|  |
| --- |
| 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. |

|  |
| --- |
| brasao |
| Demetrius Costa Rapello  Sistema de recomendação para suporte a produção de matérias relacionadas no portal G1 |
|  |
| 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. Aprovada pela Comissão Examinadora abaixo assinada. |
| Prof. Marco Antônio Casanova  Orientador e Presidente  Departamento de informática - PUC-Rio  Prof. Alessandro Fabricio Garcia  Departamento de informática - PUC-Rio  Prof. Gustavo Robichez de Carvalho  Departamento de informática - PUC-Rio  Prof.ª. Simone Diniz Junqueira Barbosa  Departamento de informática - PUC-Rio  Rio de Janeiro, 19 de novembro de 2011. |

|  |  |
| --- | --- |
|  | Todos os direitos reservados. É proibida a reprodução total ou parcial do trabalho sem autorização da universidade, do autor e do orientador. |
|  | Demetrius Costa Rapello  Graduou-se em Ciência da Computação pela Universidade de Augusto Motta (UNISUAM) em Dezembro de 2000. Tem experiência na área de Ciência da Computação, com ênfase em Desenvolvimento de Software. Tem trabalhado em analise de sistemas desde 1997. |
|  | Ficha Catalográfica |
|  | Demetrius Costa Rapello  Sistema de recomendação para suporte a produção de matérias relacionadas no portal G1 / Demetrius Costa Rapello; orientador: Marco Antônio Casanova. - Rio de Janeiro: PUC-Rio, Departamento de Informática, 2011.  v., 100 f.: il. ; 29,7 cm  Dissertação de Mestrado - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática  Referencias bibliográficas incluídas.  Sistemas de Recomendação; Recuperação da Informação; Extração de Entidades Nomeadas; |

A Deus, aos meus pais, a minha esposa e aos meus filhos.

|  |
| --- |
| Agradecimentos |
| A Deus pelo apoio incondicional, conforto de coração e paz de espírito que tanto foram importantes para alcançar este objetivo.  Aos meus pais, Tarciso Rapello e Maria da Conceição costa Rapello, pelo apoio, educação, carinho e dedicação.  A minha esposa Gabriela Barbosa da silva Rapello, pelo apoio constante, carinho e compreensão.  Aos meus filhos Gabriel e Matheus que entenderam a ausência do pais com carinho e compreensão.  Ao meu orientador, prof. Marco Antônio Casanova, por sua dedicação, motivação, ensinamentos e orientação.  À Globo.com, pelo financiamento e auxílios concedidos, sem os quais este trabalho não poderia ter sido realizado.  Aos professores da Comissão examinadora.  A todos os amigos e familiares que de alguma forma contribuíram para a realização deste trabalho. |

|  |
| --- |
| Resumo |
| Demetrius Costa Rapello. **Sistema de recomendação para suporte a produção de artigos do portal G1.** Rio de Janeiro, 2011. 100p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.  bla |
| Palavras-chave  Sistemas de Recomendação; Recuperação da Informação; Extração de Entidades Nomeadas. |

|  |
| --- |
| Abstract |
| Demetrius Costa Rapello. **Recomended System to support News article production on G1.** Rio de Janeiro, 2011. 101p. Dissertação de Mestrado - Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.  BLa. |
| Keywords  Recommended Systems; Information Retriaval; Named Entity Extraction. |

