Universidade de Évora

Escola de Ciências e Tecnologia

Mestrado em Engenharia Informática

Um Sistema de Recuperação de Informação para Língua Tétum

Borja Loedaci Cauthe Patrocinio Antonino

Orientador
Professor Paulo Miguel Torres Duarte Quaresma

Évora, Março 2013

Mestrado em Engenharia Informática

Um Sistema de Recuperação de Informação para Língua Tétum

Borja Loedaci Cauthe Patrocinio Antonino

Orientador
Professor Paulo Miguel Torres Duarte Quaresma

Sumário

As tecnologias de informação atuais e os serviços baseados na *web* necessitam de gerir, selecionar e filtrar quantidades crescentes de informação textual. A classificação dos textos permite aos utilizadores consultar mais facilmente o conjunto dos textos do seu interesse. Este paradigma é muito eficaz tanto na filtragem de informação como no desenvolvimento dos serviços *online* dirigidos para o utilizador.
Propõe-se nesta dissertação um sistema de recuperação de informação para Língua Tétum baseada na plataforma open-source "Apache Solr". Este sistema foi desenvolvido com o objetivo de facilitar mais os utilizadores timorenses de fazer pesquisa com sua própria língua, o tétum. Foi testado versus os principais sistemas de recuperações de informações atuais como *Google* e *Bing* para ver a sua adaptação com a língua tétum.
Os resultados dos testes mostram que este sistema desenvolvido ultrapassou os dois grandes sistemas de recuperações de informações atuais em nível da resposta da pesquisa, e no futuro pode tornar-se como sistema de recuperação de informação recomendado para os utilizadores timorenses de fazer pesquisa com sua própria língua.

Palavra-chave: Recuperação de Informação, Sistema de Recuperação de Informação, Apache Solr, Lingua Tétum

A Information Retrieval System for Tetum Language

Abstract

Current information technology and Web-based services need to manage, select and filter out increasing amounts of textual information. The text classification allows users to browse more easily the set of texts of interest. This paradigm is very efficient for filtering information in the development of online services directed to the user.
Proposed in this dissertation an information retrieval system for language Tetum based on open-source platform "Apache Solr". This system was developed to facilitate Timorese users to do research with their own language, Tetum language. Was tested versus the current major information retrieval system like Google and Bing to see the adaptation of current system with Tetum.
The test results show that this system developed exceeded two current major information retrieval systems at the level of survey response, and in the future can make this system as the recommended information retrieval system for Timorese users to do their own research language.

Keyword : Information Retrieval, Information Retrieval System, Apache Solr, Tétum.

Pelos meus pais e irmãos, pela minha pátria.

Agradecimentos

A Deus pela grandeza e graças na minha vida em concluir o meu estudo.
Gostaria de agradecer profundamente ao meu orientador, Prof. Paulo Quaresma, pelo seu apoio e incentivo.
Gostaria de agradecer aos meus pais Antonino Gonçalves e Ivone José Patrocinio, aos meus irmãos Ivan Patrocinio Antonino e Francisco José Patrocinio Antonino, pelas orações, amor e paciência.
Agradeço ao Universidade Nacional de Timor Lorosa'e (UNTL) e Instituto Português de Apoio ao Desenvolvimento (IPAD) pela bolsa de estudo.
Gostaria também de agradecer aos amigos do curso Mestrado em Engenharia Informática, em especial Emanuel, Helder, Serge, que sempre disposto a resolver os meus problemas que tiveram durante a frequentar o curso.
Por ultimo queria agradecer aos meus companheiros estudantes Timorenses em Évora, em especial Maria e Domingos pelo apoio e compreensão desde o inicio até finalização do meu estudo.

Acrónimos

CSV Comma-separated Values

EUA Estados Unidos da America

HTML Hyper Text Markup Language

HTTP Hyper Text Transfer Protocol

INL Instituto Nacional de Linguística

PDF Portable Document Format

REST Represantational State Transfer

RI Recuperação de Informação

RTF Rich Text Format

SOAP Simple Object Access Protocol

SGML Standard Generalized Markup Language

SRI Sistema de Recuperação de Informação

WWW World Wide Web

XML eXtensible Markup Language

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Sistemas para recuperação de informação baseados na *web* são sistemas complexos e exigem a soma de conhecimentos de profissionais com domínio de conhecimento em diversas áreas. A *Internet* se transformou em um vasto repositório de informações. Podemos encontrar sites sobre qualquer assunto, de futebol a religião. O difícil, porém é conseguir encontrar a informação certa no momento desejado. Desta forma, para auxiliar na pesquisa de conteúdo dentro da *internet* foram criadas as ferramentas de pesquisa.
O tema Recuperação de Informação sempre foi um tema muito explorado na academia e no mercado. A forma com que os eventos académicos são conduzidos, demonstra uma maturidade muito grande da área, inclusive, uma ligação muito forte com o mercado, como documentos são geralmente textos ou partes do texto de documentos e o principal objectivo de um sistema de recuperação de informação é recuperar as informações contidas nos documentos que possam ser úteis ou relevantes para os utilizadores.
As Tecnologias de Informação atuais e os serviços baseados na *web* necessitam de gerir, selecionar e filtrar quantidades crescentes de informação textual. A classificação de textos permite aos utilizadores consultar mais facilmente o conjunto de textos do seu interesse. Este paradigma é muito eficaz tanto na filtragem de informação como no desenvolvimento de serviços *online* dirigidos para o utilizador.
Com o desenvolvimento deste trabalho que pretende-se criar um sistema de recuperação da informação para a língua tétum, o autor quer resolver um problema que descobriu no trabalho anterior, referente a "Erros na Pesquisa com Língua Tétum nas Maquinas de Pesquisa". Naquele trabalho o autor chegou a esta conclusão: os resultados dos testes efetuados demonstram que a maioria, ou seja, mais de 80% das palavras usadas não foram reconhecidas pelas máquinas de pesquisa.
Com o resultado deste trabalho, acredita-se que pode diminuir os problemas se os utilizadores com língua Tétum forem recolhendo as informações espalhadas na *internet*.

O conteúdo deste trabalho e organizado por seguinte : no ponto 2 apresenta as ferramenta utilizadas para desenvolver o sistema; no ponto 3 apresenta o sistema desenvolvida e as técnicas utilizadas.; no ponto 4 apresenta testes e resultados versus outros SRI atual (*Google* e *Bing*); e por último é a conclusão trabalho futuro do autor.

1.1 Justificação do Tema

Durante o mestrado, optou por realizar uma análise sobre as margens de erros nas pesquisas com língua Tétum e chegou a esta pequena conclusão: a maioria das máquinas de pesquisa não conseguiu recuperar a informação com a língua tétum.
Adequado ao tema, descreve os conceitos e premissas que regem o desenvolvimento da área de Recuperação de Informação, os modelos existentes e os métodos para avaliação dos Sistemas de Recuperação da Informação.

1.2 Objetivo do trabalho

O presente estudo visa desenvolver um sistema de recuperação da informação para a Língua Tétum. No que concerne aos objetivos específicos pretende-se:

- Contribuir para desenvolvimento de informação e tecnologia de Timor Leste,
- Facilitar os Utilizadores Timorenses de fazer uma pesquisa a partir da sua própria língua.

1.3 Metodoligia do Trabalho

A metodologia utilizada para a realização do trabalho está dividido em três fases:

- Fase do estudo
  Este é a fase inicial da realização deste trabalho, nesta fase estudou-se o resultado que teve no seu trabalho anterior, também estudou-se a ferramenta e técnicas que vão ser utilizadas para desenvolver este sistema.
- Fase de levantamento dos dados
  Após estudar e analisar os resultados e técnicas que vão ser usadas neste sistema, foram levantados os dados de várias fontes de informações, os dados levantados são conjunto das palavras em tétum, sinónimos e palavras paradas.
- Fase de desenvolvimento
  É a fase mais importante, porque nesta fase foram implementadas todas as técnicas

estudadas e dados levantados, também nesta fase se fizeram testes de desempenho para o sistema desenvolvido.

Capítulo 2

Ferramentas Utilizadas

2.1 Apache Solr

Solr é uma das plataformas de pesquisa aberta altamente escalável do projeto de Apache Lucene[1] da *Apache Software Foundation*. É escrito em Linguagem Java e funciona como um "*servidor de pesquisa*" autónomo do texto completo dentro de um conjunto do SERVLET[2] , como o *Apache Tomcat* ou *Jetty*. As suas principais características incluem pesquisa do texto completo, os resultados destacam agrupamento dinâmico e gestão dos documentos de texto rico[3] (como o Word e PDF). Solr é escalável, permitindo pesquisa distribuída e replicação do índice, e atualmente está sendo usado em muitos sites da *internet*.

Figura 2.1: Uso do solr em comum (Smiley D., Pugh E. (2009). Solr 1.4 Enterprise Search Server).

[1]O Apache Lucene, é um software de pesquisa e uma API de indexação de documentos, escrito na linguagem de programação Java.

[2]SERVLET (servidorzinho em tradução livre) é um componente como um servidor, que gera dados HTML e XML para a camada de apresentação de uma aplicação Web. *(wikipedia)*

[3]Em Inglês : *Rich Text Document*

A principal característica do Solr é que em seu núcleo para indexação e pesquisa do texto utiliza a biblioteca pesquisa do Lucene, escrito em línguagem *JAVA* e tem uma *API* estilo REST[4] como HTTP / XML e JSON que tornam mais fácil de usar, comparando com qualquer outro, porque em vez de usar os *drivers* ou *API's* programáticos para se comunicar, o Solr pode fazer pesquisa através de HTTP e devolve os resultados em XML ou JSON.
Alguns das principais características do Solr são:

- Servidor com interface como o REST (interação via HTTP, XML, JSON, CSV, etc.)
- Esquema de dados configuráveis.
- Utiliza várias caches para pesquisas mais rápidas.
- Interface *web* administrado.
- Escalável aos vários servidores para pesquisa distribuída.
- Módulos de importação de dados desde o base de dados, *e-mail* e documentos de texto ricos (PDF, Word, RTF).
- Análise de texto (Tokenização padronizada[5] , etc.).

Figura 2.2: Mapa conceitos do solr (Iccha Sethi, Serdar Aslan, Dr. Edward Fox)

[4]Representational State Transfer (REST) é uma técnica de engenharia software para sistema hipermédia distribuídos como a World Wide Web
[5]Em Ingles : *Standard Tokenizer*

Solr não expõe uma interface REST "perfeito"que significa não ser possível de usar todos os princípios de HTTP, mas os dados têm uma representação simples, que circula entre o cliente e o servidor, sem qualquer encapsulamento raro com SOAP[6] ou outros protocolos de acesso. Além disso, o XML é legível, por isso JSON pode ser utilizada juntamente com *JavaScript* e também para realizar testes.
As vantagens do REST que utilizado pelo Solr são :

- Linguagem XML e JSON são independentes, e pode interpretar qualquer linguagem com eles. Como a métrica é normalmente escrita em *JavaScript* e tem suporte nativo para JSON e XML, assim podemos interpretá-los dentro de todas as limitações de um navegador, também ser aplicado em outras partes.
- Os tipos de dados são independentes, por exemplo, HTTP transmite apenas texto.
- Podemos inventar uma base de dados, publicar o protocolo de comunicação binário próprio e assim ter muito mais bibliotecas.

2.1.1 Indexação e exclusão de documentos em Solr

Antes de indexar documentos em Solr é necessário definir os campos que compõem os documentos indexados e especificar o tipo de dados de cada campo, a ligação e muito mais. É também necessário definir o campo que vai identificar o documento exclusivamente dentro do servidor Solr. Esta configuração é definido no *schema.xml*. O Solr fornece uma interface estilo REST para enviar e receber as mensagens entre o cliente e o Solr.
Para indexar documentos utiliza a instrução *ADD* do Solr no formato XML via HTTP POST.

```
<add>
        <doc>
                <field name="id"> TetumDoc</field>
                <field name="name">Ezemplu ho Lingua Tetum</field>
                <field name="features"> Universidade Nacional de Timor
                Lorosa'e</field>
                <field name="features"> Faculdade de Enjenaria</field>
                <field name="features"> Departamento de Informatica </field>
                <field name="features"> Kampus Hera </field>
        </doc>
</add>
```

Por outro lado, para remover um documento utiliza-se o comando *DELETE* com o identificador do documento a remover, ou com um critério de seleção para eliminar vários registos de uma vez:

```
<delete>
        <id>TetumDoc<id>
<delete>
```

[6]SOAP (Simple Object Access Protocol) é um protocolo para troca de informações estruturadas em uma plataforma descentralizada e distribuída.

