|  |
| --- |
| brasao |
| Demetrius Costa Rapello  Sistema de recomendação para suporte a produção de matérias relacionadas no portal G1 |
| Dissertação de mestrado  Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre pelo Programa de Pós-graduação em Informática do Departamento de Informática do Centro Técnico e Científico da PUC-Rio.  Orientador: Prof. Marco Antônio Casanova |
|  |
| Rio de Janeiro, 19 de novembro de 2011. |

# GRNews – Sistema de recomendação de matérias de segundo nível.

GRNews é um sistema de recomendações de matérias de segundo nível que visa prover aos editores de conteúdo do portal G1 uma função para a geração de matérias relacionadas ao texto produzido. Esta geração leva em consideração a análise de informações existentes no texto. Com estas informações, o sistema estabelece critérios de consulta de conteúdo relacionado e executa estas consultas contra a base de dados indexados no sistema de busca interno **solr**. Os resultados obtidos são organizados em ordem de recenticidade e exibidos em uma listagem para o editor. Desta forma o editor pode relacionar conteúdos de diferentes critérios para compor o elemento de matérias relacionadas.

Neste capítulo discutiremos a arquitetura dos sistema, bem como os seus conponentes internos de modo a entender todo o seu funcionamento.

## Arquitetura do sistema e sumário

O principal objetivo do GRNews é gerar conteúdo relacionado a um determinado texto que é produzido pelo editor para isso o sistema conta com 3 componentes principais como pode ser visto na figura 1.

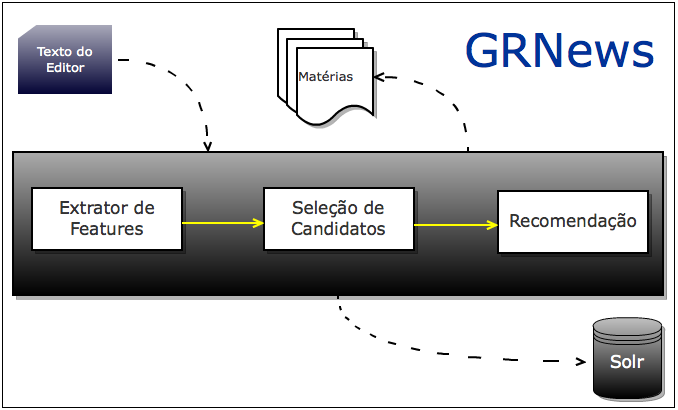


Figure 1 – GRNews arquitetura

Para ser possível acionar o motor do GRNews, é necessário que o editor informe os parâmentros de entrada e para isso, o ponto de partida é o formulário de matéria. O formulário de matéria, figura2, é apresentado ao editor como uma interface web que solicita ao editor o preenchimento de um conjunto de informações divididas em obrigatórias e não obrigatórias.

As informações obrigatórias são necessárias para configurar uma notícia que pode ser publicada.