1. Introdução 14

1.1. Objetivo da Dissertação 15

1.2. Organização da Dissertação 15

2. Trabalhos Relacionados 16

2.1. Sistemas de recomendação 16

2.1.1. Modelos de sistemas de recomendação 17

2.1.2. Baseados em conteúdo 17

2.1.3. Baseado em filtragem colaborativa 17

2.1.4. Baseado em nichos demográficos 17

2.1.5. Baseado em conhecimento 18

2.1.6. Baseado na comunidade 18

2.1.7. Sistemas híbridos de recomendação 18

2.2. Projeto Pure 18

2.2.1. Interface para registro de artigos do usuário 19

2.2.2. Treinamento do modelo probabilístico 20

2.2.3. Seleção de palavras e atribuição de peso 20

2.2.4. Etapa de Geração do modelo probabilístico 20

2.2.5. Recuperação diária de novos artigos da base PubMed. 21

2.2.6. Recomendação dos artigos 21

3. GRNews – Sistema de recomendação de matérias de segundo nível. 22

3.1. Arquitetura do sistema e sumário 22

3.2. Extrator de Features 25

3.2.1. Termos mais frequentes 25

3.2.2. Tags HTML informativas 26

3.2.3. Texto em títulos de vídeos e fotos 27

3.2.4. Entidades Nomeadas 27

3.3. Seleção de candidatos 34

3.3.1. Definição dos critérios de filtragem e ordenação 35

3.3.2. Combinação de features 36

3.3.3. Recuperação das candidatas 37

3.4. Recomendação 38

3.4.1. Conceito de “está relacionada” 38

3.4.2. Performance da comparação 39

3.5. Decisões de Projeto 39

4. Experimento 41

5. Conclusões e trabalhos futuros 42

6. Bibliografia 43

7. APÊNDICE A 46

|  |
| --- |
| Lista de imagens |
|  |

Figure 1 - Função de Recomendação 21

Figure 2 – Função do Zcore 21

Figure 3 – GRNews arquitetura 22

Figure 4 – Interface Web de Matéria 23

Figure 5 – Componente de Matérias Recomendadas 24

Figure 6 – Exemplo de matéria 27

Figure 7 – Processo de extração de entidades 31

Figure 8 – Função de Score do Lucene 35

|  |
| --- |
| Lista de tabelas |

Table 1 – Campos do Formulário de Matéria 23

Table 2 – Comparativo de extratores 30

Table 3 – Formatos de n-grams 31

Table 4 – Lista de N-grams 32

Table 5 – Comparativo de Entidades 34

Table 6 – Exemplo de combinação de features 37

Lista de pseudocódigos

pseudo-código 1 – Função de limpeza 25

pseudo-código 2 – Yahoo term extractor 28

pseudo-código 3 – Nltk entities extractor 29

pseudo-código 4 – Ltasks entities extractor 29

pseudo-código 5 – Zemanta concepts extractor 30

# Introdução

Com o aumento considerável de informações de notícias na web, a recomendação tem se tornado extremante importante para os usuários. Estudos indicam que o volume de páginas indexadas nos principais buscadores aproximasse da casa dos 20 bilhões [2,3]. Desta forma podemos perceber que os usuários não tem como absorver todo esse conteúdo produzido sem que um agente organize em contextos específicos. O sistemas de recomendação são os agentes responsáveis por organizar estas informações baseados em lógicas previamente definidas. Quando estamos navegando na web em um site de compra de jogos e nos deparamos com uma lista dos dez jogos mais procurados pelos usuários ou quando estamos numa locadora virtual e verificamos uma lista com os filmes mais vistos, estamos na verdade recebendo recomendações para facilitar nossas decisões.

Sistemas de recomendação tem sido largamente utilizados para atender a esta necessidade, pois em geral, estes sistemas são construídos com o intuito de predizer as necessidades do usuário de modo que se possa oferecer conteúdos com uma maior probabilidade de aproveitamento.

Para aprender a predizer as necessidades dos usuários, os sistemas de recomendação necessitam de acesso ao feedback dos usuários quando estes estão consumindo um determinado conteúdo. Este feedback pode ser dado de forma explicita quando um usuário informa ao sistema quais são as suas preferencias, ou de forma implícita quando o sistema descobre as preferencias do usuário baseado em suas interações.

## Objetivo da Dissertação

Diante deste cenário, propomos um sistema de recomendação que aprenda as necessidades dos usuários baseado no fluxo de navegação entre as matérias para sugerir conteúdos relacionados aos novos artigos produzidos.