Por último, para qualquer mudança a ser refletida é necessário enviar a instrução *COMMIT* do Solr:

```
<commit/>
```

Solr é escrito em Java, mas podemos utilizar qualquer linguagem preferida para pesquisar o índice e para adicionar documentos.

2.1.2 Linguagem de Consulta

A linguagem de consulta[7] usada por Solr suporta operadores booleanos e qualificadores campos úteis para aplicar uma pesquisa a um campo específico, usando carateres-curinga[8], operadores de ranking[9], também suporta para o uso de funções matemáticas e opções de classificações. Uma consulta típica em Solr é através de um HTTP *GET* usando o parâmetro para especificar a consulta:

```
http://localhost:8983/solr/select/?q=hakerek+Tetum
```

Este pedido em HTTP é o tipo de pedido que o Solr deve gerar para fazer a pesquisa. Quanto ao formato dos resultados, o Solr tem vários formatos para mostrar o resultado da pesquisa como XML, JSON, Python, Ruby, PHP, CSV. No exemplo a baixo, a resposta da consulta feita em XML.

```
<response>
        <lst name="responseHeader">
                <int name="status">0</int>
                <int name="QTime">49</int>
                        <lst name="params">
                                <str name="indent">on</str>
                                <str name="start">0</str>
                                <str name="q">hakerek Tetum</str>
                                <str name="version">2.2</str>
                                <str name="rows">10</str>
                        </lst>
        </lst>
        <result name="response" numFound="38" start="0">
                <doc>
                        <arr name= "content_type">
                                <str>text/plain</str>
                        </arr>
                        <arr name="id">
                                <str>
                                /Users/patant/Documents/apache-solr -3.6.0/
                                SRITetum/documentos/tetum.xml
                                </str>
```

[7] Em Inglês : *Query Language*

[8] Em computação os carateres-curinga são utilizados em casamento de padrões para substituir algum outro caráter desconhecido numa sequência de carateres.

[9] Em Inglês : *Rangking Operator*

```
                    </arr>
                </doc>
        </result>
</response>
```

2.1.3 eXtensible Markup Language (XML) no Solr

É uma linguagem que foi desenhada por W3C[10] para gerar linguagem de marcação[11] . XML é classificada como linguagem extensível porque permite definir os elementos de marcação e também porque consegue combinar a flexibilidade da SGML com a simplicidade do HTML.
A principal característica do XML é criar uma infraestrutura única para diversas linguagens. Baseada em vários princípios importantes, o XML é um formato que não depende das plataformas de *hardware* ou de *software* e que cria documentos a partir de dados organizados de forma hierárquica.

schema.xml

É um ficheiro codificado baseado em padrão do XML que contém a definição da estrutura de um documento XML, as definições de tipo, tamanho, ocorrência e regras de preenchimento dos elementos que compõem o documento XML.
É também o primeiro ficheiro que devemos configurar na nova instalação do Solr e que se declara:

- Os tipos[12] e campos[13] que vai ser utilizado na sistema;
- O campo que vai ser utilizado como a chave primária;
- Os campos que são obrigatórios;
- Como indexar e pesquisar.

solrconfig.xml

Geralmente é o segundo arquivo (depois *schema.xml*) que devemos configurar depois de uma nova instalação do Solr. Os elementos deste arquivo mais utilizados são:

[10] O **World Wide Web Consortium (W3C)** é a principal organização de padronização da World Wide Web.
[11] *Linguagem de marcação* é um agregado de códigos que podem ser aplicados a dados ou textos para serem lidos por computadores ou pessoas. Exemplo: HTML
[12] Em Inglês: *Type*
[13] Em Inglês: *Field*

- Local do diretório de dados; Usado para especificar um diretório alternativo para armazenar todos os dados de índice.
- Parâmetros de cache;
- Manipuladores de solicitação[14]; São responsáveis por aceitar pedidos HTTP, a realização de pesquisas, em seguida, retornar os resultados.
- Componentes de pesquisa. São responsáveis por executar as pesquisas reais.

2.1.4 Ruby no Solr

Ruby é uma linguagem orientada a objetos, com tipagem forte e dinâmica[15] . Foi apresentada ao público pela primeira vez em 1995 por Yukihiro Matsumoto e é uma das únicas linguagens nascidas fora do eixo EUA - Europa que atingiram enorme sucesso comercial. Uma das suas principais características é a expressividade que possui. Teve como objetivo desde o início uma linguagem muito simples de ler e ser entendida para facilitar o desenvolvimento e manutenção dos sistemas escritos com ela. Ruby é uma linguagem interpretada e, como tal, necessita da instalação de um interpretador em sua máquina antes de executar algum programa. O solr-ruby no solr encapsula as operações fundamentais do solr.

2.2 TÉTUM

História do Tétum

É uma das línguas oficiais de Timor-Leste. É uma língua austronésia como a maioria das línguas autótones da parte leste da ilha com muitas palavras derivadas do português e outras comuns ao malaio. O primeiro tétum, o tétum-térique[16], já se estabelecera como língua franca antes da chegada dos portugueses, aparentemente em consequência da conquista da parte oriental da ilha pelo reino de Ué-Hali, da província dos Belos e da necessidade de um instrumento de comunicação comum para as trocas comerciais.
Com a chegada dos portugueses e estabelecido o contacto entre a comunidade timorense e a presença lusa na missionação, nos quadros administrativos e comerciais ocasionou a adopção de termos e conceitos novos do português para completar as lacunas do léxico tétum.
Muito embora, em finais do século XIX, os jesuítas de Soibada, em 1938, tenham já traduzido para tétum parte da Bíblia e, em 1916, o governador Filomeno da Câmara tenha

[14]Em Inglês: *Request handlers*

[15]A **tipagem forte** ocorre quando a linguagem não permite que uma variável tenha seu valor automaticamente alterado para outro tipo para possibilitar uma operação, A **tipagem dinâmica** ocorre quando a linguagem não obriga a prévia declaração de tipo de uma variável. O tipo é assumido na atribuição de valor a variável, que pode ser por presunção ou forçado com casting. Além disso, é possível modificar o tipo da variável atribuindo-lhe outro valor.

[16]Em Tétum: *tetun-terik*

adoptado o livro Cartilha Tétum-Português como manual oficial para aprendizagem nas escolas oficiais de Timor, foi apenas em 1981 que o tétum se tornou língua oficial da Igreja nos Atos Litúrgicos.
O português foi língua do ensino e da administração do Timor Português, mesmo assim o tétum veicular era a língua de interação social e comercial de todos os grupos étnico-linguísticos. Quando a Indonésia invadiu e ocupou Timor-Leste em 1975, declarando-o 27ª Província da República Indonésia, o uso do português foi proibido. Mas a Igreja Católica, em vez de adoptar a língua indonésia como língua litúrgica, optou por tétum, tornando-o num pilar da identidade cultural e nacional.
Atualmente, o tétum é a língua mais falada[17] em Timor-Leste. Apesar do tétum-praça possuir variações regionais e sociais, hoje o seu uso é alargado porque é compreendido por quase toda a população timorense. É este tétum-praça que foi adaptado como "língua oficial[18] "com a designação de Tétum Oficial.

Ortografia

A ortografia moderna do tétum é fonémica e corresponde em grande parte á pronúncia real das palavras (Geoffrey Hull, *Manual de Língua Tétum para Timor Leste, p.4.*). Tal como o Português, o tétum usa o alfabeto latino, distribuído em 5 vogais: a, e, i, o, u e 20 consoantes: **b, d, f, g, h, j, k, l, ll, m, n, ñ, p, r, s, t, v, x, y, z** e ainda o apóstrofe [**'**] como um tom.
No tétum são excluídas as letras c e q. Além disso g, h, s e x nem sempre têm os mesmos valores do português. O INL (Instituto Nacional de Linguística) adotou os dígrafos do galego ll e ñ para os do português lh e nh, (por exemplo: Toalla ↔ Toalha, Liña ↔ Linha).
A acentuação em tétum cai geralmente na penúltima sílaba da palavra e nos casos irregulares cai na última ou na antepenúltima sílaba. A sílaba tónica da palavra é assinalada graficamente com acento agudo na última e na antepenúltima sílaba. Palavras com acento tónico na penúltima sílaba dispensam ser grafadas com acento. O acento circunflexo não é usado. (Geoffrey Hull, *Manual de Língua Tétum para Timor Leste, p.5.*).
Exemplos da acentuação:

Tétum	**Português**
Matak[má-tàk]	Cru, verde
Labarik [la-bá-rik]	Criança
Kafé	Café
Xumingál	Pastilha elástica
Portugés	Português
Kópia	Cópia

Tabela 2.1: Exemplo palavras com acentuação em Lingua Tétum

[17] Resultado do Censo de Timor Leste de 2010 (Senso populasaun no uma-kain 2010, *distribuisaun populasaun tuir área administrativos, volume 2, p.203*)
[18] Vê constituição da RDTL artigo 13º (língua oficiais e língua nacionais).

2.3 Processamento Linguagem Natural

O Processamento da Linguagem Natural (PLN) pode ser definido como um conjunto de técnicas computacionais para a análise dos textos em um ou mais níveis linguísticos, com o propósito de simular o processamento da linguagem humana.
Estas técnicas costumam ser utilizadas em SRI para a melhoria de algumas tarefas no processo de recuperação, como a extração automática de descritores.
As técnicas de PLN que vão ser utilizadas neste sistema são:

- morfológico: da construção das palavras a partir das unidades de significado primitivas e de como classificá-las em categorias morfológicas;
- lexical: onde são analisados a estrutura e o significado da palavra, tendo como exemplo técnicas de alguns tipos de expansão da consulta e da correção ortográfica;
- sintático: do relacionamento das palavras entre si, cada uma assumindo o seu papel estrutural nas frases e de como as frases podem ser partes de outras, constituindo sentenças;
- semântico: onde é interpretado não o sentido isolado das palavras, mas sim o sentido das expressões e frases, além do tratamento e resolução de ambiguidades;

2.3.1 Normalização De Variações Linguísticas

A normalização das variações linguísticas pode ser segmentada em três tipos:

Normalização morfologica

Na normalização morfológica ocorre à tentativa de reduzir os itens lexicais, como as palavras, a uma forma que visa simbolizar classes de conceitos. O processo de colação é geralmente utilizado nesta normalização, pois este visa reduzir variantes de uma palavra a uma forma, possibilitando assim que os termos com variações diferentes possam ser combinados.
Os procedimentos mais conhecidos para colação são:

- Derivação[19] : visa fundir variações das palavras em uma representação comum, o radical, que não é necessariamente igual à raiz linguística.
- Lematização[20] : também conhecido como redução à forma canónica, visa reduzir as palavras à sua forma canónica, gerando comummente verbos no infinitivo e palavra com adjetivo no singular masculino.

[19] Em Inglês: *Stemming*
[20] Em Inglês: *Lemmatization*

Normalização Sintatica

Na normalização sintática, as frases semanticamente similares são normalizadas a uma forma única e representativa das mesmas. Um exemplo deste tipo de frases é "*uma ne'e di'ak i kapás*" e "*uma ne'e kapás i di'ak*".

Normalização Léxico-sêmantica

Na normalização léxico-semântica são utilizados relacionamentos semânticos, por exemplo sinonímia, entre os itens lexicais a fim de estabelecer agrupamentos de similaridades semânticas. Um processo inverso a este em geral utilizado no sistema de pesquisa, no qual um item lexical é expandido com o auxílio de seus relacionamentos semânticos. Este procedimento é conhecido como expansão das consultas.
A utilização da normalização de variações linguísticas auxilia no controle do vocabulário, sendo que diferentes tipos de normalizações podem ser combinadas, conforme a necessidade.

2.3.2 Expansão das Consultas

A expansão da consulta é uma técnica na qual visa aumentar a quantidade de termos que devem ser pesquisados numa consulta, sendo que estes possuem certo grau de equivalência entre eles.
Esta técnica torna-se necessária para que haja uma maior efetividade nas pesquisas, tendo em vista dois problemas que afetam a qualidade das pesquisas em SRI, que são a sinonímia e a polissemia.

- A *sinonímia* esta relacionada ao fato de haverem diversas maneiras de se referir o mesmo objecto.
  Exemplo :

  Ha'u nia aman fuuk la iha
  Ha'u nia aman ulun molik[21]

- A *polissemia* diz respeito ao fato de algumas palavras possuírem diferentes significados, dependendo do contexto na qual elas estão inseridas.
  Exemplo:

  Kuda = Cavalo
  Kuda = Cultivar

[21] Em Português: *O meu pai é careca.*

Atualmente vários métodos de expansão de consulta foram propostos, sendo possível classificá-los em dois principais grupos: métodos iterativos e métodos automáticos.