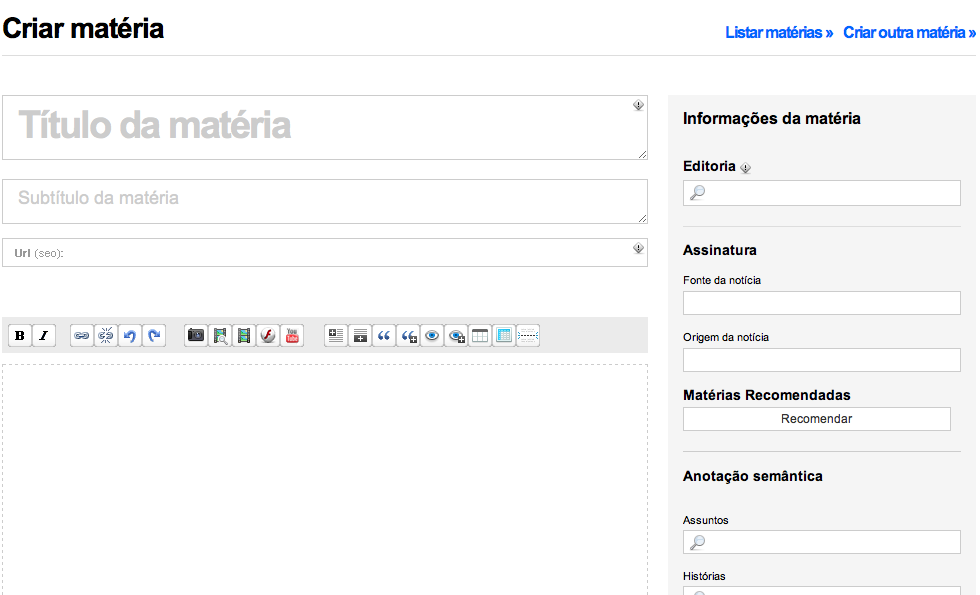


Figure 2

Dentre as informações solicitadas pelo formulário, vamos apresentar na tabela 1, apenas aquelas que serão usadas pelo sistema GRNews por entender-mos que são suficientes para realizarmos a função proposta.

Table 1

|  |  |  |
| --- | --- | --- |
| Título | O título da notícia é um dado textual limitado em 255 caracteres alfa numéricos. | Obrigatória |
| Subtítulo | O subtítulo da notícia é um dado textual limitado em 1000 caracteres alfa numéricos. | Obrigatória |
| Corpo | O corpo da notícia é um dado textual sem limite de caracteres alfa numéricos. O Editor pode fazer uso de markup html na construção do conteúdo da notícia. | Obrigatória |
| Editoria Principal | Dado que agrupa o conjunto de notícias. Esta informação é selecionada na interface através de um caixa de seleção. Este agrupamento é criado pelos editores e geralmente refletem seções do site na internet. Exemplo de Editoria para o G1: Rio de Janeiro, Economia, Mundo | Obrigatória |
| Editorias Secundárias | Agrupamentos secundários para organização das notícias | Não Obrigatória |
| Entidades Associadas | Informações sobres entidades que são passadas manualmente pelo editor para enriquecimento do conteúdo | Não Obrigatória |

Tendo preenchido o formulário de matéria de acordo como as regras descritas anteriormente, o editor pode solicitar a geração das matérias relacionadas.

A geração de matérias relacionadas é realizada através do acionamento do botão recomendar existente no lado direto do formulário de matéria, como pode ser visto na figura 2. Após a recomendação, o sistema exibe uma lista das cinco matérias encontradas ordenadas por data de publicação. Caso não existam matérias relacionadas o sistema não apresenta o elemento de listagem.

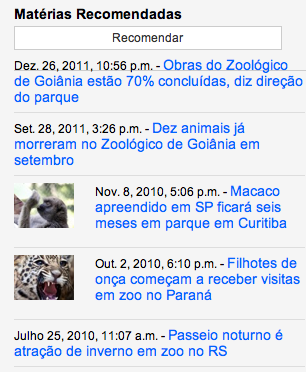


Figure 3

## Extrator de Features

Como observado no capítulo 3, features são características ou aspectos de um exemplo como por exemplo a frequência de exibição de uma determinada palavra em um texto. Escolher as melhores features é uma atividade de extrema importância para o sucesso da nossa abordagem. Abaixo descrevemos algumas das features que usaremos no projeto.

### Termos mais frequentes

É o conjunto de termos do documento ordenados por sua frequencia de aparição no texto. Segundo [irbook] a frequencia de um termo é o número de vezes que este termo aparece no documento.

A idéia é que um termo que aparece com muita frequência no texto pode indicar o tema central deste texto. Entretanto, deve-se levar em conta como dito em [nltkbook] que alguns termos que aparecem como mais frequentes podem não apresentar um significado para o texto por que são muito comuns. Para corrigir este desvio, são utilizadas técnicas de exclusão desses termos que apresentam pouco significado. Uma dessas técnicas chamada stopword list, é a elaboração de listas de exclusão baseadas nos termos mais frequentes encontrados em uma coleção de documentos.

Uma vez removidos os termos não relevantes o texto é então submetido a duas etapas em sequencia: limpeza e tokenização.