Com a necessidade de recomendação de conteúdo relacionado, os editores do portal Techtudo passaram a ter uma segunda função. Atualmente, além da criação da nota ou artigo sobre um determinado tema, eles ainda precisam relacionar outras notas e ou artigos no sistema de recomendação existente. Deste modo o processo de produção de conteúdo se torna lento e a totalidade do tempo do editor, que deveria ser para confecção mais elaborada da matéria, passar a ser destinado a um trabalho manual de mineração de texto para realização de relacionamentos de conteúdos.

Para gerar recomendações de conteúdos para uma matéria, o editor precisa utilizar um fluxo de trabalho que passa pelas seguintes etapas:

1. Identificação de palavras/frases chaves relevantes ao tema.
2. Consulta de conteúdos anteriores na base de conhecimento utilizando as chaves.
3. Reconhecimento da similaridade entre os conteúdos e o tema.
4. Relacionamento manual dos conteúdos.

A expectativa é:

* completar.

## Organização da Dissertação

O restante deste documento está organizado em cinco capítulos, da seguinte forma:

O **Capítulo 2** apresenta o estado da arte em sistemas de recomendação e as ferramentas e *frameworks* existentes comparados a ferramenta proposta.

O **Capítulo 3** descreve a ferramenta desenvolvida para apoiar a recomendação de conteúdo na produção de matérias do G1.

O **Capítulo** 4 apresenta os resultados obtidos com a experimentação da ferramenta contra um corpus real.

O **Capítulo 5** apresenta as conclusões e trabalhos futuros.

# Trabalhos Relacionados

Existem três conceitos importantes que serão abordados durante essa dissertação: sistemas de recomendação, técnicas de recuperação da informação e a técnica de extração de entidades nomeadas. Foi realizada uma pesquisa na literatura para cada um desses tópicos. Na última seção será apresentada a diferença do que já foi realizado para o trabalho que está sendo desenvolvido.

## Sistemas de recomendação

Sistemas de recomendação são softwares especializados em apresentar “opções” para serem usadas por seus usuários [5,6,18]. Esses sistemas auxiliam pessoas que não possuem muita experiência ou competência para pesquisar dados acerca de um determinado assunto.

Em linhas gerais, os sistemas de recomendação procuram oferecer as melhores opções de resposta para as necessidades dos usuários num processo de tomada de decisão.

Os sistemas de recomendação podem oferecer sugestões em diferentes domínios como por exemplo: que item comprar em uma loja virtual, que artigo ler em um site de notícias ou mesmo que restaurante visitar em uma cidade.

No sistema de recomendação do **IMDB** (**Internet Movie Database**), quando o usuário seleciona um filme do catálogo para ler a respeito, o sistema apresenta também uma lista de sugestões de filmes relacionados para o usuário.

Assim como o **IMDB**, o site de compras **Amazon** apresenta para cada produto selecionado pelo usuário, uma lista de recomendações de outros produtos que possam servir para o usuário dentro de uma mesma compra.

Sistemas de recomendação são usualmente personalizados com base em características individuais ou coletivas, todavia, os sistemas de recomendação não personalizados também tem o seu espaço. Sistemas de recomendação não personalizados são mais simples de serem implementados e geralmente são usados para a recomendações mais gerais como por exemplo, a lista dos “10 mais” de um determinado assunto ou tema.

Sistemas de recomendação personalizados tentam antecipar as necessidades do usuário analisando informações do seu perfil e levando em consideração as restrições de domínio para então recomendar sugestões. Ainda no site de compras da **Amazon**, uma vez realizada uma compra, as informações são armazenadas no perfil do usuário de modo que para as compras futuras o usuário receba como recomendação, não apenas itens relacionados pela categoria em que se encaixam, mas também por que estão relacionados com a última compra feita por ele.

