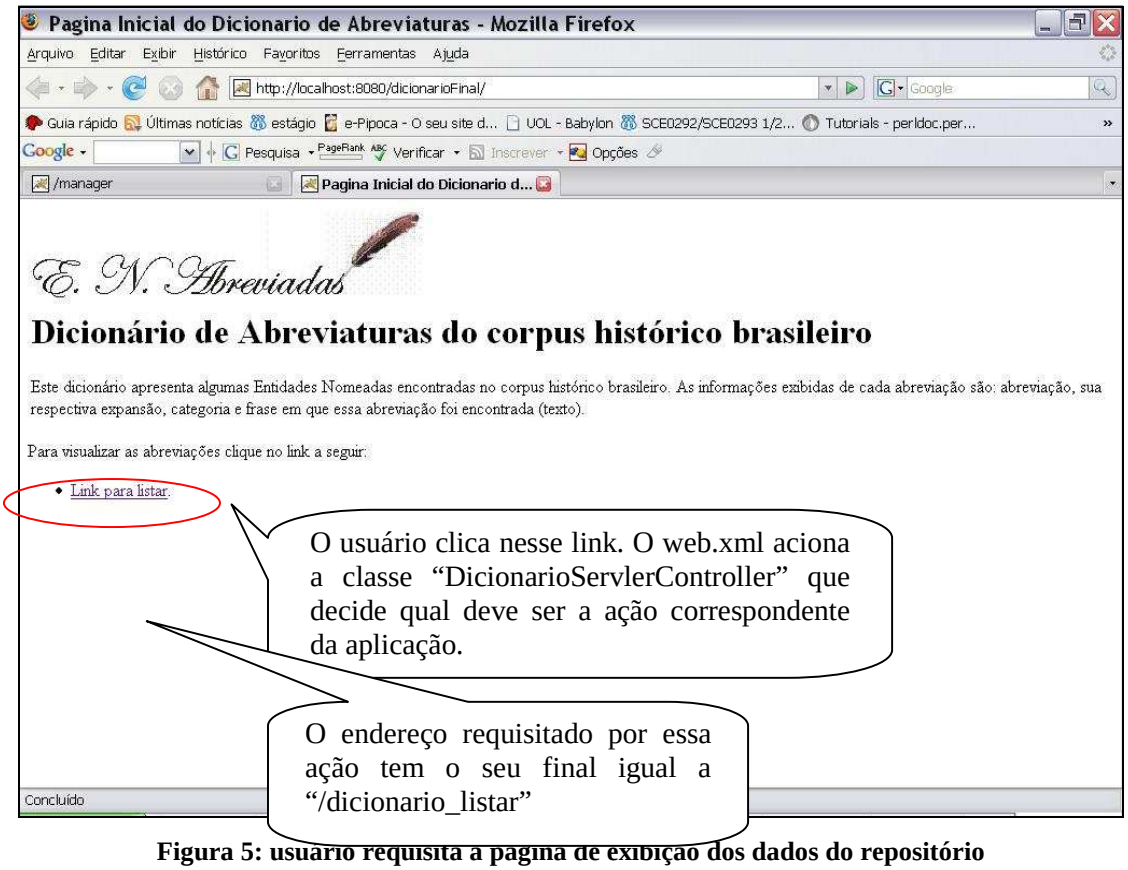


Figura 5: usuário requisita a página de exibição dos dados do repositório

Ao receber uma requisição (pode ser do tipo “get” ou do tipo “post”), o método “processRequest” é acionado e verifica se o usuário quer visualizar os dados do repositório. Caso ele queira, ou seja, o final do endereço requisitado é igual a “/dicionario_listar”, então um processo para acessar e exibir os dados do banco de dados é iniciado.

Este processo se inicia com a chamada ao método “listaDicionario” do Servlet. Este por sua vez cria uma instância da classe “DicionarioBO”, cuja função é intermediar a comunicação entre o controlador e a classe que faz a conexão no Postgre através do método “listarDicionario”.

A conexão com o banco de dados é executada através do construtor da classe “DicionarioDAO”. Os métodos desta classe são responsáveis por todas as manipulações necessárias no Postgre. Por enquanto a ação mais útil nesse momento é a exibição dos dados armazenados no banco de dados, que é desempenhada pelo método “listar”. A listagem dos valores dos campos da tabela “dicionario” precisa de uma instância do tipo “ArrayList”

(“listaAbrev”) e uma da classe “DicionarioBean” (“dicionarioBean”), porque depois de recuperar as informações da tabela, elas serão atribuídas aos atributos do “dicionarioBean”. Este objeto será adicionado a lista “listaAbrev”, que resulta em uma lista de objetos do tipo “dicionarioBean” que armazena os dados das EN abreviadas em seus atributos.