A etapa de limpeza, como pode ser visto no pseudo-código 1, consiste na remoção do ruído presente no texto, são removidos os caractéres de pontução bem como os marcadores html. Após esta etapa estamos aptos a realizar a tokenização.



pseudo-código 1

Segundo [irbook] **tokenização** é o processo de quebrar o documento em pedaços baseando-se em um padrão de corte. Durante nosso processo de tokenização, os termos são separados a medida que um espaço em branco aparece entre eles e são colocados em uma lista de termos também conhecido como **bag of word**. Neste momento, de posse da lista dos termos mais relevantes, o sistema realiza uma contagem da aparição destes termos no documento e ao final ordena a lista partindo dos termos mais frequentes para os menos frequentes.

### Tags HTML informativas

Tags informativas são tags que enfatizam a importância de um texto dentro de um documento. Entendemos que estas tags traduzem o grau de relevância que o termo possui dentro do documento. As tags reconhecidas como informativas para este trabalho são <**em**> que representa os textos escritos em itálico e a tag <**strong**> que representa dos textos escritos em negrito.

As tags informativas são procuradas dentro do documento construído pelo editor e seus textos são separados e armazenados em um vetor de termos.

De acordo com a figura 1 abaixo, serão selecionados os termos “Biblioteca Pública Dolor Barreira” e “Festival de Cinema e Cultura da Diversidade Sexual.”



### Texto em títulos de vídeos e fotos

Os termos que ocorrem dentro de estruturas que apresenta vídeos ou imagems podem representar fontes de informação valiosa para descobri ou certificar quais os temas centrais do documento.

Para isso o sistema procura pelas estruturas de mark-up que definem estes elementos afim de separar as informações existentes.

Na figura 1 o texto “**Quinta edição do festival For Rainbow começa nesta quinta-feira (27) (Foto: Divulgação**)” será selecionado.

### Entidades Nomeadas

Segundo [Mining Wiki Resources for Multilingual Named Entity Recognition] o reconhecimento de entidades nomeadas é um dos principais desafios no campo do processamento de linguagem natural. A maioria das pesquisas nesta área são restritas a um conjunto pequeno de idiomas e quase todos os métodos requerem um conhecimento linguístico refinado. Ainda, de acordo com [A Clustering Method for News Articles Retrieval System], A tarefa de extração de entidades criada nos anos 1990’s tem como objetivo o reconhecimento de unidades de informação importantes tais como: nomes de pessoas, nomes de organizações, nomes de localizações, datas, valores financeiros e etc... Diante deste cenário propomos um componente de reconhecimento de entidades que faz uso de um classificador binário para separar possíveis n-grams em entidades nomeadas.

Para o reconhecimento das entidades nomeadas nos textos das matérias, procuramos no mercado ferramentas que pudessem atender a esta demanda. A seguir apresentamos um resumo das principais ferramentas observadas durante a fase de estudos e a conclusão sobre o seu uso.

**Yahoo Term Extraction**

O serviço do yahoo de extração de termos permite a seus usuários a analise de textos fornecendo uma lista de palavras ou frases relevantes em inglês a partir de um documento em texto. O Serviço pode ser acessível através do protocolo **REST** e responde os dados de saída em formatos XML e json.

O serviço é gratuito porém, possui limite de requisições diárias em 5000 requisições. O serviço pode ser utilizado mediante o cadastramento e obtenção da chave de acesso.



pseudo-código 2

**Nltk**

É um serviço web que funciona sob o protocolo REST para mineração de texto e processamento de linguagem natural. A API foi concebida com base nas premissas do NLTK cookbook e não tem fins comerciais de modo que, possui limites tanto para o número de requisições (1000 requisições diárias) quanto para o tamanho do texto enviado (10000 caracteres). O formato de saída pode ser em XML ou json.

pseudo-código 3



Ltasks

É um serviço web que funciona sob o protocolo REST e que apresenta várias possibilidades de extração de informação do texto entre elas o reconhecimento de entidades nomeadas em lingua portuguesa. Para utilização on-line é necessária a utilização de uma chave de acesso que é obtida atravéz de um cadastro no site.