### **Modelos de sistemas de recomendação**

Sistemas de recomendação variam de acordo com a técnica utilizada para descobrir as preferencias do usuário. Entre os sistemas de recomendação mais comuns temos:

### **Baseados em conteúdo**

Sistemas de recomendação baseados em conteúdo tentam recomendar opções que são similares a algum item que o usuário já selecionou no passado usando como features informações extraídas do produto/conteúdo que o usuário consultou. Por exemplo, se um usuário leu um artigo sobre política em um portal de notícias, este portal poderia sugerir em consultas futuras artigos relacionados a política.

Segundo [1] os sistemas de recomendação baseados em conteúdo junto com os baseados em filtragem colaborativa, são os mais difundidos e utilizados entre os sistemas de recomendação.

### **Baseado em filtragem colaborativa**

Sistemas de recomendação baseados em filtragem colaborativa, tentam recomendar ao usuário opções que foram utilizadas por usuários com os mesmos interesses. Por exemplo, se um grupo de usuários selecionam o produto X e em seguida selecionam o produto Y, o sistema de recomendação entende que usuários que acessam o produto X também acessam o produto Y e passa a recomendar o produto Y toda vez que algum novo usuário se interessar pelo produto X.

### **Baseado em nichos demográficos**

Este tipo de sistema de recomendação baseia-se no perfil demográfico do usuário para recomendar opções. Usuários de um determinado estado recebem recomendações diferentes de usuários de outros estados. Os nichos demográficos podem ser: idade, sexo, língua etc.

### **Baseado em conhecimento**

Sistemas de recomendação baseados em conhecimento tentam recomendar ao usuário opções baseando-se no conhecimento específico do domínio do sistema. Este modelo também é reconhecido como um modelo baseado em caso onde o problema é a analise das necessidades do usuário e a solução é o conjunto de opções a serem recomendadas.

Sistemas de recomendação com esta técnica tendem a trabalhar melhor que os outros no início, porém se não são acompanhados de componente de aprendizagem se tornam ineficientes.

Neste sistema o principio de recomendação está baseado na similaridade de uma solução para um dado problema.

### **Baseado na comunidade**

Este tipo de sistema de recomendação procura basear-se nas relações que o usuário possui com sua rede de amigos para realizar as recomendações. Neste modelo acredita-se que a recomendação de pessoas ligadas ao usuário tendem a ser mais efetivas. Esta abordagem tem se tornado bastante atrativa tendo em vista o grande crescimento das redes de relacionamento.

### **Sistemas híbridos de recomendação**

Sistemas de recomendação híbridos procuram utilizar técnicas de outros sistemas de recomendação de forma a complementar as deficiências entre as técnicas.

## Projeto Pure

O projeto PURE [15] é um sistema para recomendação de artigos médicos que utiliza o princípio de recomendação baseado na filtragem de conteúdo. Em linhas gerais, o sistema executa uma classificação dos artigos preferidos do usuário para então recomendar outros artigos para leitura.

O sistema PURE pode ser entendido pelo fluxo de operações a seguir:

1. O usuário acessa o sistema para informar os artigos do seu interesse. Estes artigos são armazenados na base de dados do PURE.
2. Um sistema de aprendizado de máquina é aplicado para extrair as preferências do usuário com base nos seus artigos de interesse.
3. O sistema PURE consulta a base de dados PubMed para baixar os novos artigos publicados.
4. Os artigos baixados da PubMed são ordenados com base no modelo treinado com as preferências do usuário.
5. Os artigos são então apresentados para o usuário.

O propósito do PURE desenvolvido por TAKASHI YONEYA e HIROSHI MAMITSUKA, é oferecer uma interface que facilite o processo de busca de artigos relevantes pela comunidade científica na base de dados PubMed. A base de dados PubMed mantém um grande acervo de artigos de biologia e medicina com um volume diário de atualização da ordem de centenas artigos. Diante deste cenário, os autores perceberam que um sistema de recomendação baseado em filtragem de conteúdo poderia auxiliar a comunidade científica no trabalho de consulta de novos artigos. Os módulos a seguir fazem parte do sistema PURE.