Com base no padrão MVC, o papel de cada uma das classes é:

- “dicionario_lista.jsp” é a visão;

- “DicionarioServlerController.java” é o controlador;

- “DicionarioBean.java” é o modelo;

A estrutura de diretórios da aplicação web está apresentada a seguir:

```
dicionario/
        |------ dicionario_lista.jsp
        |------ estilo.css
        |------ index.html
        |------ WEB-INF/
                |------ classes/
                |       |------ pacote/
                |               |------ controller
                |               |       |------ DicionarioServlerController.class
                |               |       |------ DicionarioServlerController.java
                |               |------ model
                |               |       |------ DicionarioBO.java
                |               |       |------ DicionarioBO.class
                |               |       |------ DicionarioBean.class
                |               |       |------ DicionarioBean.java
                |               |------ dao
                |                       |------ DicionarioDAO.class
                |                       |------ DicionarioDAO.java
                |------ lib/
                |       |------ postgresql-8.3-603.jdbc4.jar
                |------ web.xml
```

O acesso ao banco de dados para listar seus dados é feito por “usuario” com a senha “1234”. Esses dados estão diretamente no código-fonte de “DicionarioDAO.java” e não são requeridos aos usuários da aplicação.

Para testar a aplicação web utilizou-se o gerenciador de aplicação web Tomcat[16]. A sua escolha é baseada em algumas características: por ser um software livre, pela sua facilidade de uso, e a experiência adquirida em práticas anteriores. Para efetuar o “deploy” do código fonte da aplicação é necessário compactar os arquivos no formato “zip” e depois mudar a extensão para “war”.

[16] http://tomcat.apache.org

3.3 Resultados obtidos

A métrica utilizada para avaliar os resultados obtidos nesse estudo é a precisão. Ela mede a qualidade das respostas conseguidas. A fórmula adotada para obter essa métrica é a mesma que a utilizada pelo segundo HAREM, que é:

Precisão = Nº de EM corretamente classificadas / Nº de EM classificadas pelo sistema

Porém ao invés de número de Entidades Mencionadas (EM) deve-se considerar número de Entidades Nomeadas (EN).

A busca de EN abreviadas utilizando termos rotulados com o rótulo <INIT> retornou 1186 resultados diferentes, porém destes apenas 163 se encaixavam nas características desejadas, o que leva a uma precisão de 13,74%. Já o método utilizando palavras cujo contexto se relacione com a categoria escolhida retornou 7542 resultados, dos quais 213 estão corretas. Portanto, a taxa de precisão é de 2,82%.

O cálculo da métrica cobertura não foi realizado devido ao fato do número total de EN da categoria “local” nos documentos processados não ser conhecida, dessa forma essa métrica não pode ser medida.

Desse modo, pode-se dizer que a busca por palavras com o rótulo <INIT> se mostrou uma estratégia mais precisa do que a utilização pelo contexto, pelo menos considerando a categoria “local”. Contudo, a união das duas estratégias é a melhor opção, já que elas retornam diferentes exemplos de EN abreviadas. Se apenas a taxa de precisão for observada, pode-se concluir que o método utilizado nesse projeto foi ruim, porém o objetivo principal não é obter uma alta taxa de precisão e sim coletar o maior número possível de EN abreviadas da categoria escolhida para que elas sejam armazenadas no repositório com sua respectiva expansão correta.

O resultado final obtido depois de passar por todas as etapas pode ser visto através da aplicação web (Figura 6).