pseudo-código 4

Zemanta

É uma ferramenta concebida para geração de conteúdo relacionado para blogs, contúdo, seus idealizadores proveem uma api REST que permite a extração de entidades contextualizadas ao texto submetido. Para fazer uso do serviço é necessário um cadastro e a obtenção de uma chave de acesso. O serviço é, em princípio, independente de idioma.



pseudo-código 5

Abaixo segue a tabela comparativa das principais características das ferramentas observadas:

Table 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Características | **ltasks** | **yahoo** | **nltk** | **zemanta** |
| Tem suporte ao idioma português | sim | não | sim | sim |
| Linguagem de desenvolvimento | Java | - | Python | - |
| Limite de acesso diário | - | 5000 | 1000 | 1000 |
| Tamanho máximo do texto | - | - | 10000 | 10000 |
| Possui código aberto | não | não | não | não |

Dentre as ferramentas observadas, o serviço Ltasks foi a que apresentou o melhor aproveitamento, porém, a ausência de um código aberto para aprimoramento do algorítimo, a limitação de acessos ao serviço e o tempo gasto em cada requisição foram determinantes para que partissemos para uma abordagem própria para extração de entidades nomeadas.

O processo de extração de entidades nomeadas desenvolvido consiste na identificação e recuperação de parcelas de texto conhecidos como n-grams a partir de sentenças do texto para então classificar os n-grams em entidades.

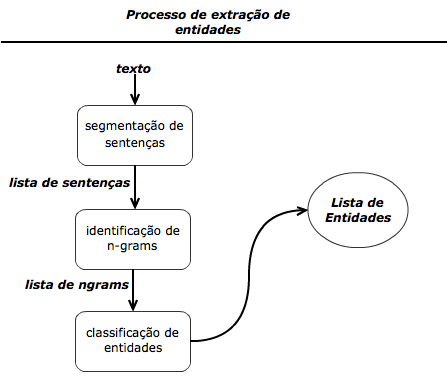


Figure 4

Conforme a figura 4, o módulo de reconhecimento de entidades proposto foi dividido em 3 fases.:

#### A fase de segmentação de sentenças

Nesta fase, o texto recebido como entrada é submetido a uma limpeza como visto no pseudo-código 1 e em seguida é tokenizado em sentenças que indicam fim ou pausa do período. Deste modos os caracteres de pontuação: ponto final, ponto e ponto e virgula foram tratados como delimitadores de sentença. Após a tokenização, as sentenças são armazenadas em um vetor de sentenças e o fluxo segue para a etapa seguinte.

#### A fase de identificação de n-grams

N-gram ou N-grama em português, é o nome dado a um conjunto de palavras em sequência obtidos a partir de um texto. Os n-grams são classificados de acordo com o total de palavras que os compoem assim temos:

Table 3

|  |  |  |
| --- | --- | --- |
| Brasil, Acre | Unigram | Uma palavra |
| São Paulo, Rio Negro | Bigram | Duas palavras |
| Rio de Janeiro | Trigram | Três palavras |
| São José de Itabapoana | Quadrigram | Quatro palavras |

Embora existam n-grams com diversos números de palavras iremos adotar para este trabalho apenas n-grams com até quatro palavras.

Nesta etapa as sentenças são tokenizadas em palavras e os n-grams são então separados de acordo com o número de palavras.

Considerações importantes a respeito da identificação dos n-grams:

1. Durante a identificação de unigrams (n-gram com apenas uma palavra), são removidas as palavras consideradas stopwords. Para tanto utilizamos o mesmo procedimento adotado para a montagem da lista de palavras mais frequentes.
2. Durante a identificação de n-grams, levamos em consideração algumas regras de formação dos n-grams para diminuir a lista de ocorrências possíveis, estas regras estão descritas no apêndice B e levam em consideração o **pós tagger** que será tratado adiante.
3. Durante a identificação dos n-grams identificamos também a posição do n-gram dentro do texto de origem para efeito de eliminação de n-grams contidos em outros n-grams.