### **Interface para registro de artigos do usuário**

Para utilizar o sistema, o usuário precisa registrar os seus artigos de interesse na base de dados do PURE. Para esta atividade o usuário acessa a interface web do sistema e seleciona os artigos de sua preferência em uma listagem. Os artigos selecionados são então gravados no perfil do usuário e armazenados no banco de dados do PURE.

O usuário tem a permissão de adicionar novos arquivos e alterar sua lista de interesse.

### Treinamento do modelo probabilístico

Os artigos de interesse do usuário são utilizados para a concepção de um modelo probabilístico que procura identificar as preferências do usuário para novos artigos. Esse modelo é dividido em duas etapas:

### **Seleção de palavras e atribuição de peso**

Nesta etapa o sistema trata os artigos do PubMed como um vetor de palavras ordenadas por peso. Estas palavras são obtidas a partir da eliminação de palavras irrelevantes para o sistema classificadas como stopwords.

As stopwords são obtidas por duas estratégias distintas, a primeira consiste na geração do DF e TF-IDF das palavras oriundas de uma porção aleatória de artigos da base de dados do PubMed. As palavras com alto DF (número de documentos onde a palavra aparece) ou com baixo TF-IDF (métrica que informa quanto a palavra é relevante para o documento) são consideradas stopwords. A segunda estratégia consiste em considerar como stopwords as palavras que respeitam as seguintes regras: 1) palavras com menos de 3 letras; 2) palavras sem caracteres alfabéticos; 3) palavras que aparecem no **Journal of Business Research** de Jan. 2005 a 2006.

Após a eliminação das stopwords é dado um peso para cada palavra restante do documento. Este peso é obtido pela verificação da distribuição da palavra pelo documento (TF).

### **Etapa de Geração do modelo probabilístico**

As palavras selecionadas na etapa anterior são usadas para treinar um classificador probabilístico que será usado para gerar uma métrica de recomendação para os novos artigos. A função utilizada para computar o grau de recomendação de um artigo é dada pela fórmula abaixo:

Sendo d um artigo, z a variável correspondente ao cluster, s um campo existente na estrutura do artigo exe., título e w uma palavra do artigo temos:

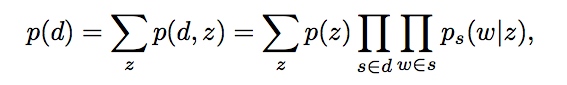


Figure 1 - Função de Recomendação

Em seguida os autores treinam os parâmetros de probabilidade p(z) e ps(w|z) a partir dos artigos preferidos do usuário utilizando o algoritmo de Maximização de Expectativa (EM).

### **Recuperação diária de novos artigos da base PubMed.**

O sistema PURE diariamente executa uma operação de recuperação dos novos artigos publicados na base de dados do PubMed. Os novos artigos são armazenados na base de dados do sistema para serem classificados de acordo com as preferências de cada usuário.

### **Recomendação dos artigos**

Para cada artigo recuperado, são extraídas palavras que são usadas como base para geração do critério de ordenação que é dado de acordo com a função de recomendação definida anteriormente. Como forma de ajustar possíveis desvios do algoritmo de recomendação, os autores apresentam um score adicional para cada artigo. O Z-score é obtido através do agrupamento dos artigos em conjuntos de artigos com o mesmo número de palavras. Deste modo é separado o mínimo e o máximo graus de recomendação do grupo onde o Z-score do artigo é dado pela formula:



Figure 2 – Função do Zcore

Os artigos com o maior Z-score são então recomendados para o usuário.

## Query by Document

Neste trabalho os autores defendem uma técnica de recuperação de conteúdo relacionado utilizando informações existem no texto para consulta na base de dados.

A primeira idéia dos autores consiste em é extrair do texto frases substantivas que possam ser relevantes para pesquisar por conteúdo relacionado. Em seguida eles defendem que estas frases podem ser substituídas ou melhoras através do uso de fontes externas, no caso, o Wikipédia.

Para melhorar as frases com o uso do Wikipédia, eles utilizam um grafo de conceitos do Wikipédia onde os nós também representam frases substantivas de modo que eles podem recuperar as frase presentes no texto e em seguida melhorá-los percorrendo o grafo.