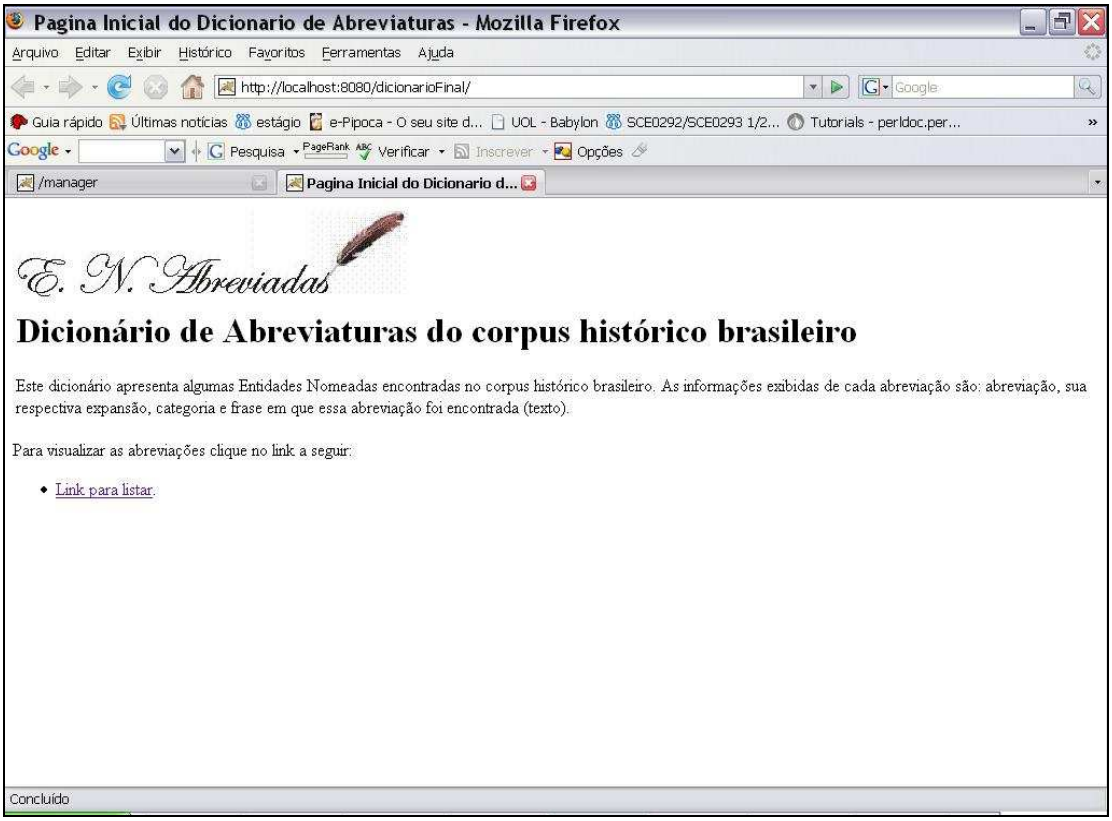
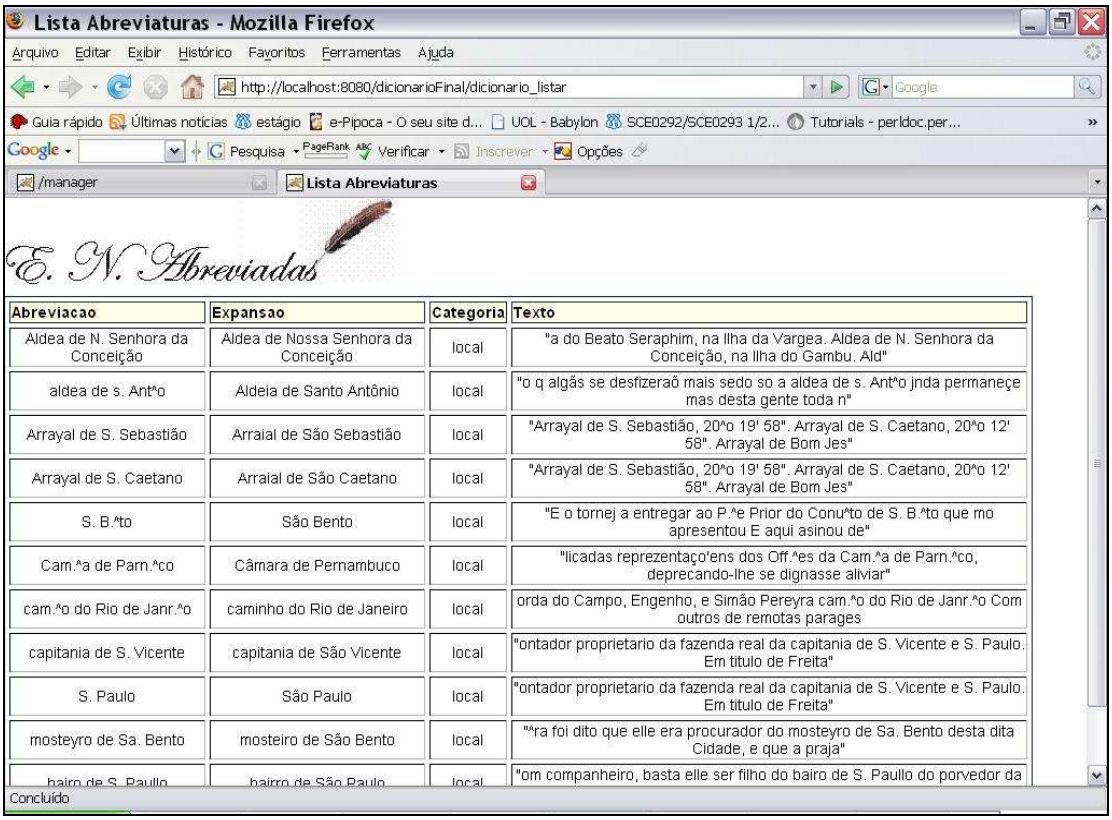