Dado o texto: **São José de Itabapoana é uma cidade linda**.

Se separassemos os n-grams sem levar em consideração a posição dos mesmos no texto acima, teríamos a seguinte conclusão:

|  |  |
| --- | --- |
| São, José, Itabapoana, Cidade, Linda | Unigrams |
| São José, Cidade Linda | Bigram |
| José de Itabapoana | Trigram |
| São José de Itabapoana | Quadrigram |

Contúdo, se observarmos a identificação das posições, teremos apenas os n-grams São José de Itabapoana e Cidade Linda selecionados.

Assim, ao final da fase de identificação de n-grams, são identificados n-grams distintos separados em vetores por seus respectivos números de palavras.

#### A fase de classificação de entidades

Para esta fase era necessário um módulo que respondesse se um determinado n-gram podia ou não ser uma entidade. Algumas alternativas de mercado foram observadas mas como abordagem final fomos compelidos a desenvolver um classificador binário capaz de responder com um percentual maior de assertividade.

A solução caseira consistiu na construção de um classificador binário que identifica a probabilidade de um determinado n-gram ser ou não uma entidade nomeada.

Toda a documentação do classificador bem como os detalhes de sua implementação estão melhor detalhados no apêndice A.

Em linhas gerais, o classificador foi desenvolvido para reconhecer entidades baseando-se no padrão de contrução gramatical dos termos que compoêm a entidade. Por exemplo, o n-gram “Rio de Janeiro”, possui a seguinte formação gramatical segundo o pós tagger desenvolvido: NPROP+PREP+N. Desta forma a probabilidade obtida pelo classificador para o n-gram ser uma entidade é de 98%.

Para nosso trabalho, foram tratadas como entidades, propabilidades acima de 80%.

Na tabela 4 temos um comparativo das entidades que foram reconhecidas por estes serviços para um determinado texto. Podemos notar o desempenho do nosso classificador que reconheceu todas as entidades mapeadas.

“*A* ***Polícia Federal*** *informou que foi instaurado um inquérito, após uma denúncia para o* ***DNPM*** *de crime de usurpação de bens públicos decorrentes da exploração ilegal de pedras preciosas.* ***Rogério Castro****, responsável pela agência de recursos naturais do* ***Ibama****, que atuava em* ***Minas Gerais****, foi destituído do cargo.”*

Table 4

|  |  |
| --- | --- |
| Ltasks | Polícia Federal, DNPM, Rogério Castro |
| Yahoo | pedras preciosas, bens, castro |
| Nltk | Castro, Rogério Castro, após uma denúncia para, cargo, responsável pela agência, DNPM, Federal informou que foi instaurado um inquérito, usurpação de bens |
| Zemanta | Brazil, South America, States, People, Business, Lake Chapala, Oaxaca, LinkedIn |
| Nosso Extrator | Policia Federal, Rogerio Castro, Minas Gerais, DNPM, Ibama |

Ao final da fase de classificação de entidades, temos uma lista com as entidades prováveis que foram encontradas no texto.

## Seleção de candidatos

A fase de seleção de candidatos consiste em realizar consultas no servidor de busca interno para identificar um conjunto de matérias que possam ser aproveitadas para a recomendação. Esta consulta precisa levar em consideração alguns critérios que permitam diminuir os ‘ruídos’ durante a fase de recomendação bem como reduzir o espaço de amostra da base de dados. O projeto Google News que pode ser estudado no paper [Google News Personalization: Scalable Online Collaborative Filtering], faz uso de uma fase de seleção de artigos candidatos que utiliza como características para sua seleção informações como: a edição da notícia, o idioma, a recência, seções selecionadas pelo usuário e etc.

Nosso mecânismo de seleção de candidatos é dividido em três etapas definidas como definição de critérios de ordenação e filtragem, combinação de features e recuperação das candidatas. Na primeira etapa, vamos adotar alguns critérios semelhantes ao modelo explicado no projeto do Google News. Na etapa de combinação de features, vamos fazer uso das features extraídas pelo extrator de features combinando-as para melhorar os resultados retornados na seleção das matérias candidatas a recomendação. Na etapa de recuperação, iremos montar consultas escritas na sintaxe lucene para executar consultas sobre nosso servidor de busca em solr.