Expansão iterativa da consulta

A expansão iterativa da consulta, também conhecida como expansão semiautomática ou *relevance feedback*, caracteriza-se por uma técnica na qual é necessária a iteração entre o utilizador e o sistema para que possa ser efectivada a expansão da consulta.

Expansão automática da consulta

A expansão automática da consulta é uma técnica que, ao contrário da expansão iterativa da consulta, não necessita da intervenção do utilizador para que a expansão possa ser efectuada, o que torna uma técnica mais interessante, uma vez que o processo é transparente para o utilizador. Esta técnica é também conhecida como *pseudo* realimentação de relevantes.

2.4 Recuperação de Informação

A Recuperação da informação é um processo fundamental da comunicação, onde utilizadores com necessidades de informação descrevem essas mesmas necessidades e a coleção onde será efetuada a pesquisa contém documentos descritos de uma forma que os utilizadores entendam (Blair, 1990).
É uma ciência da pesquisa sobre pesquisa por informações em documentos, pesquisa pelos documentos propriamente ditos, pesquisa por meta de dados que descrevem documentos e pesquisa em bases de dados, sejam eles relacionais e isolados ou bases de dados interligados em rede de hipermédia, tais como a *World Wide Web*. A média pode estar disponível sob forma de textos, de sons, de imagens ou de dados.
Segundo Souza (2006, p.166), os modelos de recuperação dividem-se em modelos clássicos e modelos estruturados. Nos modelos clássicos, cada documento é descrito por um conjunto de palavras-chave representativas – também chamadas de termos de indexação – que pesquisa representar o assunto do documento e sumarizar o seu conteúdo de forma significativa. Nos modelos estruturados, podem-se especificar, além das palavras-chave, algumas informações acerca da estrutura do texto (como seções a serem pesquisadas, fontes de letras, proximidade das palavras, entre outras informações).
Para os utilizadores, o processo da Recuperação de Informação parte de uma necessidade da informação. Os utilizadores fornecem a um Sistema da Recuperação de Informação uma consulta formulada a partir da sua necessidade de informação. O sistema então compara a consulta com documentos armazenados.
A tarefa do Sistema da Recuperação de Informação é retornar ao utilizador os documentos que mais satisfazem a necessidade do utilizador. Para um Sistema da Recuperação

de Informações, um processo da Recuperação de Informação inicia quando o utilizador informa uma consulta ao sistema, consultas são representações formais das necessidades de informação de um utilizador.

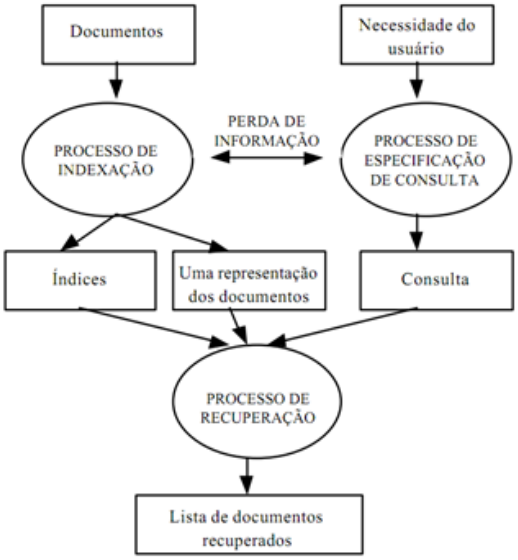


Figura 2.3: Componente de um SRI(elaboração própia)

Em um Sistema da Recuperação de Informação uma consulta não é associada a um único documento numa coleção. Ao contrário, diversos documentos são retornados através de uma consulta, selecionando-se os documentos que se apresentam como mais relevantes, comparando a consulta com as representações dos documentos previamente armazenados. Um Sistema da Recuperação de Informação tem três componentes básicos:

- aquisição e representação da necessidade de informação,
- identificação e representação do conteúdo do documento,
- especificação da função da comparação que seleciona os documentos relevantes, baseada nas representações.

A consulta em Recuperação de Informação é a forma que o utilizador possui para representar a sua necessidade de informação num Sistema da Recuperação de Informações.
A necessidade de informação normalmente não é especificada em linguagem natural mas sim através de palavras-chave. Após a elaboração da consulta, o Sistema da Recuperação de Informações tenta localizar objetos que possam ser relevantes para o utilizador. Um Sistema da Recuperação de Informações não tem a responsabilidade de responder precisamente a uma consulta mas sim de identificar objetos que permitem que o utilizador satisfaça a sua necessidade de informação.

2.5 Máquina de Pesquisa

Uma máquina de pesquisa é um sistema de software projetado para encontrar informações armazenadas num sistema computacional a partir de palavras-chave indicadas pelo utilizador, reduzindo o tempo necessário para encontrar informações. Conhecidos também como programas de pesquisa, mecanismo de procura, ou ferramenta de pesquisa[22].
Uma máquina de pesquisa é um programa feito para auxiliar a procura de informações armazenadas na *World Wide Web (WWW)*, dentro de uma rede corporativa ou de um computador pessoal. Ele permite que uma pessoa solicite conteúdo de acordo com um critério específico (tipicamente contendo uma dada palavra ou frase) e responde com uma lista de referências que combinam com tal critério, ou seja, uma espécie de catálogo mágico.
As maquinas de pesquisas têm três funções principais:

- Um robô que localiza os documentos;
- Um indexador que extrai a informação dos documentos;
- Uma interface com o utilizador.

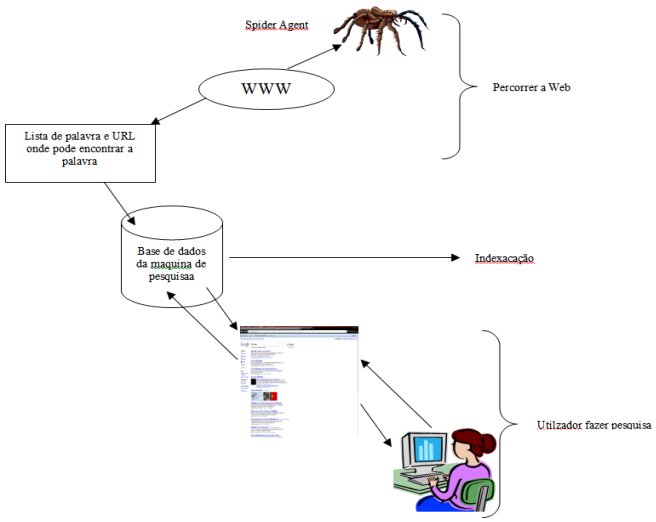


Figura 2.4: Processo de Pesquisa (elaboração própia)

Mas, diferentemente dos livros de referências comuns nos quais está acessível a informação que alguém organizou e registou, o catálogo da máquina de pesquisa está em branco, como um livro vazio. Ao realizar-se uma consulta, a lista de ocorrências de assuntos é criada em poucos segundos por meio de um conjunto de *software's* de computadores, conhecidos como *spiders*, que vasculham toda a Web em pesquisa de ocorrências de um determinado assunto em uma página. Ao encontrar uma página com muitos links, os

[22]Fonte: *Wikipedia, a enciclopédia livre http://pt.wikipedia.org/wiki/Motor_de_pesquisa*

spiders embrenham-se por eles, conseguindo, inclusive, vasculhar os diretórios internos ou aqueles que tenham permissão de leitura para utilizadores dos *sites* nos quais estão a trabalhar.
Os primeiros motores de pesquisa como *Yahoo* baseiam-se na indexação de páginas através da sua categorização. Posteriormente, surgiram as meta-pesquisas. A mais recente geração de motores de pesquisa como *Google* utiliza outras diversas tecnologias, como, a procura por palavras-chave diretamente nas páginas, o uso de referências externas espalhadas pela *web*, permitindo até a tradução direta de páginas (embora de forma básica ou errada) para a língua do utilizador.
O resultado de uma pesquisa é classificado e apresentado por um método conhecido como "relevância". Cada mecanismo de pesquisa utiliza método próprio de classificação. Os mecanismos de pesquisa têm as seguintes abordagens:

- Indexação de documentos da *web* pesquisando o máximo de acesso
- Recuperação por relevância utilizando técnicas de indexação automática

Além dos três componentes descritos acima, a maioria dos Sistemas da Recuperação de Informação incluem um quarto elemento que é chamado de realimentação de relevância. Pode-se pedir ao utilizador para selecionar documentos, ou frações de documentos, considerados relevantes a partir de um conjunto recuperado. Este conjunto de documentos relevantes pode ser usado para modificar as representações da necessidade de informação ou a função de comparação para melhorar as respostas a consultas posteriores. Máquina de pesquisas que vai ser utilizados para avaliar o desempenho do sistema desenvolvido:

2.5.1 Google

Google é um site de pesquisa que é considerado o maior *site* da internet e o mais visitado desde a sua criação em 1996 pelos americanos Lary Page e Serge Brin[23]. O *Google* indexa triliões de páginas *web* e executados através mais de um milhão de servidores em várias datas, espalhadas pelo mundo.

Figura 2.5: Janela inicial do google.

[23]Fonte: *Wikipedia, a enciclopédia livre (http://pt.wikipedia.org/wiki/Google).*

Basicamente, é uma técnica comum utilizada para pesquisar quaisquer informações espalhadas na *www* e os utilizadores usam muito em pesquisas de palavras e o *Google* transforma as palavras dadas em palavra-chave. Por fim, os operadores em pesquisa recuperam as informações por ordem de relevância. O *google* utiliza a tecnologia *PageRank*[24] para medir a importância e relevância de escala 1-10 e ordena as informações antes de recuperar as informações dadas pelos utilizadores.

2.5.2 Bing

Bing, anteriormente chamado *Live Search*, é uma máquina de pesquisa que foi desenvolvido por Microsoft e lançado ao público em 28 de maio de 2009, designado a competir as gigantes máquinas de pesquisa como *Google* e *Yahoo*. Uma semana depois do lançamento o *Bing* conseguiu alcançar o segundo lugar na área de pesquisa *online* atrás do *Google* e ultrapassou o *Yahoo* que ficou no terceiro lugar[25].

Figura 2.6: Janela inicial do Bing

Bing foi lançado com o seguinte objetivo:

- Mostrar o melhor resultado da pesquisa e ao clicar uma vez no botão pesquisa mostra-se a informação.
- Conseguiu organizar a consulta para maximizar o resultado da pesquisa.
- Simplificar e reduzir a pesquisa por meio de ajuda e tudo se resume a palavras-chave para tomar a decisão.

[24] PageRank foi a base do surgimento do Google.
[25] Fonte: *Wikipedia, a enciclopédia livre (http://pt.wikipedia.org/wiki/Bing).*

Capítulo 3

Aplicação Desenvolvida

Geralmente, o texto não é armazenado por inteiro num sistema de Recuperação de Informação. Para cada texto são criadas estruturas de dados, com o objetivo de acelerar o seu processo de recuperação. Por exemplo, os textos que devem ser indexados são submetidos a um processo de filtragem de termos relevantes, denominado extração de atributos. Os atributos extraídos a partir deste processo serão utilizados para caracterizar os documentos armazenados.
Os Sistemas de Recuperação de Informação devem possuir uma função que compara a consulta fornecida pelo utilizador com os textos armazenados no repositório. Esta função deve retornar ao grau de relevância dos documentos para a consulta. Os documentos identificados com maior grau de relevância são mostrados primeiro para o utilizador.
Neste capítulo vai falar-se das técnicas utilizadas para construir os sistemas e o funcionamento dos sistemas. Como já tinha referido no título, este sistema é para língua tétum baseado na ferramenta Apache Solr[1] , por isso, antes de construir este sistema o autor recolheu todos os dados precisos sobre a língua tétum, os dados que foram pesquisados são :

- Lista das Palavras (A-Z):
  Foram criadas um conjunto de palavras em tétum desde a letra A até Z, de acordo com Instituto Nacional de Linguística (INL).

- Os sinónimos:
  Para melhorar a relevância, é útil para um aplicativo de pesquisa entender como é que uma palavra pode ser usada por outra. E o ficheiro synonyms.txt do Solr é

[1] A ferramenta Apache Solr versão 3.6.0 está disponível no site do Apache Foundation (http://lucene.apache.org/solr/downloads.html).

o conjunto de palavras que têm o mesmo significado ou quase o mesmo com outra palavra numa língua. Com os problemas gramaticais que o tétum tem o autor decidiu elaborar este ficheiro como um conjunto de verbos e correções de algumas palavras que podem ser mal escritas no processo de consulta. Exemplo:

luku → luku ona, sei luku, luku
tasak → tasak ona, sei tasak, tasak
liki → liki ona, sei liki, liki
koñesimentu → konhesimentu, kuñisementu, konhisemento
sobriñu→ subrinhu, subrinho, subrinu, subrino

- Palavras paradas:
  Como a maioria das palavras mais frequentemente usadas na língua tétum são palavras sem valor de índice. Entre estas, por exemplo, “hela”, “hotu”, “di’ak”, “aat”, entre muitas outras, e estas palavras compõem uma grande fração da maioria dos textos de cada documento e também não contêm significado importante para ser usado em consultas de pesquisa. Por isso, foram criadas uma lista de palavras paradas. Normalmente, estas palavras são filtradas a partir de consultas de pesquisa porque retornam à vasta quantidade de informação desnecessária.