Figura 6: página inicial da aplicação web deste projeto

A apresentação das EN abreviadas é mostrada na Figura 7.



Abreviacao	Expansao	Categoria	Texto
Aldea de N. Senhora da Conceição	Aldea de Nossa Senhora da Conceição	local	"a do Beato Seraphim, na Ilha da Vargea. Aldea de N. Senhora da Conceição, na Ilha do Gambu. Ald"
aldea de s. Ant^o	Aldeia de Santo Antônio	local	"o q algãs se desfizeraõ mais sedo so a aldea de s. Ant^o jnda permaneçe mas desta gente toda n"
Arrayal de S. Sebastião	Arraial de São Sebastião	local	"Arrayal de S. Sebastião, 20^o 19' 58". Arrayal de S. Caetano, 20^o 12' 58". Arrayal de Bom Jes"
Arrayal de S. Caetano	Arraial de São Caetano	local	"Arrayal de S. Sebastião, 20^o 19' 58". Arrayal de S. Caetano, 20^o 12' 58". Arrayal de Bom Jes"
S. B.^to	São Bento	local	"E o tornej a entregar ao P.^e Prior do Conu^to de S. B.^to que mo apresentou E aqui asinou de"
Cam.^a de Parn.^co	Câmara de Pernambuco	local	"licadas reprezentaçoens dos Off.^es da Cam.^a de Parn.^co, deprecando-lhe se dignasse aliviar"
cam.^o do Rio de Janr.^o	caminho do Rio de Janeiro	local	orda do Campo, Engenho, e Simão Pereyra cam.^o do Rio de Janr.^o Com outros de remotas parages
capitania de S. Vicente	capitania de São Vicente	local	"ontador proprietario da fazenda real da capitania de S. Vicente e S. Paulo. Em titulo de Freita"
S. Paulo	São Paulo	local	"ontador proprietario da fazenda real da capitania de S. Vicente e S. Paulo. Em titulo de Freita"
mosteyro de Sa. Bento	mosteiro de São Bento	local	"^ra foi dito que elle era procurador do mosteyro de Sa. Bento desta dita Cidade, e que a praja"
bairo de S. Paullo	bairro de São Paulo	local	"om companheiro, basta elle ser filho do bairo de S. Paullo do porvedor da

Figura 7: página retornada após o pedido de exibição das EN abreviadas

4. Conclusões e trabalhos futuros

A atividade de REN utilizando algum algoritmo de aprendizado de máquina mostra resultados com boa precisão, e sua implementação pode ser considerada uma boa sugestão durante a continuidade do projeto. Outras propostas são: a pesquisa de EN abreviadas das demais categorias e melhoramento da aplicação web para que ela se torne cada vez mais uma ferramenta fácil de usar e útil (como por exemplo, a aplicação de conceitos de HCI - Human–computer interaction).

Este estudo não só trouxe conhecimentos técnicos da área de computação, mas também ofereceu uma visão de como uma ferramenta computacional realmente pode ajudar as pessoas. Pois durante a graduação, o principal objetivo dos trabalhos é gerar conhecimento e muitas vezes é difícil encontrar uma situação em que esse trabalho é útil para alguém.

Desenvolver um projeto em uma área, que não foi diretamente abordada no decorrer da graduação, produz uma grande quantidade de conhecimento. Nesse caso, muita informação é gerada com relação à história do Brasil, sobre a área da Lingüística Computacional, além das soluções para os problemas encontrados durante o desenvolvimento desse projeto (como a instalação do programa Postgre em Windows com sistema de arquivos FAT32).

Referências

ALUÍSIO, S. M. E CANDIDO JR., A. Córpus históricos para a construção de dicionários. A ser publicado no Livro Introdução à Lingüística Computacional: 1ª Escola Brasileira, 2008.