### Definição dos critérios de filtragem e ordenação

O primeiro critério para recupeção das candidatas se dá pela **data de publicação**. Quanto mais próxima for a publicação da matéria relacionada à data de criação do novo texo, acreditamos que maior será a chance dos textos falarem do mesmo assunto. Este é um critério de ordenação.

O segundo critério para seleção é fornecido pelo próprio editor. A **editoria** da matéria permite-nos filtrar candidatos da mesma editoria aproximando com isso a relação de conteúdo entre o texto e as matérias candidatas. Este é um critério de filtragem.

O último critério e o mais importante é o **score** da matéria relacionada que é mais um critério de ordenação. O **score** é dado de acordo com a função de similaridade existente no solr que faz uso da biblioteca lucene [Lucene API]. O score possui a seguinte fórmula:

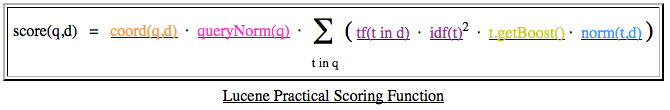


Figure 5

Onde:

* ***tf(t in d)*** ou Term Frequency, define o número de vezes que um termo t aparece no documento d. Isto implica em documentos com o maior número de termos encontrados recebem um maior score.
* ***idf(t)*** ou Inverse Document Frequency. Significa o inverso da frequência nos documentos. Frequência nos documentos é o número de documentos em que um termo t aparece. Ou seja quanto mais raro o termo maior é o valor do idf e melhor é o score.
* ***coord(q,d)*** é um fator baseado no no número de termos da consulta que foram encontrados em um determinado documento. Tipicamente, um documento que possui mais termos da consulta em seu corpo, tende a ter um melhor score.
* ***queryNorm(q)*** é um fator de normalização que visa permitir a comparação de scores entre consultas diferentes.
* ***t.getBoost()*** é o peso dado a um termo t para uma determinada consulta. Ou seja, em tempo de execução de consulta é possível definir pesos para termos específicos da consulta aumentando assim o score de documentos que possuem estes termos.
* ***norm(t,d)*** é o produto de um conjuto de pesos que são atribuídos em tempo de indexação dos documentos.

Em linhas gerais, esse score visa ordenar as matérias de acordo com a similaridade existente com os parêmetros de consulta que são elaborados na etapa de combinação de features.

### Combinação de features

Na etapa de combinação de features para extração de matérias candidatas, as features são combinadas para alcançar o melhor resultado.

Para descobrir a melhor combinação de features, precisavamos definir um baseline. Para este trabalho, utilizamos nosso corpus de matérias já relacionadas manualmente para contabilizar o total de acertos baseado em um critério simples. O critério adotado para seleção de candidados no baseline foi o uso dos termos mais frequentes extratídos da matéria.

Estabelecemos então, com o total de acertos apurados, o baseline para as combinações futuras.

A partir deste baseline, as features foram combinadas sempre objetivando o aumento percentual dos valores obtidos com o baseline. Desse modo, quando combinamos as features extraídas com o extrator de entidades às features utilizadas para montagem do baseline, conseguimos um aumento percentual de aproximadamente 38% no número de acertos. Após este momento, não conseguimos mais aumentar o percentual de acertos.

No capítulo 4 que fala sobre a fase de experimentos, onde entraremos em mais detalhes sobre a combinação das features por hora vamos nos fixar na combinação que apresentou os melhores resultados.

### Recuperação das candidatas

Após a identificação das features a serem utilizadas e dos critérios de ordenação e filttragem, passamos para a etapa de recuperação das candidadas que consistem em realizar a busca destas informações no nosso servidor de busca em solr.