Apache Solr também é uma ferramenta que permite criar vários índices em campos diferentes, os campos estão definidos no ficheiro *schema.xml*. Na tabela abaixo vai descrever os campos utilizados neste sistema.

Tabela 3.1: Definição de todos os tipo de campos usados na sistema

Classe	**Descrição**
StrField	Campo de String
BinaryField	Campo de dados Binários
BoolField	Contém verdadeiro ou falso. Os valores de ”1”, ”t”, ou ”T”do primeiro caráter são interpretados como verdadeiro. Quaisquer outros valores na primeiro personagem são interpretado como falso.
TextField	Campo Texto, geralmente múltiplas palavras ou tokens
TrieField	Se este classe é usado, deve-se especificar o tipo de atributos, como : integer, long, float, double, date. Usar esta classe significa que vai usar todos os atributos desta classe.
TrieIntField	Campo integer, acessível no processamento de Lucene TrieRange

Continued on next page

Tabela 3.1 – *Continued from previous page*

Classe	**Descrição**
TrieLongField	Campo Longo, que acessível no processamento de Lucene TrieRange
TrieDoubleField	Campo Double, acessível no processamento de Lucene TrieRange
TrieFloatField	Campo integer, acessível no processamento de Lucene TrieRange
TrieDateField	Campo de data, acessível para o processamento de Lucene TrieRange
IntField	Campo Inteiro (32-bit)
LongField	Campo inteiro longo (64-bit)
FloatField	Pontos flutuante (32-bit IEEE pontos flutuantes)
DoubleField	Campo do Double (64-bit IEEE pontos flutuantes)
DateField	Representa um ponto no tempo com precisão de milissegundos.
SorttableIntField	Ordenar numericamente o campo inteiro
SorttableLongField	Ordenar numericamente o campo inteiro longo
SorttableFloatField	Ordenar numericamente os pontos flutuantes
SorttableDoubleField	Os campos Sortable fornecem uma ordenação numérica correta. Se usar os tipos simples (DoubleField, IntField, e assim por diante) a ordenação será lexicográfica em vez de numérica.
RandomSortField	Não contém um valor. As consultas que ordenam este tipo de campo irão retornar resultados em ordem aleatória. Use um campo dinâmico para usar este recurso.

3.1 Instalação do Apache Solr

Solr pode ser executado em qualquer recipiente SERVLET Java de nossa escolha. Para iniciar o solr basta executar somente o ficheiro *start.jar*.

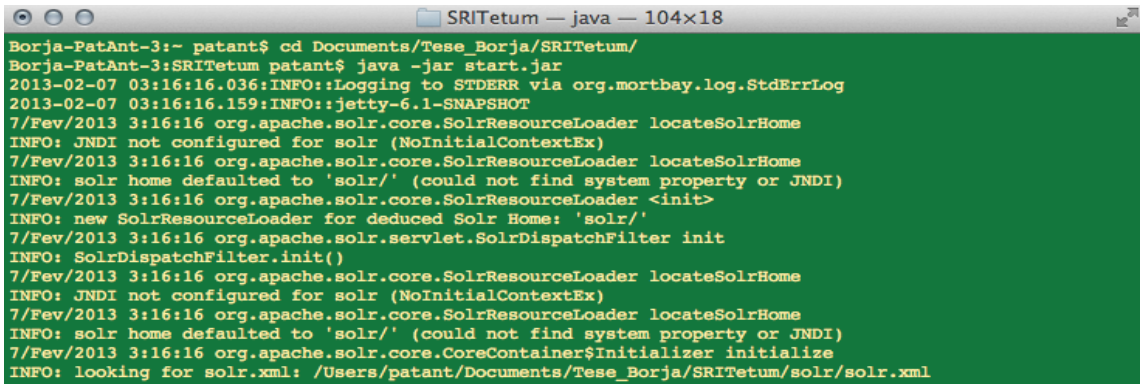


Figura 3.1: Execução do ficheiro start.jar do Solr.

Com a execução do ficheiro *start.jar* irá iniciar o servidor de aplicação na porta 8983 e usar o nosso terminal para exibir as informações de registo do solr. Podemos ver que o solr está sendo executado por carregar *http://localhost:8983/solr/admin/* no nosso navegador. Este é o principal ponto de partida para a Administração Solr.

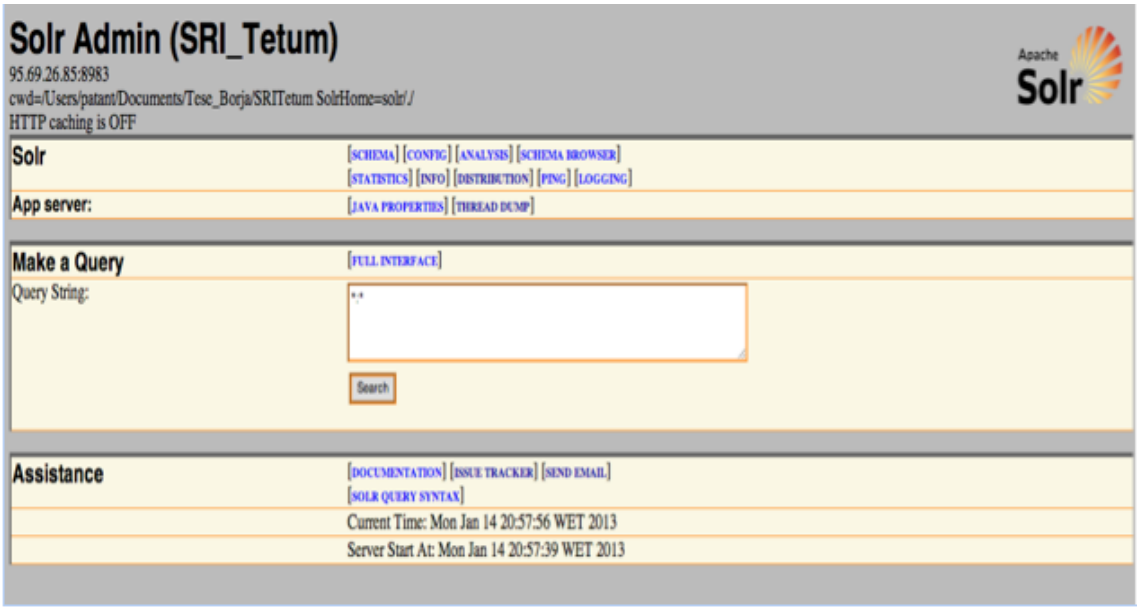


Figura 3.2: Página inicial do Sistema desenvolvido.

3.2 Implementação do Analisador do texto, simbolizador e Filtragem do Solr

Para que sistema desenvolvido consiga adaptar bem com a língua tétum foram implementadas várias técnicas do Apache Solr.

3.2.1 Analisador do texto

Um analisador do texto examina o campo de texto e gera um fluxo de símbolos , analisadores dos textos especificados como um filho do elemento *fieldtype* no ficheiro da configuração schema.xml. Os analisadores dos textos são utilizados durante a tomada de ação quando um documento é indexado no momento da consulta. Analisador do texto pode ser uma única classe ou podem ser compostos por uma série de classes de simbolizador e filtragem. O exemplo que se segue vai mostrar como um analisador do texto funciona.

```
<fieldType name="text_tt" class="solr.TextField">
 <analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.StandardFilterFactory"/>
        <filter class="solr.LowerCaseFilterFactory"/>
        <filter class="solr.StopFilterFactory" ignoreCase="true"
        words="lang/stopwords_tt.txt"
        enablePositionIncrements="true "/>
 </analyzer>
</fieldType>
```

Neste caso, a classe analisadora do texto foi especificada no elemento <*analyzer*>. Depois disso, é uma sequência de classes mais especializadas estão ligadas entre si e colectivamente, agindo como um analisador do texto nos textos ou nas palavras no momento da consulta. O campo do texto é passado para o primeiro item da lista que é *solr.StandardTokenizerFactory*, depois são transformados em letras minúsculas no item *solr.LowerCaseFilterFactory* e, por último, as palavras encontradas no campo do texto que estão listados no ficheiro *stopwords_tt* vão ser eliminados. A partir disso saem os símbolos finais que vão ser usados para indexar ou fazer consulta a qualquer campo de texto que usa o *fieldType=text_tt*.

3.2.2 Simbolizador

O simbolizador tem a função de partilhar os campos de textos em unidade lexical ou símbolos, as filtragens examinam a classificação dos símbolos e mantêm-no, transformam-no até poder eliminar ou criar um novo.
O simbolizador e a filtragem podem combinar-se para formar uma corrente, onde a saída ou *output* de uma das duas pode ser como entrada ou *input* da próxima saída.
A sequência ou conjunto do Simbolizador e filtragem é chamado analisador do texto e o

output do analisador do texto utilizado para construir índice da pesquisa.

```
<fieldType name="text" class="solr.TextField">
 <analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
 </analyzer>
</fieldType>
```

Standard Tokenizer

Este tipo de simbolizador divide o campo do texto em símbolos, trata os espaços brancos e as pontuações como delimitadores. Os carateres delimitadores vão ser rejeitados, mas com seguintes exceções:

- Pontos ou vírgulas não são seguidos por espaços brancos e mantidos como parte de símbolos.
- As palavras divididas com hífens, com a exceção da existência de números na palavra, os símbolos, nestes casos, não são separados, os números e o hífen preservados.

```
<analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
</analyzer>
```

Exemplo:
Entrada:

Aban ha'u ba halimar iha imi nia uma

Saída:

org.apache.solr.analysis.StandardTokenizerFactory {luceneMatchVersion=LUCENE_36}

position	1	2	3	4	5	6	7	8
term text	Aban	hau	ba	halimar	iha	imi	nia	uma
startOffset	0	5	9	12	20	24	28	32
endOffset	4	8	11	19	23	27	31	35
type	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>

Figura 3.3: Resultado do Standard Tokenizer.

Keyword Tokenizer

Este é o simbolizador que trata um campo do texto inteiro como um único símbolo.

```
<analyzer>
        <tokenizer class="solr.KeywordTokenizerFactory"/>
</analyzer>
```

Exemplo:
Entrada:

Aban ha'u ba halimar iha imi nia uma

saída :

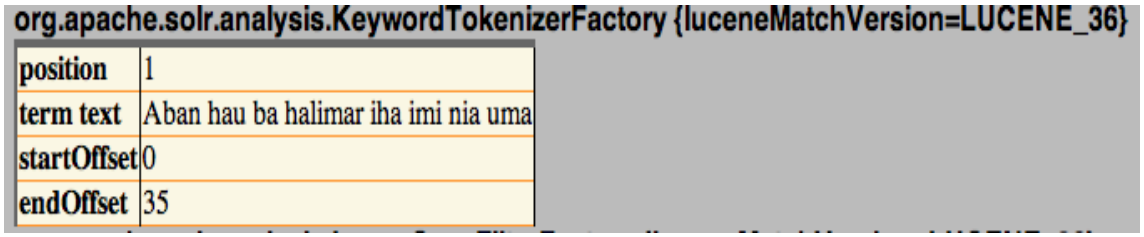


Figura 3.4: Resultado do Keyword Tokenizer.

Lower Case Tokenizer

Este simbolizador é para analisar o fluxo da entrada de palavras, delimita os não letras e depois converte todas as letras em minúscula, e, por último, vai rejeitar todos os espaço brancos e não letras.

```
<analyzer>
        <tokenizer class="solr.LowerCaseTokenizerFactory"/>
</analyzer >
```

Exemplo:
Entrada:

ABAN HÁ'U BA HALIMAR IHA IMI NIA UMA

Saída:

org.apache.solr.analysis.LowerCaseTokenizerFactory {luceneMatchVersion=LUCENE_36}

position	1	2	3	4	5	6	7	8
term text	aban	hau	ba	halimar	iha	imi	nia	uma
startOffset	0	5	9	12	20	24	28	32
endOffset	4	8	11	19	23	27	31	35

Figura 3.5: Resultado do LowerCase Tokenizer.