ARANHA, C. N. O Cortex e a sua participação no HAREM, 2007. Disponível no site: http://acdc.linguateca.pt/aval_conjunta/LivroHAREM/Cap09-SantosCardoso2007-Aranha.pdf

CANDIDO JR., A. Criação de um ambiente para o processamento de córpus de Português Histórico. Dissertação de Mestrado do ICMC-USP, 143 p., 2008.

CARDOSO, D. S. N. HAREM, a primeira avaliação conjunta de sistemas de reconhecimento de entidades mencionadas para português, 2007. Disponível em: http://xldb.di.fc.ul.pt/linguateca/prefacio.pdf

FLEXOR, MARIA HELENA M. O. Abreviaturas - Manuscritos dos Séculos XVI Ao XIX. 2nd ed. São Paulo: UNESP. 468 p., 1991.

GIUSTI, R.; CANDIDO JR, A.; MUNIZ, M. C. M.; CUCATTO, L. A.; ALUÍSIO, S. M. Automatic detection of spelling variation in historical corpus: An application to build a Brazilian Portuguese spelling variants dictionary. In: **Corpus Linguistics**, 2007, Londres. Corpus Linguistics, 2007.

SARMENTO, L. (2006). SIEMÊS: a named entity recognizer for Portuguese, In: Proceedings of PROPOR´2006, LNCS Volume 3960/2006, p. 90-99, 2006.

SARMENTO, L., PINTO, A. S., CABRAL, L. (2006). REPENTINO: A wide-scope gazetteer for Entity Recognition in Portuguese, In: Proceedings of PROPOR´2006, LNCS Volume 3960/2006, p. 31-40, 2006.

SOLORIO, T. MALINCHE: A NER system for Portuguese that reuses knowledge from Spanish. Reconhecimento de entidades mencionadas em português: Documentação e actas do HAREM, a primeira avaliação conjunta na área, Capítulo 10, p. 123–136, 2007. Disponível em http://acdc.linguateca.pt/aval_conjunta/LivroHAREM/Cap10-SantosCardoso2007-Solorio.pdf

STITSON, M.O., WESTON, J.A.E., GAMMERMAN, A., VOVK, V., VAPNIK, V. Theory of Support Vector Machines (Dept. Comp. Sci. Tech. Rep. CSD-TR-96-17; London: Univ. London Royal Holloway College), 1996

VALE, O.; CANDIDO Jr. A.; MUNIZ, M. ;BENGTSON, C.; CUCATTO, L.; ALMEIDA, G.;BATISTA, A.;PARREIRA, M.C.; BIDERMAN, M.T. ALUÍSIO, S. Building a large dictionary of abbreviations for named entity recognition in Portuguese historical corpora. In the Proceedings of the LREC 2008 Workshop on Language Technology for Cultural Heritage Data (LaTeCH 2008), p. 1-10.

VAPNIK, V. N. The Nature of Statistical Learning Theory (New York: Springer), 1995.

Wikipédia: http://pt.wikipedia.org/wiki/Ling%C3%BC%C3%ADstica_de_Corpus

Acessado em: 19/05/08

Apêndices

A. Palavras cujo contexto indica local e foram utilizadas para REN abreviada

Padrão original	Variações
Província	aprovincai áProvincia
Colônia	coionia Collonia colona COLONIA colonias colonie colonnia
Lugar	elugar iugar lagar llugar logar Lucar lugor LVGAR Olugar ugar
Localidade	
País	país paîs paíz
Província	Provinci Provincia Provinçia Província Provinciae provinciæ
	PROVINCIAS Provincies Provinciis provinsias Provinvia provncia Proficia Prouincia prouincias prouinsia province
Cidade	aCidade Ccidade Cdade cidad Cidadde Cidade;de CIDADES cjdade ClDADE cydade eCidade Sidade vidade
Fazenda	aFazenda efazenda facenda fasenda fazanda fazemda fazendas
	fazendaz fazendb fazienda fezenda ffazenda
Aldeia	Aldea ALDEIA Aldeias Aldeya aldia alldeia
Arraial	Arraal arraia arraias Arrayal
Bairro	Bairo Bairros bayrro
Mosteiro	Domosteiro Esteiro moesteiro Mosteiro Mosteirono Mosteiros Mosteitro Mosteyro Mosteyros mosteyroz Nosteyro Omosteiro
Perto	perto

	pertto
Vila	uiLa
	ila
	via
	viela
	viila
	vilas
	villa
	vjla
	Vjlla
	vlla
	vyla
	vylla
Sítio	sítios
	sytio
	sityo
	sittio
	sitjo
	sitiou
	sítios
	sitio
	ositio
	esitio
	citio
Capitania	acapitana
	acapitania
	acappitania
	capelania
	capiania
	capiiania
	capillania
	capiranga
	capitai
	capitaina
	capitais
	capitaja
	capitama
	capitana
	capitanas
	capitanea