Estas consultas são escritas na sintaxe lucene [lucene sintaxe] e submetidas ao solr atravéz de sua api de consulta. Como exmplo, para a matéria “**Reabertura do Zoológico de Goiânia é adiada para 2012**” que pode ser vista no anexo C, foram estraídas as seguintes features segundo a tabela 5:

Table 5

|  |  |
| --- | --- |
| Termos mais Frequentes | parque, animais, adiada, estao, mes, prefeitura, reabertura, reinauguracao |
| Entidades | Reabertura do Zoologico, Zoologico de Goiania, Recursos Naturais Renovaveis, Ministerio Publico Federal, Cristiane Borges Miguel, Instituto Brasileiro, Zoologico, Goiania, Ibama, MPF, Amma |

Após a tradução das features em sintaxe lucene e o incremento dos filtros das editorias “Goiás” e “Brasil” e da ordenação descrescente por “score” e “data de publicação”, temos a seguinte consulta:

**(((parque) OR (Ibama) OR (animais) OR (prefeitura) OR (Reabertura do Zoologico) OR (Zoologico de Goiania) OR (Recursos AND Naturais AND Renovaveis) OR (Ministerio AND Publico AND Federal) OR (Cristiane AND Borges AND Miguel) OR (Instituto AND Brasileiro) OR (Previsao) OR (Zoologico) OR (Goiania) OR (MPF) OR (Amma)) AND (editoria\_principal\_s:"Goiás" OR editoria\_principal\_s:"Brasil" ) ) isIssued:true type:texto publisher:G1 sort=score desc, issued, rows=50**

Esta consulta retorna o conjunto das 50 primeiras matérias pois limitamos os resultados retornados no parâmetro **rows**.

## Recomendação

Comecando

## Decisões de Projeto

As principais tecnologias descritas abaixo, foram utilizadas para projetar, desenvolver e suportar a ferramenta pois são tecnologias de código aberto, gratuitas, do conhecimento técnico do autor e por se adequarem as práticas utilizadas no contexto da empresa de onde será aplicado o experimento.

***PYTHON*:** Linguagem de programação criada no inicio dos anos 90 por Guido van Rossum. Considerada uma linguagem de alto nível, permite a escrita de código consciso, enxuto e poderoso. Possui uma estrutura de código simples de modo que a maioria dos programadores conseguem fácilmente ler e entender um programa escrito com ela. Python vem com um grande conjunto de bibliotecas nativas que permitem desde o desenvolvimento com funções matemáticas até o uso de parsing de XML. Em seu modo iterativo, é possível criar, executar e avaliar o resultado de funções. Python é uma linguagem que suporta diversos paradigmas de programação desde a orientação a objetos até a programação funcional.

Python é uma linguagem multi-plataforma e tem licensa aberta para uso.

***NLTK***: É uma caixa de ferramentas escrita em python que apresenta um conjunto de bibliotecas e funções para o processamento de linguagem natural e para a análise de textos. Foi originalmente criado com o propósto de ensinar mas com o passar dos anos, tem sido adotado pela industria é por pesquisadores.

***SOLR*:** É uma plataforma de busca corporativa de código aberto oriunda do projeto Apache Lucene. Suas características principais incluem pesquisa de texto, pesquisa facetada, clustering, integração com banco de dados e a capacidade de indexação de documentos ricos (por exemplo, Word, PDF). Solr é altamente escalável, fornecendo pesquisa distribuída e replicação dos índices.

Solr é escrito em Java e funciona como serviço de busca que roda dentro de um container de servlet como o Tomcat. Solr utiliza a biblioteca Lucene de busca em seu núcleo para indexação de textos e a realização de pesquisas.

Solr tem interfaces de consulta em REST/XML e JSON que a tornam fácil de usar a partir de praticamente qualquer linguagem de programação.

***GIT*:** é um sistema de controle de versão de código aberto e gratuito que permite que se trabalhe com diversas versões de arquivos organizados em um diretório e localizados local ou remotamente, mantendo-se suas versões antigas e os logs de quem e quando manipulou os arquivos. É especialmente útil para se controlar versões de um software durante seu desenvolvimento, ou para composição colaborativa de um documento.

***MYSQL*:** é um sistema de gerenciamento de banco de dados (SGBD) que utiliza a linguagem SQL Como interface. Desenvolvido como projeto de código aberto.