Para extrair as frases substantivas, eles se utilizando de um POS-tagger onde eles consideram todas as frases cujo padrão de formação respeite o conceito de frase substantiva definido no trabalho.

A partir deste ponto as frases são ordenadas de acordo com um score que é dado por dois mecanismos distintos. O primeiro mecanismo utiliza o TF/IDF dos termos na frase para atribuir um score, enquanto que o segundo computa o score baseado nas informações mutuas dos termos da frase.

Para validar o experimento os autores utilizaram o serviço Mechanical Turk da Amazon para avaliar a qualidade da extração das frases. Basicamente, eles informavam um texto e um conjunto de frases para os usuários para que eles assinalassem se aquelas frases eram relevantes para encontrar aquele texto em um serviço de busca como o Google.

# 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 data de publicação decrescente 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 componentes 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 3.

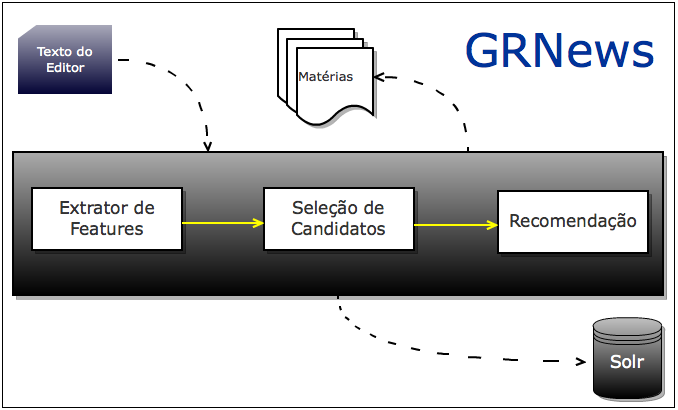


Figure 3 – GRNews arquitetura

Para ser possível acionar o motor do GRNews, é necessário que o editor informe os parâmetros de entrada e para isso, o ponto de partida é o formulário de matéria. O formulário de matéria, figura4, 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.

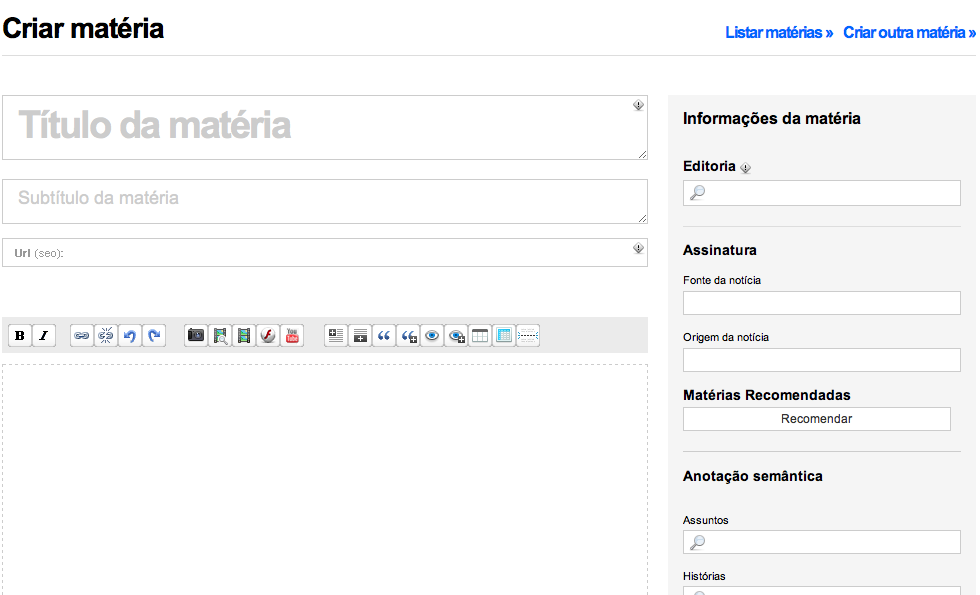


Figure 4 – Interface Web de Matéria

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

Table 1 – Campos do Formulário de Matéria

|  |  |  |
| --- | --- | --- |
| 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 5. 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.