Comarca	comarca
	comarcas
	commarca
	ecomarca

Porto	aporto
	eporto
	oporto
	poito
	porti
	portos
	portoz
	portto

B. Termos com rótulo <INIT> utilizadas para REN abreviadas

ancoradr^o
antg^o
qrce bispd^o
arcebisp^o
arcebp^do
arcebp^o
arch
asentam^to
assentam^o
assentam^to
assentam^tos
ãtiga
acid^e
aladr^a
aoR^o
ap^ra
av^a
b
b^os
b.
bx^a
bx^o
barr^as
bat^a
bat^ia
bater^a
batr^a
batt^a
c
cab.
campam^to
cabc^a
cabec^ra
cabeçr^a
cabecr^a
caber^a
cabis^ra

cabisr^a
cabacr^as
cabcr^as
cabecr^s
cachor^a
cachr^a
cacr^a
cax^ra
caxr^a
cxr^a
cad^a
c^a
cam.
cam^a
cam^r
cam^ra
camr^a
car^a
cm^a
cm^ra
cam^ras
camr^as
camr^s
c^o
cam^o
cam^os
cam^s
cãpo
cãpos
cap.
cap^a
cap^la
capl^a
capp^a
capp^la
cp^la
cap^as

cap^las
capell^a
cap^es
c^al
cap^al
cap^l
cap^tal
capit^l
cap^a
cap^ia
cap^na
cap^nia
cap^tania
cap^tia
cap^tna
capin^a
capit^a
capit^ia
capit^nia
capitan^a
capitn^a
capn^a
capn^ia
capni^a
capnn^a
capp^a
capp^ia
capp^na
capp^nia
capp^ta
capp^tta
cappit^a
cappitt^a
cappt^a
capt^a
capt^ia
capt^na

capt^nia
capti^a
captnn^a
ccapp^a
cp^ta
cpt^a
cap^as
cap^nias
cap^tas
capt^as
car^tro
cemit^o
cemit^rio
cemit^ro
cemitr^o
cemitr^os
cimitr^os
chra
can^la
canchr^a
ch
ch^a+ch^ra
chan^a
chan^ca
chan^ça
chan^ra
chanc^a
chanc^ria
chancel^ria
chancell^a
chancell^ra
chancellar^a
chancellr^a
chancelr^a
chancr^a
chans^a
chans^ra
chans^ria
chansselr^a
chanxar^a

chr^a
chan^cãs
chancelr^as
c^d
cd^e
ci^de
ci^e
cid^de
cid^e
çid^e
cida^e
col^o
colg^o
coll^o
coll^os
con^co
conc^o
comcerva^tra
cont^a
cont^ra
contad^a
contad^ra
contadr^a
contadr^ia
contd^a
contdr^a
cont^e
cont^te
conten^e
contin^e
contin^te
contint^e
comb^to
comu^to
comv^to
con^to
conu^o
conu^to
conv^o
conv^to

convt^o
cov^to
cõvento
comv^tos
conu^tos
conv^tos
con^lo
cert^es
cert^s
cruzr^o
crvz^ro
conv^ti
v.^a

C. Código-fonte da aplicação web criada

a) DicionarioBean.java

```
package pacote.model;

public class DicionarioBean{

	int idAbreviacao;
	String abreviacao;
	String expansao;
	String categoria;
	String texto;

	public DicionarioBean(){}

	public  DicionarioBean(int  idAbreviacao,  String  abreviacao,  String  expansao,
String categoria, String texto){
		super();
		this.idAbreviacao = idAbreviacao;
		this.abreviacao = abreviacao;
		this.expansao = expansao;
		this.categoria = categoria;
		this.texto = texto;
	}

	/*---------GET´s-----------*/
	public int getIdAbreviacao(){
		return idAbreviacao;
	}

	public String getAbreviacao(){
		return abreviacao;
	}

	public String getExpansao(){
		return expansao;
	}

	public String getCategoria(){
		return categoria;
	}

	public String getTexto(){
		return texto;
	}

	/*---------SET´s-----------*/
	public void setIdAbreviacao(int idAbreviacao){
		this.idAbreviacao = idAbreviacao;
	}

	public void setAbreviacao(String abreviacao){
		this.abreviacao = abreviacao;
	}

	public void setExpansao(String expansao){
		this.expansao = expansao;
	}

	public void setCategoria(String categoria){
		this.categoria = categoria;
	}

	public void setTexto(String texto){
		this.texto = texto;
	}
}
```