Whitespace Tokenizer

Este é o simbolizador que divide o fluxo do texto em espaço branco e retorna as sequências de carácter que não são espaço branco, incluindo as pontuações como símbolos.

```
<analyzer>
        <tokenizer class="solr.WhitespaceTokenizerFactory"/>
</analyzer>
```

Exemplo:
Entrada:

Keta haluha, aban ha'u ba halimar iha imi nia uma!

Saída:

org.apache.solr.analysis.WhitespaceTokenizerFactory {luceneMatchVersion=LUCENE_36}

position	1	2	3	4	5	6	7	8	9	10
term text	Keta	haluha,	aban	hau	ba	halimar	iha	imi	nia	uma!
startOffset	0	5	13	18	22	25	33	37	41	45
endOffset	4	12	17	21	24	32	36	40	44	49

Figura 3.6: Resultado do Whitespace Tokenizer.

3.2.3 Filtragem

Lower Case Filter

Converte as letras maiúsculas num símbolo para que seja equivalente em letras minúsculas e todos os outros caracteres, como pontuações, não alterados.

```
<analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.LowerCaseFilterFactory"/>
</analyzer>
```

Exemplo:
Entrada:

ABAN HAU BA HALIMAR IHA IMI NIA UMA

Saída:

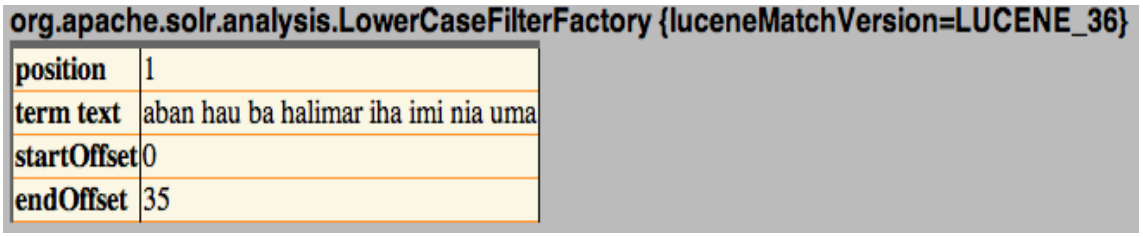


Figura 3.7: Resultado do LowerCase Tokenizer.

Stop Filter

Uma maneira de melhorar o desempenho do sistema de recuperação de informação é eliminar as palavras paradas durante o processo de indexação automática. Esta filtragem simples do Apache Solr, chamado *StopFilterFactory*, tem a função de remover as palavras paradas encontradas durante o processo de consulta.

```
<analyzer type="query">
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.StopFilterFactory" ignoreCase="true"
        words="lang/stopwords_tt.txt" enablePositionIncrements="true" />
</analyzer>
```

Exemplo:
Entrada :

Keta haluha, aban ha'u ba halimar iha imi nia uma

Saída:

"aban", "ba", "iha", "imi", "nia", estão na lista das palavras paradas (*stopword*) e foram removidas automaticamente no processo de indexação também na consulta.

org.apache.solr.analysis.StopFilterFactory {words=lang/stopwords_tt.txt, ignoreCase=true,

position	1	2	4	6	10
term text	Keta	haluha	ha'u	halimar	uma
startOffset	0	5	18	26	46
endOffset	4	11	22	33	49
type	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>	<ALPHANUM>

Figura 3.8: Resultado do Stop Filter.

Synonym Filter

Esta filtragem do sinónimo é usada quando fazemos uma pesquisa ou consulta usando uma palavra que não esteja no documento original, mas é sinónimo de uma palavra que está indexada e você quer que o documento coincida com a consulta.

```
<analyzer type="query">
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filtre class="solr.SynonymFilterFactory" synonyms="synonyms.txt"
        ignoreCase="true" expand="true"/>
</analyzer>
```

Exemplo:
Entrada:

Keta haluha, aban ha'u ba halimar iha imi nia uma

Saída:

halimar → halimar, halimar ona, sei halimar

org.apache.solr.analysis.SynonymFilterFactory {synonyms=synonyms.txt, expand=false, ignoreCase=true, luceneMatchVersion=LUCENE_36}

position	1	2	3	4	5	6	7	8
term text	Aban	hau	ba	halimar	iha	imi	nia	uma
				halimar	ona			
				sei	halimar			
type	word	word	word	SYNONYM	word	word	word	word
				SYNONYM	SYNONYM			
				SYNONYM	SYNONYM			
startOffset	0	5	9	12	20	24	28	32
				12	20			
				12	20			
endOffset	4	8	11	19	23	27	31	35
				19	23			
				19	23			

Figura 3.9: Resultado do Synonym Filter.

Keyword Marker Filter Factory

Protege as palavras modificadas por derivações. Uma lista de palavras personalizadas e protegidas pode ser especificada com o atributo "protegido"no esquema. Qualquer palavra na lista de palavras protegidas não será modificada por qualquer derivação do Solr.

```
<fieldtype name="myfieldtype" class="solr.TextField">
 <analyzer>
        <tokenizer class="solr.WhitespaceTokenizerFactory"/>
        <filter class="solr.KeywordMarkerFilterFactory"
        protected="protwords.txt" />
 </analyzer>
</fieldtype>
```

Exemplo:
Entrada:

abanbainrua, tenke kuidadu ita nia meio-ambiente

Saída:
"abanbainrua"esta na lista da palavras protegidas (*protwords.txt*)

org.apache.solr.analysis.KeywordMarkerFilterFactory {protected=protwords.txt, luceneMatchVersion=LUCENE_36}

position	1	2	3	4
term text	abanbainrua	tenke	kuidadu	meioambiente
keyword	true	false	false	false
startOffset	0	13	19	35
endOffset	11	18	26	48
type	word	word	word	word

Figura 3.10: Resultado do Keyword Marker Filter .

Remove Duplicate Token Filter

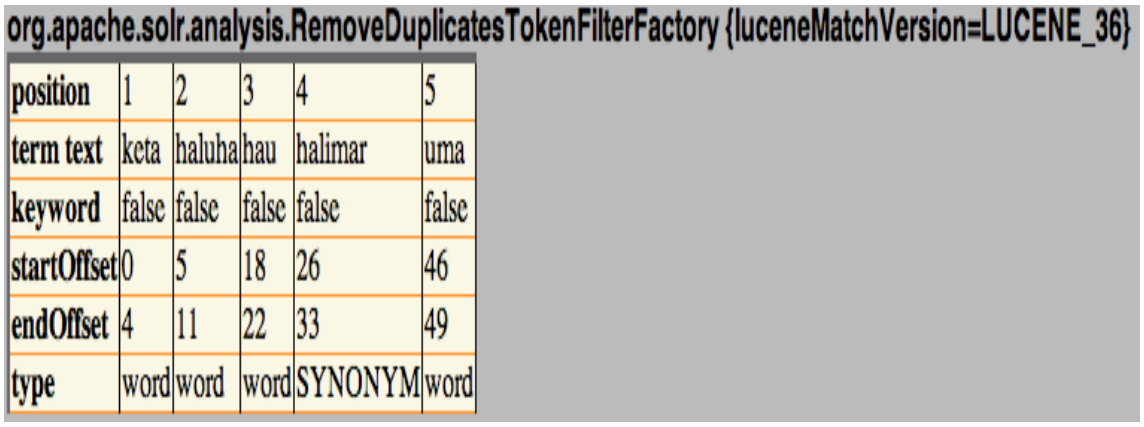
A filtragem remove duplicatas símbolos no fluxo. Os símbolos são considerados duplicatas se eles têm o mesmo texto e valores de posição.

```
<analyzer>
        <tokenizer class="solr.StandardTokenizerFactory"/>
        <filter class="solr.SynonymFilterFactory" synonyms="synonym.txt"/>
        <filter class="solr.RemoveDuplicatesTokenFilterFactory"/>
</analyzer>
```

Exemplo:
Entrada:

Aban ha'u ba halimar iha imi nia uma

Saída: Esta saída é a continuação do Synonym Filter.



org.apache.solr.analysis.RemoveDuplicatesTokenFilterFactory {luceneMatchVersion=LUCENE_36}

position	1	2	3	4	5
term text	keta	haluha	hau	halimar	uma
keyword	false	false	false	false	false
startOffset	0	5	18	26	46
endOffset	4	11	22	33	49
type	word	word	word	SYNONYM	word

Figura 3.11: Resultado do Remove Duplicates Token Filter.

3.2.4 Indexação dos Documentos

Como já tinha dito no capítulo anterior, o solr-ruby encapsula as operações fundamentais do Solr. Por esse motivo, o autor decidiu usar essa linguagem para indexar os documentos que vão ser usados no sistema.

```
require 'net/http'

@dir = Dir.new("/Users/patant/Documents/apache-solr-3.6.0/SRITetum/documentos")
@url = URI.parse("http://localhost:8983/solr")
@connection = Net::HTTP.new(@url.host, @url.port)
def index(nomedodocumento)
@connection.get(@url.path + "/update/extract?stream.file=#{ nomedodocumento }
&literal.id=#{ nomedodocumento }")
end
def commit
   @connection.get(@url.path + "/update?commit=true")
end
@dir.each {|name|f = "#{@dir.path}/#{name}"
   if File.file?(f)
     puts "Indexar#{f}..."
      index(f)
   end
  }
puts "Esta a mandar os documentos"
commit
puts "Terminou!!!!"
```

O programa escrito acima é o programa utilizado para indexar os documentos no solr, basicamente este processo de indexação é igual ao XML via HTTP POST, o que diferencia é que, através deste programa, o autor conseguiu indexar todos os documentos de uma só vez.

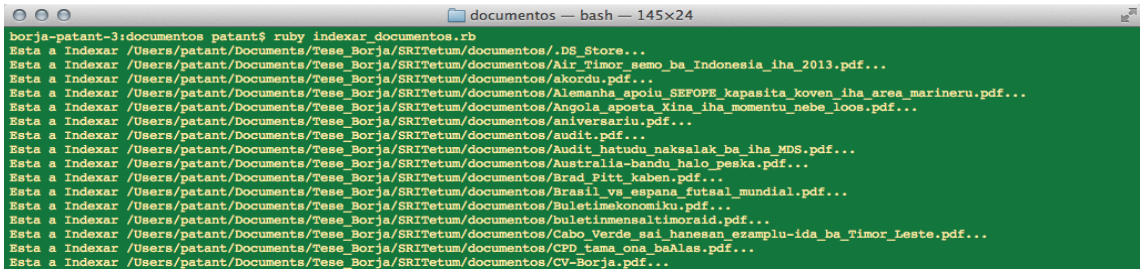


Figura 3.12: Processo de indexação de documentos utilizando Ruby

Capítulo 4

Teste de Desempenho do Sistema Desenvolvido

Para saber se o sistema desenvolvido funciona bem, é preciso fazer um teste de avaliação do desempenho a si próprio e também versus outro sistema atualmente utilizado. Na seção anterior o autor mencionou que escolheu duas grandes máquinas *Google* e *Bing* para avaliar o sistema desenvolvido. No entanto, neste capítulo, vai falar sobre os resultados dos testes feitos e também da análise do resultado dos testes.
Existem diferentes tipos de consultas que podem ser utilizados por SRI, dependendo do modelo de recuperação que o sistema adote, por exemplo, um sistema *full* text não irá responder ao mesmo tipo de consulta de um sistema baseado em ordenação de palavras-chaves.
Uma constatação importante é que a maioria das linguagens de consulta tenta usar o conteúdo (semântica) e a estrutura (sintaxe) do texto da consulta para encontrar documentos que são relevantes:

- consultas que podem ser formuladas com linguagens de consultas baseadas em palavras-chaves *(keyword-based)*;
- um tipo mais complexo de consulta envolvendo o casamento de padrões *(pattern matching)*;
- consultas em estruturas mais dependentes dos modelos de recuperação;
- padrão de protocolos usados na Internet.