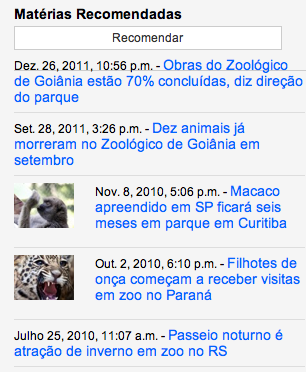


Figure 5 – Componente de Matérias Recomendadas

## 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 frequência de aparição no texto. Segundo [irbook] a frequência 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 pseudocódigo 1, consiste na remoção do ruído presente no texto, são removidos os caracteres de pontuação bem como os marcadores html. Após esta etapa estamos aptos a realizar a tokenização.



pseudocódigo 1 – Função de limpeza

Segundo [12] **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 6 abaixo, serão selecionados os termos “Biblioteca Pública Dolor Barreira” e “Festival de Cinema e Cultura da Diversidade Sexual.”



Figure 6 – Exemplo de matéria

### Texto em títulos de vídeos e fotos

Os termos que ocorrem dentro de estruturas que apresenta vídeos ou imagens 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 6 o texto “**Quinta edição do festival For Rainbow começa nesta quinta-feira (27) (Foto: Divulgação**)” será selecionado.

### Entidades Nomeadas

Segundo [14] 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 [18], 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.

****

pseudocódigo 2 – Yahoo term extractor

**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.

pseudocódigo 3 – Nltk entities extractor

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 língua portuguesa. Para utilização on-line é necessária a utilização de uma chave de acesso que é obtida através de um cadastro no site.



pseudocódigo 4 – Ltasks entities extractor

Zemanta

É uma ferramenta concebida para geração de conteúdo relacionado para blogs, contudo, 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.



pseudocódigo 5 – Zemanta concepts extractor

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

Table 2 – Comparativo de extratores

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 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 algoritmo, a limitação de acessos ao serviço e o tempo gasto em cada requisição foram determinantes para que partíssemos 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.

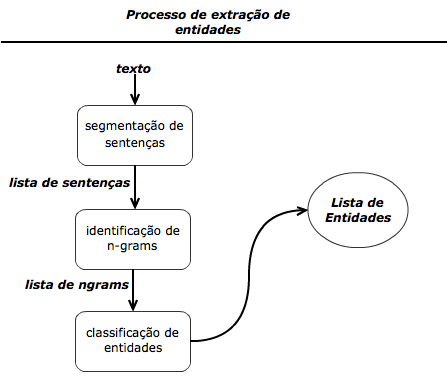


Figure 7 – Processo de extração de entidades

Conforme a figura 7, 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 pseudocódigo 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 compõem assim temos:

Table 3 – Formatos de n-grams

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

Table 4 – Lista de N-grams

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

Contudo, 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 construção gramatical dos termos que compõem 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, probabilidades 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 5 – Comparativo de Entidades

|  |  |
| --- | --- |
| 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 [16], 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 data de publicação, seções selecionadas pelo usuário e etc.

Nosso mecanismo 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 reocupação das candidatas se dá pela **data de publicação**. Quanto mais próxima for a publicação da matéria relacionada à data de criação do novo texto, 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 [22]. O score possui a seguinte fórmula:

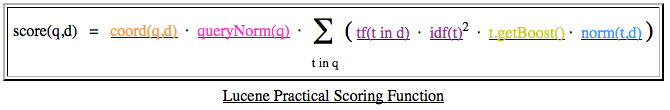


Figure 8 – Função de Score do Lucene

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 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 conjunto 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 parâmetros 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, precisávamos 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 candidatos no baseline foi o uso dos termos mais frequentes extraídos 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 filtragem, passamos para a etapa de recuperação das candidatas que consistem em realizar a busca destas informações no nosso servidor de busca em solr.