b) DicionarioBO.java

```
package pacote.model;

import pacote.dao.DicionarioDAO;

import java.io.IOException;
import java.rmi.RemoteException;
import java.util.Iterator;
import java.util.List;

public class DicionarioBO{
        DicionarioDAO dicionarioDAO;

        public List listarDicionario(){
                dicionarioDAO = new DicionarioDAO();
                return dicionarioDAO.listar();
        }
}
```

c) DicionarioDAO.java

```
package pacote.dao;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;
import pacote.model.DicionarioBean;

public class DicionarioDAO{
    private Connection conn;
    private String jdbcURL = "jdbc:postgresql://localhost:5432/projGrad";
    private String nome = "usuario";
    private String senha = "1234";
    private PreparedStatement listar;

        public DicionarioDAO(){
                DicionarioBean dicionariobean = new DicionarioBean();
                try{
                        Class.forName("org.postgresql.Driver");
                        conn = DriverManager.getConnection(jdbcURL, nome, senha);
                        listar = conn.prepareStatement("SELECT * FROM dicionario;");
                }
                catch(Exception e){}
        }

        public List listar(){
                List listaAbrev = new ArrayList();
                try{
                        ResultSet rs = listar.executeQuery();
                        while (rs.next()) {
                                DicionarioBean dicionarioBean = new DicionarioBean();
                                dicionarioBean.setIdAbreviacao(rs.getInt("idAbreviacao"));
                                dicionarioBean.setAbreviacao(rs.getString("abreviacao"));
                                dicionarioBean.setExpansao(rs.getString("expansao"));
                                dicionarioBean.setCategoria(rs.getString("categoria"));
                                dicionarioBean.setTexto(rs.getString("texto"));
                                listaAbrev.add(dicionarioBean);
                        }
                         rs.close();
                }
                catch(Exception e){}
                return listaAbrev;
        }
}
```

d) DicionarioServlerController.java

```
package pacote.controller;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.ArrayList;
import java.util.*;
import pacote.model.DicionarioBean;
import pacote.model.DicionarioBO;

public class DicionarioServlerController extends javax.servlet.http.HttpServlet
implements javax.servlet.Servlet
{
	public void doGet(HttpServletRequest request, HttpServletResponse response)
	 throws IOException, ServletException
	{
		processRequest(request, response);
	}

	public void doPost(HttpServletRequest request, HttpServletResponse response)
	 throws IOException, ServletException
	{
		processRequest(request, response);
	}

	public void processRequest(HttpServletRequest request, HttpServletResponse
response)throws IOException, ServletException{
		String pagina = null;
		pagina = listaDicionario(request, response);
		if(request.getRequestURI().endsWith("/dicionario_listar"))
			request.getRequestDispatcher(pagina).forward(request, response);
		else response.sendRedirect(request.getContextPath() + "/index.html");
	}

	public String listaDicionario(HttpServletRequest request, HttpServletResponse
response)throws IOException{
		DicionarioBO dicionarioBO = new DicionarioBO();
		List abreviaturas = new ArrayList();
		String url = "";
		abreviaturas = dicionarioBO.listarDicionario();
		url = "dicionario_lista.jsp";
		request.setAttribute("abreviaturas", abreviaturas);
		 return url;
	}
}
```