Antes de avançar para os resultados testes que foram feitos para avaliar o sistema desenvolvido, o autor quer relembrar um pouco o resultado da pesquisa que também foi a ideia

base para desenvolver este sistema, no resultado da pesquisa que o autor fez, notou que a maioria ou seja 80
% do sistema de pesquisa que o autor usou para fazer pesquisa *(Google, Bing, Ask)* não reconhece ou deteta as pesquisas por palavras do tétum mas ao contrário com pesquisa por frase os sistemas deram resposta correta mais de 80% e por isso, baseando neste resultado da pesquisa, o autor decidiu fazer teste de desempenho para o sistema só por palavra.
Os documentos indexados para testar o desempenho da sistema desenvolvida tambem foram indexados nos dois sistemas de autor escolheu para fazer testes (*Google e Bing*). os documentos que foram indexados são documentos de varios tipos e de diversas linguas como Português, Inglês, Indonesia, e o principal Tétum. Todas as palavras usadas para avaliar e analisar o desempenho dos sistemas foram palavras mais usadas para escrever notícias e conversas do dia a dia. O método usada para avaliar e analisar o desempenho do sistema desenvolvido é usar a mesma palavra para todos os sistemas e verificar os resultados da pesquisa até encontrar os erros da pesquisa, calculados com a seguinte fórmula:

$$\%erro/certo = \frac{\sum dp}{\sum derc} X100\%$$

Ou seja, a percentagem do erro ou certo da pesquisa é calculada através do somatório de cada linha do resultado (cada linha é igual a 1) e divido por somatório do erro ou certo encontrado na pesquisa vezes 100%.

4.1 Testes de pesquisa

Como já tinha dito antes, os testes foram feitos para sistema desenvolvido e também para sistema que atualmente é usado ou mais popular.

4.1.1 Pesquisa no Google

Nesta seção vamos ver o resultado da pesquisa feito no Google.

Tabela 4.1: Tabela do resultado de pesquisa com palavras tétum no GOOGLE

Palavras	**GOOGLE**					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
kolaborasaun	10	100,00%	0	0,00%	10	100%
populasaun	26	100,00%	0	0,00%	26	100%
televisaun	2	100,00%	0	0,00%	2	100%
notisias	9	100,00%	0	0,00%	9	100%
eleisaun	14	100,00%	0	0,00%	14	100%

Continua na próxima página

Tabela 4.1 – *Continuação da página anterior*

Palavaras	**GOOGLE**					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
kadeira	13	100,00%	0	0,00%	13	100%
suratahan	1	100,00%	0	0,00%	1	100%
loron	84	100,00%	0	0,00%	84	100%
matan	38	100,00%	0	0,00%	38	100%
eksportasaun	2	100,00%	0	0,00%	2	100%
di'ak	51	100,00%	0	0,00%	51	100%
inan	23	100,00%	0	0,00%	23	100%
edifisiu	8	100,00%	0	0,00%	8	100%
han	19	100,00%	0	0,00%	19	100%
haksolok	8	100,00%	0	0,00%	8	100%
hemu	8	100,00%	0	0,00%	8	100%
kanta	5	100,00%	0	0,00%	5	100%
hamutuk	75	100,00%	0	0,00%	75	100%
hakerek	37	100,00%	0	0,00%	37	100%
haburas	8	100,00%	0	0,00%	8	100%
hakmatek	7	100,00%	0	0,00%	7	100%
Tuda-malu	5	100,00%	0	0,00%	5	100%
Baku-malu	7	100,00%	0	0,00%	7	100%
hanoin	60	100,00%	0	0,00%	60	100%
kondekorasaun	5	1000,00%	0	0,00%	5	100%
Ai-laran	7	100,00%	0	0,00%	7	100%
funu	17	100,00%	0	0,00%	17	100%
hasoru	37	100,00%	0	0,00%	37	100%
Ai-hun	4	100,00%	0	0,00%	4	100%
soe	13	100,00%	0	0,00%	13	100%
hamlaha	6	100,00%	0	0,00%	6	100%
bosu	2	100,00%	0	0,00%	2	100%
hanorin	17	100,00%	0	0,00%	17	100%
edukasaun	23	100,00%	0	0,00%	23	100%
tasi	17	100,00%	0	0,00%	17	100%
biblioteka	5	100,00%	0	0,00%	5	100%
kareta	13	100,00%	0	0,00%	13	100%
sasan	24	100,00%	0	0,00%	24	100%
selu	24	100,00%	0	0,00%	24	100%
osan	34	100,00%	0	0,00%	34	100%
eskola	10	100,00%	0	0,00%	10	100%
kamisa	1	100,00%	0	0,00%	1	100%
kartaun	8	100,00%	0	0,00%	8	100%

Continua na próxima página

Tabela 4.1 – *Continuação da página anterior*

Palavaras	GOOGLE					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
MEDIA TOTAL	**CERTOS** 100,00%		**ERRADOS** 0,00%			

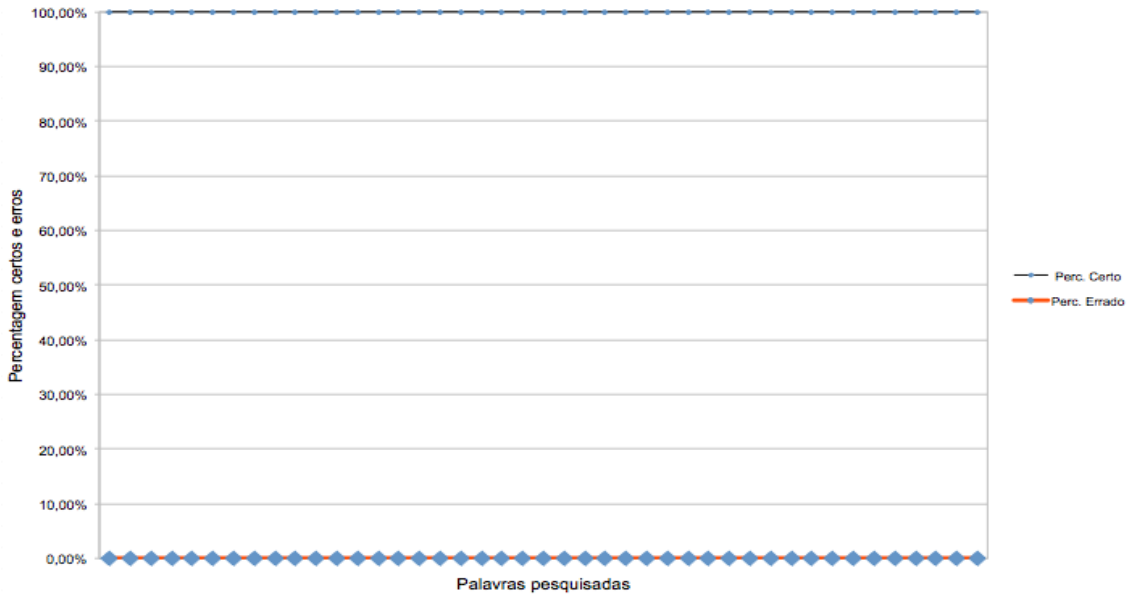


Figura 4.1: Resultado da pesquisa feito no Google representado graficamente.

4.1.2 Pesquisa no Bing

Nesta seção vamos ver o resultado da pesquisa feito no Bing.

Tabela 4.2: Tabela do resultado de pesquisa com palavras tétum no BING

Palavras	BING					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
kolaborasaun	10	100,00%	0	0,00%	10	100%
populasaun	26	100,00%	0	0,00%	26	100%
televisaun	2	100,00%	0	0,00%	2	100%
notisias	9	100,00%	0	0,00%	9	100%

Continua na próxima página

Tabela 4.2 – *Continuação da página anterior*

Palavaras	**BING**					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
eleisaun	14	100,00%	0	0,00%	14	100%
kadeira	13	100,00%	0	0,00%	13	100%
suratahan	1	100,00%	0	0,00%	1	100%
loron	84	100,00%	0	0,00%	84	100%
matan	38	100,00%	0	0,00%	38	100%
eksportasaun	2	100,00%	0	0,00%	2	100%
di'ak	51	100,00%	0	0,00%	51	100%
inan	23	100,00%	0	0,00%	23	100%
edifisiu	8	100,00%	0	0,00%	8	100%
han	19	100,00%	0	0,00%	19	100%
haksolok	8	100,00%	0	0,00%	8	100%
hemu	8	100,00%	0	0,00%	8	100%
kanta	5	100,00%	0	0,00%	5	100%
hamutuk	75	100,00%	0	0,00%	75	100%
hakerek	37	100,00%	0	0,00%	37	100%
haburas	8	100,00%	0	0,00%	8	100%
hakmatek	7	100,00%	0	0,00%	7	100%
Tuda-malu	5	100,00%	0	0,00%	5	100%
Baku-malu	7	100,00%	0	0,00%	7	100%
hanoin	60	100,00%	0	0,00%	60	100%
kondekorasaun	5	1000,00%	0	0,00%	5	100%
Ai-laran	7	100,00%	0	0,00%	7	100%
funu	17	100,00%	0	0,00%	17	100%
hasoru	37	100,00%	0	0,00%	37	100%
Ai-hun	4	100,00%	0	0,00%	4	100%
soe	13	100,00%	0	0,00%	13	100%
hamlaha	6	100,00%	0	0,00%	6	100%
bosu	2	100,00%	0	0,00%	2	100%
hanorin	17	100,00%	0	0,00%	17	100%
edukasaun	23	100,00%	0	0,00%	23	100%
tasi	17	100,00%	0	0,00%	17	100%
biblioteka	5	100,00%	0	0,00%	5	100%
kareta	13	100,00%	0	0,00%	13	100%
sasan	24	100,00%	0	0,00%	24	100%
selu	24	100,00%	0	0,00%	24	100%
osan	34	100,00%	0	0,00%	34	100%
eskola	10	100,00%	0	0,00%	10	100%
kamisa	1	100,00%	0	0,00%	1	100%

Continua na próxima página

Tabela 4.2 – *Continuação da página anterior*

Palavaras	BING					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
kartaun	8	100,00%	0	0,00%	8	100%
MEDIA TOTAL	**CERTOS** 100,00%		**ERRADOS** 0,00%			

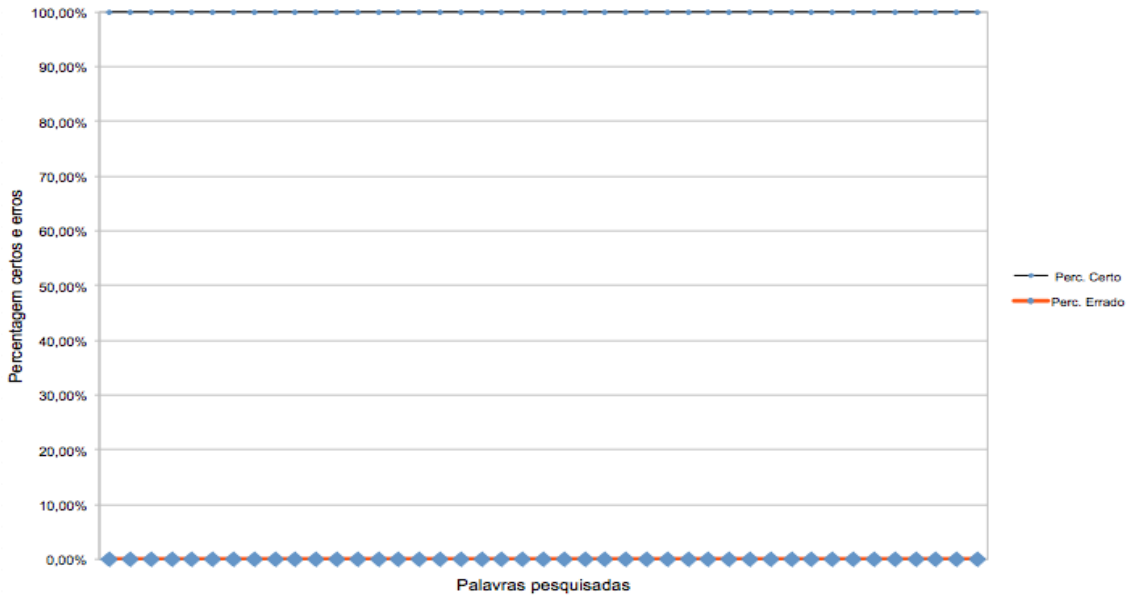


Figura 4.2: Resultado da pesquisa feito no Bing representado graficamente.

4.1.3 Pesquisa no SRI desenvolvido

Nesta seção vamos ver o resultado da pesquisa feito no SRI tétum. Para testar o desempenho através das pesquisas, antes disso foram indexadas 144 documentos[1] de diferentes tipos como word, pdf, e xml.

[1]Os documentos foram recolhidos de vários *sites* popular das notícias em língua tétum como timorhauniadoben.com, temposemanal.com, suaratimorlorasae.com, sapo.tl.