e) web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd" version="2.4">
    <display-name>Dicionario de EN abreviadas<display-name>
    <description>
		Essa é uma aplicação baseada no modelo MVC que disponibiliza informações
		 sobre as EN abreviadas.
    </description>
    <servlet>
        <servlet-name>NomeServlet</servlet-name>
        <servlet-class>pacote.controller.DicionarioServlerController</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>NomeServlet</servlet-name>
        <url-pattern>/dicionario_listar</url-pattern>
    </servlet-mapping>
</web-app>
```

f) dicionario_lista.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ page import="java.util.*" %>
<%@ page import="pacote.model.DicionarioBean" %>
<!DOCTYPE     html     PUBLIC     "-//W3C//DTD     HTML     4.01     Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>

<head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Lista Abreviaturas</title>
</head>

<link href="estilo.css" type=text/css rel=stylesheet>

       <table class=tabela_normal_borda cellspacing="4" width="95%" border="1"><tbody>
       <img valign="top" align="left"src="images/logo_provisorio.gif">
       <tr>
              <td align="left" class=tabela_titulo>Abreviacao</td>
              <td align="left" class=tabela_titulo>Expansao</td>
              <td align="left" class=tabela_titulo>Categoria</td>
              <td align="left" class=tabela_titulo>Texto</td>
       </tr>
       <br><br><br><br><br><br>
       <%
              List listaAbrev = (ArrayList)request.getAttribute("abreviaturas");
              Iterator iter = listaAbrev.iterator();
               if ((listaAbrev != null) && (listaAbrev.size() > 0)) {
                     while (iter.hasNext()) {
                            DicionarioBean dicionarioBean=(DicionarioBean)
iter.next();
                            out.println("<tr>");
                            out.println("<td>"+dicionarioBean.getAbreviacao()+"</td>");
                            out.println("<td>"+dicionarioBean.getExpansao()+"</td>");
                            out.println("<td>"+dicionarioBean.getCategoria()+"</td>");
                            out.println("<td>" + dicionarioBean.getTexto()+"</td>");
                            out.println("</tr>");
                     }
              }
              else
              out.println("Não há abreviaturas cadastrados.");
       %>
</body></table>
```

g) index.html

```
<html>
  <head>
  <title>Pagina Inicial do Dicionario de Abreviaturas</title>
  </head>

  <body bgcolor=white>
  <table border="0">
    <tr>
      <td>
        <img valign="top" align="left"src="images/logo_provisorio.gif">
      </td>
      <tr>
        <tr>
          <tr>
            <td>
              <h1>Dicionário de Abreviaturas do corpus histórico brasileiro</h1>
              <p>Este dicionário apresenta algumas Entidades Nomeadas
                 encontradas no corpus histórico brasileiro. As informações exibidas de
                 cada abreviação são: abreviação, sua respectiva expansão, categoria e
                 frase em que essa abreviação foi encontrada (texto).
            </td>
          </tr>
        </tr>
      </tr>
    </tr>
  </table>
```

```
  <p>Para visualizar as abreviações clique no link a seguir:
  <ul>
    <li><a href="dicionario_listar">Link para listar</a>.
  </ul>

  </body>
</html>
```

h) estilo.css

```
body {
    PADDING-RIGHT: 0px; PADDING-LEFT: 0px; BACKGROUND: #ffffff; PADDING-BOTTOM: 0px;
MARGIN: 0px; FONT: 12px arial,comic sans ms,technical; COLOR: #000000; PADDING-TOP: 0px;
TEXT-ALIGN: center
}
a:link {
    TEXT-DECORATION: none; COLOR: #777777
}
a:visited {
    TEXT-DECORATION: none; COLOR: #777777
}
a:hover {
    COLOR: #000000; TEXT-DECORATION: underline; COLOR: #777777
}
a:active {
    TEXT-DECORATION: none; COLOR: #777777
}
.tabela_normal_borda {
    BORDER-RIGHT: #152246 thin solid; BORDER-TOP: #152246 thin solid; BORDER-LEFT:
#152246 thin solid; BORDER-BOTTOM: #152246 thin solid; FONT-SIZE: 14px; COLOR: #000000
}
.tabela_titulo {
    BORDER-RIGHT: #152246 thin solid; BORDER-TOP: #152246 thin solid; BORDER-LEFT:
#152246 thin solid; BORDER-BOTTOM: #152246 thin solid; FONT-SIZE: 14px; COLOR: #000000;
FONT-WEIGHT: bold; BACKGROUND-COLOR: #ffffe8
}
```

Glossário

Concordanciadores - programas que permitem que o usuário procure por palavras específicas em um corpus, trazendo o contexto de busca.

Corpus - conjunto de dados lingüísticos coletados criteriosamente para serem objeto de pesquisa lingüística e computacional.

Entidades Mencionadas - entidade referenciada num determinado contexto, podendo assim assumir papéis semânticos diferentes em função desse mesmo contexto.

Entidades Nomeadas – são nomes próprios de pessoas, organizações, locais, acontecimentos, coisas (objetos nomeados), obras (artefatos e construções humanas), conceitos abstratos, além de datas e valores. Informações sobre o contexto em que a entidade está presente são ignoradas.

Etiquetadores – programas que fazem análises automáticas do corpus e inserem etiquetas (códigos) de ordem morfossintática, sintática, semântica, ou discursiva.

Support Vector Machine – algoritmo de aprendizado de máquina supervisionado.