Tabela 4.3: Tabela do resultado de pesquisa com palavras tétum no sistema desenvolvida

Palavras	**SRI_Tetum**					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
kolaborasaun	9	100,00%	0	0,00%	9	100%
populasaun	25	100,00%	0	0,00%	25	100%
televisaun	1	100,00%	0	0,00%	1	100%
notisias	7	100,00%	0	0,00%	7	100%
eleisaun	10	100,00%	0	0,00%	10	100%
kadeira	10	100,00%	0	0,00%	10	100%
suratahan	0	0,00%	0	0,00%	0	0%
loron	84	100,00%	0	0,00%	84	100%
matan	37	100,00%	0	0,00%	37	100%
eksportasaun	1	100,00%	0	0,00%	1	100%
di'ak	7	100,00%	0	0,00%	7	100%
inan	23	100,00%	0	0,00%	23	100%
edifisiu	6	100,00%	0	0,00%	6	100%
han	132	100,00%	0	0,00%	132	100%
haksolok	132	100,00%	0	0,00%	132	100%
hemu	132	100,00%	0	0,00%	132	100%
kanta	4	100,00%	0	0,00%	4	100%
hamutuk	74	100,00%	0	0,00%	74	100%
hakerek	37	100,00%	0	0,00%	37	100%
haburas	7	100,00%	0	0,00%	7	100%
hakmatek	4	100,00%	0	0,00%	4	100%
Tuda-malu	4	100,00%	0	0,00%	4	100%
Baku-malu	5	100,00%	0	0,00%	5	100%
hanoin	58	100,00%	0	0,00%	58	100%
kondekorasaun	5	100,00%	0	0,00%	5	0%
Ai-laran	5	100,00%	0	0,00%	5	100%
funu	17	100,00%	0	0,00%	17	100%
hasoru	35	100,00%	0	0,00%	35	100%
Ai-hun	3	100,00%	0	0,00%	3	100%
soe	132	100,00%	0	0,00%	132	100%
hamlaha	5	100,00%	0	0,00%	5	100%
bosu	1	100,00%	0	0,00%	1	100%
hanorin	15	100,00%	0	0,00%	15	100%
edukasaun	20	100,00%	0	0,00%	20	100%
tasi	17	100,00%	0	0,00%	17	100%
biblioteka	4	100,00%	0	0,00%	4	100%
kareta	10	100,00%	0	0,00%	10	100%

Continua na próxima página.

Tabela 4.3 – *Continuação da página anterior*

Palavaras	**SRI_Tetum**					
	CERTOS	%	ERRADOS	%	TOTAL PESQ.	%
sasan	21	100,00%	0	0,00%	21	100%
selu	21	100,00%	0	0,00%	21	100%
osan	31	100,00%	0	0,00%	31	100%
eskola	19	100,00%	0	0,00%	19	100%
kamisa	0	0,00%	0	0,00%	0	0%
kartaun	7	100,00%	0	0,00%	7	100%
MEDIA TOTAL	**CERTOS** 95,35%		**ERRADOS** 0,00%			

Na tabela em cima, mostrou que o somatório da media do resultado das pesquisas erradas e resultados das pesquisas que acertaram não foi 100%, porque o sistema não consegue detectar algumas palavras pesquisadas nos documentos indexados.

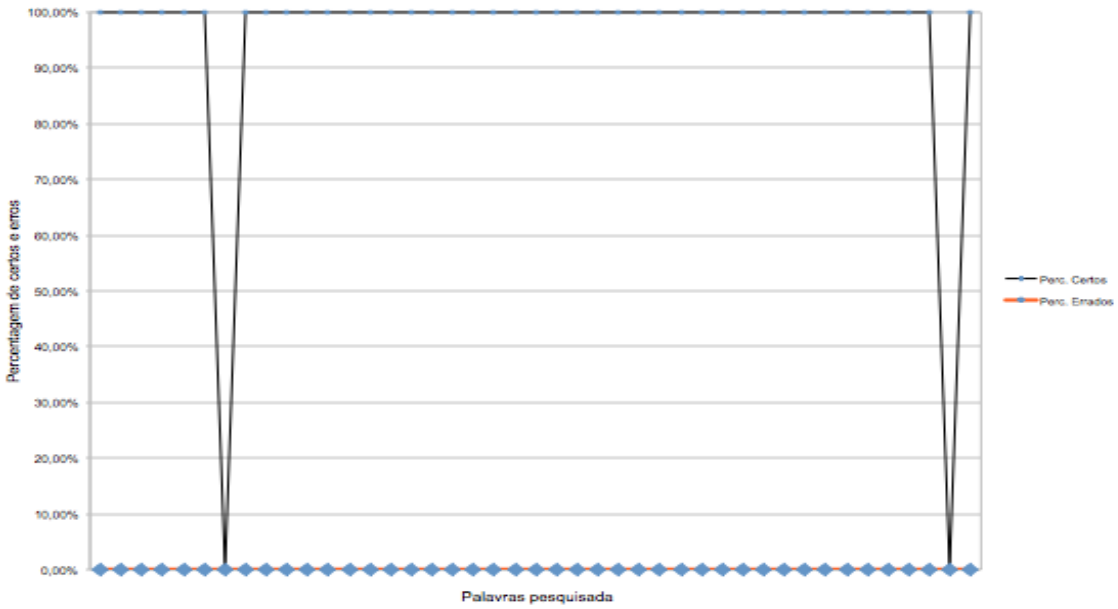


Figura 4.3: Resultado da pesquisa feito no SRI representado graficamente.

4.2 Comparação dos resultados

Para facilitar uma comparação o autor criou uma tabela que mostra todos os resultados da pesquisa.

•	CERTO	ERRO
GOOGLE	100%	0%
BING	100%	0%
SRI_TETUM	95,35%	0%

Tabela 4.4: Tabela de comparação dos resultados

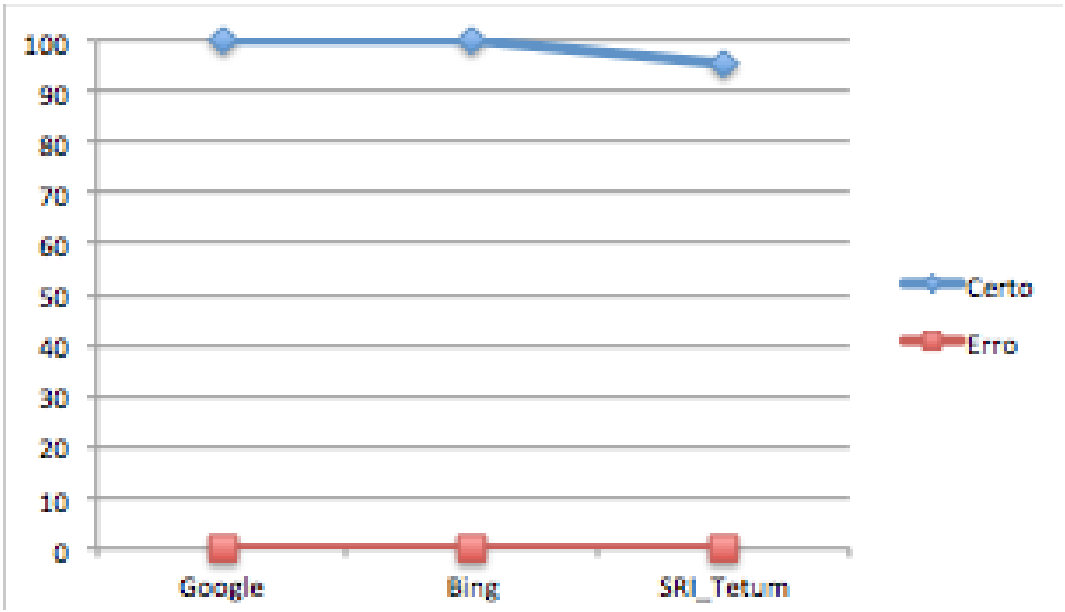


Figura 4.4: Comparação do resultado representado graficamente.

A tabela e o gráfico acima demostram havia grandes diferença de resultados entre as duas grandes máquinas de pesquisa e sistema desenvolvido. O resultado mostra que o sistema desenvolvido consegiu reagir como os dois grandes sistemas mesmo que não consegiu ter resultado perfeito como dois grandes sistemas, sendo assim o sistema desenvolvido tem alguns vantagens que os dois grandes sistema, por exemplo alguns resultados da pesquisa dos dois sistemas é diferente com do sistema desenvelvovido porque os dois sistemas não conseguiram recuperar todas as informações de algumas perguntas dadas, isso acontece porque não existe os sinonimos da palavras pesquisadas nos dois sistemas, pelo contrario no sistema desenvolvido existem os sinonimos para palavras em lingua tétum.

Capítulo 5

Conclusões e trabalho futuro

Neste capítulo apresenta as principais conclusões, as limitações do estudo e as sugestões para trabalho futuro.

5.1 Principais conclusões do trabalho

Apache Solr é uma ferramenta muito poderosa que construída pelo projeto *Apache Lucene*, gostou de trabalhar com esta ferramenta mesmo que seja nova para si. Trabalhou de forma intensiva e esta ferramenta quase nunca se decepcionou, o autor sabe que deve aprender muito mais coisas para conseguir compreender profundamente como é que esta ferramenta trabalha. Nota-se que este sistema não faz uma correção ortográfica porque a ferramenta Apache Solr é muito dedicado à língua inglesa e outras línguas que estruturalmente clara, mesmo assim não desisti de desenvolver este sistema até finalizar este sistema. Mas de tudo isso, baseado nos resultados encontrados neste presente trabalho, conclui-se que SRI desenvolvido através da ferramenta *Apache Solr* conseguiu atingir o objetivo inicial deste presente trabalho que é facilitar os utilizadores timorenses de fazer pesquisa com língua tétum, porque conseguiu ultrapassar as duas grandes máquinas de pesquisa na apresentação dos resultados da pesquisa feitos para testar desempenho deste sistema.

5.2 Limitações do estudo

Este trabalho contém algumas limitações, metodologicamente este estudo foi elaborado de forma concisa e objetiva, baseado essencialmente nos diversos relatórios e documentos

produzidos por diversos autores. Por outro, falta de informações sobre a língua tétum também deram impacto ao processo de construção técnica para este sistema. E por fim, a ferramenta utilizada é apropriada para a língua inglesa e outras línguas com estruturas claras, por isso para adaptar à língua tétum é preciso mais estudo aprofundado desde a criação de novas técnicas para língua tétum baseada na biblioteca do *Apache Lucene*.

5.3 Trabalho Futuro

Este trabalho é a base para a criação um novo sistema de recuperação de informação que pode competir com grandes sistemas atuais, por isso para o trabalho futuro é preciso fazer uma investigação mais profunda na estrutura da língua tétum principalmente na parte gramatical, além disso para trabalho futuro propõe-se uma integração do *Apache Solr* com *Drupal* para criar uma interface gráfica no sistema.

Bibliografia

[Ams] Amit, S., *Modern Information Retrieval: A Brief Overview*, Google.Inc.

[Ash] Ashish, Y., *Apache Solr - I can haz Search*, Barcamp 5, Chennai, Dec 2010, disponivel em http://www.slideshare.net/ashish0x90/apache-solr-6031297, acessado em Jan 2013.

[Avi2010] Avi, R., "Indexing Text and HTML Files with Solr, the Lucene Search Server", *A Lucid Imagination Technical Tutorial*, pp.1-3, Fev 2010.

[Bla] Blair, D.C., Maron, M.E., "Full text information retrieval: Further analysis and clarification", *Information Processing and Management*,pp. 437-447, 1990.

[Chr] Chris, H., "Faceted Searching with Apache Solr", *hossman-apache-org*, Oct 2006.

[Cat] Catharina, W. K., "Tetun 2", *Sentru Lingua*, Dili Institute of Technology, Oct 2008.

[Dav] Davi, B.A., Esboço Gramatical do Tétum Prasa : Língua Oficial de Timor-Leste, *Dissertação do Mestrado em Linguística, Instituto de Letras, Departamento da Linguística, Português e Línguas Clássicas*, Universidade Brasilia.

[David] David, S., Eric, P., "Apache Solr 3 Enterprise Search Server : Enhance your search with faceted navigation, result highlighting, relevancy ranked sorting, and more", *Open-source community experience distilled*, Packt Publishing, Birmingham-Mumbai, 2012.

[Esp] Esperança, J. P. T., "Um brevissimo olhar sobre a Literatura de Timor", *Mealibra – Revista de Cultura*, Viana do Castelo (Portugal), Centro Cultural do Alto Minho, Série 3 (16), Verão 2005, p. 131-134.

[Erik] Erik, H., "Solr Application Development Tutorial", *Lucid Imagination*, Jul 2011, disponivel em http://www.slideshare.net/erikhatcher/solr-application-development-tutorial, acessado em Nov 2012.

[Erik] Erik, H., "Indexing rich files into Solr, quickly and easily", Aug 2011, disponivel em http://searchhub.org/2011/08/31/indexing-rich-files-into-solr-quickly-and-easily/, acessado em Nov 2012.

[Fil] Filipe, H., "Como funciona uma maquina de pesquisa? Crawlers/Spiders/Robô/Coletores", *Blog do Hummel – Tecnologias e coissa aleatorias*, disponivel em http://www.blogdohummel.com/2007/10/como-funciona-uma-mquina-de-pesquisa_17.html, accesado no 22 de Janeiro 2012.

[Fab] Fabrício, J. B., "Indexing, searching, and faceted browsing with Solr", *Search smarter with Apache Solr, Part 1: Essential features and the Solr schema*", IBM developerWorks, May 2007, disponivel em http://www.ibm.com/developerworks/java/library/j-solr1/ accesado no 22 de Nov 2012.

[Gra] Grant, I., "*Uma breve introdução ao tema Recuperação de Informação*", São Paulo, 2010, disponivel em http://fbarth.net.br/materiais/introducaoRecuperacaoInformacao/introducaoRecuperacaoInformacao.pdf, accesado no 22 de Janeiro 2012.

[Hull] Geoffrey, H., "*Mai Kolia Tétum: A Beginner's Course in Tétum Praça (The Lingua Franca of East Timor)*", Revised Edition, Sebastião Aparicio da Silva Project, 2003.

[Hull] Geoffrey, H., "Manual de Língua Tétum para Timor Leste", *Sebastião Aparício da Silva Project for the protection and promotion of East Timorese Languages.* 2004.

[Hull] Geoffrey, H., "The Languages of East Timor : Some Basic Facts", *Instituto Nacional de Linguística (INL)*, Universidade Nacional de Timor Lorosa'e, Feb 2002.

[Icc] Iccha, S., Serdar, A., Dr. Edward, F., "Apache Solr: Indexing and Searching Information Storage and Retrieval", *Virginia Polytechnic Institute and State University*, Blacksburg, VA 24061 USA, disponível em http://curric.dlib.vt.edu/modDev/package_modules/MidtermModuleTeam1-Solr.pdf, acessado no 22 de Janeiro de 2013.

[INL] Instituto Nacional de Linguística, "*A Ortografia Padronizada do Tétum Os Seus 115 anos de Construção*, Aug 2004.

[INL] Instituto Nacional de Linguística, "*Matadalan Ortográfiku ba Lia-Tetun (Lista-badak)*".

[Kel] Kelvin T., "Solr tutorial", *Kelvin Tan - Lucene Solr ElasticSearch consultant*, Flexile Vertical Search Crawler, disponivel em http://www.solrtutorial.com/, acessado em Nov 2012.

[Leo] Leon, M., "Information Retrieval & Apache Solr Use Case", *Netherland Bioinformatics Centre*, May 2010.

[Luc] Lucid Imagination, "*Lucid works for solr : Apache Solr reference guide*".

[Lui] Luis, M. E. R., "Apache Solr, Un Motor de Busqueda de Codigo Abierto", *Revista Digital Universitaria*, 1 de noviembre 2012, Volumen 13 Numero 11, ISSN: 1067-6079, UNAM.

[Mad] Madian, K., Stephen, C., Sagnik, R. C., Lee, C. G., "A Framework for Bridging the Gap Between Open Source Search Tools", *The Pennsylvania State University.*

[Mar] Marta, R. C., Rafael, E. B., Jens, G., Joan, C., "Plagiarism detection using information retrieval and similarity measures based on image processing techniques" *Barcelona Media – Innovation Center.*

[Marv] Marvin, O. S., "Processamento de Linguagem Natural (PLN)", *Curso De Mestrado Em Sistemas De Computação*, Pontifícia Universidade Católica De Campinas.

[Nick] Nick, V., "Improving Acquia Search and the Apache Solr Search Integration Drupal Module", *Master Thesis in Information Technology*, Barcelona School of Informatics (FIB), 2012.

[NN] ..., "Apache Solr: Indexing and Searching", *(Draft last modified 10/26/2010)*, disponivel em http://curric.dlib.vt.edu/modDev/package_modules/MidtermModuleTeam1-Solr.pdf. acessado em Jan 2013.

[Olin] Olinda, N. P. C., "Recuperação de Informação", Universidade Federal de Lavras, 2010, disponivel em http://www.dcc.ufla.br/infocomp/artigos/v2.1/art07.pdf, acessado em Nov 2012.

[Otis] Otis, G., Erik, H., Manning, "*A guide to the Java search engine : Lucene in Action*", 2005.

[Pat] Antonino, B. L. C. P., "Avaliação das pesquisas com Lingua Tétum (Lingua Official de Timor-Leste) no GOOGLE, BING e ASK", *Curso De Mestrado Em Engenharia Informatica*, Universidade de Évora. June 2010

[Phil] Philip, M., Vivien P., "Cross-concordances: terminology mapping and its effectiveness for information retrieval", *GESIS Social Science Information Centre (GESIS-IZ)*, Bonn, Germany.

[Pust] Pustejovsky J., Boguraev B., "Lexical knowledge representation and natural language processing", *Natural Language Processing (Fernando C. N. Pereira and Barbara J. Grosz)*, pp 193-223.

[Raf] Rafal Kuc, "*Apache Solr 3.1 Cookbook : Over 100 recipes to discover new ways to work with Apache's Enterprise Search Server*"Firts Edition, Packt Publishing Ltd, 2011.

[Reg] Regina, H. P. de B., "*Temas para a compreensão do atual quadro linguístico de Timor Leste*". Ciências & Letras, Porto Alegre, n. 48, p. 175-194, jul./dez. 2010, disponível em: http://seer1.fapa.com.br/index.php/arquivos, acessado Nov 2012.

[Ron] Roni, R. R. F., "*Recuperacão de Informações por similaridade de fonemas adaptada á Lingua Portuguesa*", disponivel em http://www.uniritter.edu.br/graduacao/informatica/sistemas/downloads/tcc2k9/TCCII_2009_1_Roni.pdf, acessado em Novembro 2012.

[Rui] Rui, G. F., "*Língua, nome e identidade numa situação de plurilinguismo concorrencial: o caso de Timor-Leste*".

[Souza] Souza, R. R., "Sistemas de recuperação da informações e mecanismos de pesquisa na Web", *Perspectivas em Ciência da Informação*, Belo Horizonte, v. 11, n.2, p. 161-173, maio/ago. 2006.

[Solr] The Apache Software Foundation, "*Solr tutorial*", disponivel em http://svn.apache.org/repos/asf/lucene/solr/tags/release-1.1.0/site/tutorial.pdf, acessado em Nov 2012.

[Ter] Teresa, C. de F. G., "Utilização de Informação Linguística na classificação de documentos em Língua Portuguesa", *Tese Doutoramento em Informática*, Departamento de Informática, Universidade de Évora, Nov 2007.

[Tim] Tim, R., Elena, B., Pamela, F., "Java and its Role in Natural Language Processing and Machine Translation", *Dep. of Experimental Psychology, University of Granada and Dep. of Translation*, Campus Universitario de Cartuja, Spain.

[Tom] Tom, H., "Searching with Solr", *eBig Java SIG*, June 2008, disponivel em http://www.slideshare.net/tomhill/an-introduction-to-solr, acessado em Nov 2012.

[Wiki] The Apache Software Foundation, "*Solr wiki*", disponivel em http://wiki.apache.org/solr/, acessado em Nov 2012.

Anexos

Anexo A

Anexo 1

A.1 Lista dos sinonimos criados

joga → joga ona, sei joga, joga
han → han ona, sei han, han
halimar → halimar, halimar ona, sei halimar
toba → toba ona, sei toba, toba
hemu → hemu ona, sei hemu, hemu
vota → vota ona, sei vota, vota
estuda → estuda ona, sei estuda, estuda
le → le, le ona, sei le
lao → lao ona, sei lao, lao
husik → husik ona, sei husik, husik
hein → hein ona, sei hein, hein
hatais → hatais ona, sei hatais, hatais
hananu → hananu ona, sei hananu, hananu
halai → halai ona, sei halai, halai
haksolok → haksolok ona, sei haksolok, haksolok
hamnasa → hamnasa ona, sei hamnasa, hamnasa
haneruk → haneruk ona, sei haneruk, haneruk
hakilar → hakilar ona, sei hakilar, hakilar
hakneak → hakneak on a, sei hakneak, hakneak
halibur → halibur ona, sei halibur, halibur
tanis → tanis ona, sei tanis, tanis
tane → tane ona, sei tane, tane
harohan → harohan ona, sei harohan, harohan

haksoit → haksoit ona, sei haksoit, haksoit
tur → tur ona, sei tur, tur
nani → nani ona, sei nani, nani
pasiar → pasiar ona, sei pasiar, pasiar
konsulta → konsulta ona, sei konsulta, konsulta
semo → semo ona, sei semo, semo
tesi → tesi ona, sei tesi, tesi
kado → ado ona, sei kado, kado
haree → haree ona, sei haree, haree
xateia → xateia ona, sei xateia, xateia
promete → promete ona, sei promete, promete
rai → rai ona, sei rai, rai
taru → taru ona, sei taru, taru
bosok → bosok ona, sei bosok, bosok
latan → latan ona, sei latan, latan
tau → tau ona, sei tau, tau
bolu → bolu ona, sei bolu, bolu
manán → manán ona, sei manán, manán
manan → manan ona, sei manan, manan
kaer → kaer ona, sei kaer, kaer
simu → simu ona, sei simu, simu
sura → sura ona, sei sura, sura
fó → fó ona, sei fó, fó
fo → fo ona, sei fo, fo
fuma → fuma ona, sei fuma, fuma
hadeer → hadeer ona, sei hadeer, hadeer
aluga → aluga ona, sei aluga, aluga
hadi'a → hadi'a ona, hadia ona, sei hadi'a, sei hadia, hadia
re'i → re'i ona, rei ona, sei re'i, sei rei, rei
soe → soe ona, sei soe, soe
hili → hili ona, sei hili, hili
losu → losu ona, sei losu, losu
kore → kore ona, sei kore, kore
lakon → lakon ona, sei lakon, lakon
buka-hatene → buka-hatene ona, buka hatene ona, sei buka-hatene, sei buka hatene, buka hatene
pasiente → pasiente ona, sei pasiente, pasiente
impasiente → impasiente ona, sei impasiente, impasienta
sukat → sukat ona, sei sukat, sukat
sui → sui ona, sei sui, sui
arraska → arraska ona, sei arraska, arraska
nonook → nonook ona, nonok ona, sei nonook, sei nonok, nonok
dansa → dansa ona, sei dansa, dansa

hakotu → hakotu ona, sei hakotu, hakotu
fo-hatene→ fo-hatene ona, fo hatene ona, sei fo-hatene, sei fo hatene, fo hatene
toman → toman ona, sei toman, toman
nakdedar → nakdedar ona, sei nakdedar, nakdedar
taa → taa ona, ta ona, sei taa,sei ta, ta
kumprimenta → sei kumprimenta, kumprimenta ona, kumprimenta
hamulak → sei hamulak, hamulak ona, hamulak
kosar → kosar ona, sei kosar, kosar
luku → luku ona, sei luku, luku
tasak → tasak ona, sei tasak, tasak
liki → liki ona, sei liki, liki
koi → koi ona, sei koi, koi
koir → koir ona, sei koir, koir
kamat → kamat ona, sei kamat, kamat
dere → dere ona, sei dere, dere
toka → toka ona, sei toka, toka hakfodak → hakfodak ona, sei hakfodak, hakfodak
doko → doko ona, sei doko, doko

toalla → toalha, toalya
diretor → director, diretur
desempeñu → desempenhu, desempenho, desempenyu
koñesimentu → konhesimentu, kuñisementu, konhisemento
sobriñu → subrinhu, subrinho, subrinu, subrino
sobriña → subrinha, subrina

A.2 Palavras paradas & Alfabeto Tétum

A.2.1 Palavras paradas usadas

hau
o
hela
hotu
ou
la'e
nafatin
uluk
aban
ba
maske

no
nia
imi
sira
ita
hau nian
imi nian
ita nian
ami nian
o nian
sira nian
ne'é
ne'ebá
ida ne'ebá
ida ne'é
iha
ho
husi
maibe
liu ona
sei
ona
se
bainhira
aban
horiseik
kuandu
hira
nu'udar
oinsa
fahe
kontrario
hanesan

A.2.2 Alfabetos Tétum

a
b
d
e
f

g
h
i
j
k
l
ll
n
ñ
m
o
p
r
s
t
u
v
w
x
y
z

A.2.3 Palavras Protegidas

Algumas não-palavras que normalmente não serão encontrados :

envairomento
sefe
haree
bukahatene
abanbainrua
bainrua
dalabarak
dalahirak
nainrua
naintolu
bamai