Estas consultas são escritas na sintaxe lucene [22] e submetidas ao solr através de sua api de consulta. Como exemplo, para a matéria “**Reabertura do Zoológico de Goiânia é adiada para 2012**” que pode ser vista no anexo C, foram extraídas as seguintes features segundo a tabela 5:

Table 6 – Exemplo de combinação de features

|  |  |
| --- | --- |
| Termos mais Frequentes | parque, animais, adiada, estão, mês, prefeitura, reabertura, reinauguração |
| Entidades | Reabertura do Zoológico, Zoológico de Goiânia, Recursos Naturais Renováveis, Ministério Publico Federal, Cristiane Borges Miguel, Instituto Brasileiro, Zoológico, Goiânia, Ibama, MPF, Amma |

Após a tradução das features em sintaxe lucene e o incremento dos filtros das editorias “Goiás” e “Brasil” e da ordenação decrescente 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

A fase de recomendação consisti em sugerir ao editor as matéria que apresentam a melhor similaridade com a matéria que está sendo criada. Para isso, após a seleção de matérias candidatas, aplicamos a medida de similaridade vector space model (VSM). A medida de similaridade obtida é incorporada ao score dado a matéria relacionada e a listagem de candidatas é então reordenada.

Após a reordenação da listagem, são selecionadas as 5 matérias mais bem avaliadas segundo seus scores. E com isso o sistema GRNews encerra seu ciclo de recomendação.

O Processo de auferir a similaridade entre as matérias candidatas e a matéria que esta sendo construídas possui algumas deficiências como veremos a seguir:

### Conceito de “está relacionada”

Segundo [1], a medida de similaridade sozinha não é suficiente para capturar o relacionamento entre duas matérias. Um exemplo dado por eles é o caso de matérias com conteúdos duplicados onde a medida de similaridade é alta porém o grau de relacionamento não existe por se tratarem do mesmo assunto. Ainda em [1], os autores definem em seu trabalho 3 critérios afim de modelar uma função capaz de entregar uma melhor relação entre as matérias na fase de recomendação, são eles: Relevância e novidade, Clareza de conexão e Suavidade de transição. Ainda neste tema, [9] entendem que o fator de relacionamento pode ser explicado por 3 idéias: Inclusão, quando uma matéria está inserida em outra. Intercessão, quando duas matérias apresentam alguns conceitos em comum. E opostas quando não apresentam conceitos em comum. Além destes trabalhos, é possível encontrar outros que tratam do tema de relacionamento de conteúdo mostrando-nos a dificuldade que o tema implica.

### Performance da comparação

Outra questão importante é a performance do sistema levando se em consideração o volume de comparações que precisam ser executadas para assegurar uma listagem final de matéria recomendadas. Durante a fase de seleção de candidatas, selecionamos 50 matérias. Desta forma para cada recomendação no sistema GRNews, estamos realizando 50 comparações de similaridade usando a função de distância de cossenos (VSM). Não foi observado durante este trabalho uma possibilidade de melhoria no algoritmo ou na abordagem utilizada.

## 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 conciso, enxuto e poderoso. Possui uma estrutura de código simples de modo que a maioria dos programadores conseguem facilmente 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 propósito de ensinar mas com o passar dos anos, tem sido adotado pela indústria 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.

# Experimento

Para avaliar o GRNews e testar a acurácia do processo de recomendação de matérias relacionadas, desenvolvemos um conjunto de experimentos que certificam os módulos do sistema sobre um corpus de matérias já relacionadas manualmente.

Primeiro vamos detalhar como esse dataset foi obtido, depois vamos definir uma herística para avaliação e em seguida vamos executar uma função especialmente desenvolvida para combinar as diversas features obtidas no capítulo 3 de modo a descobrir que combinação possui a maior assertividade sobre o corpus.

Ao final vamos apresentar os resultados obtidos e vamos discutir as possibilidade de melhoria sobre o sistema GRNews.

## O corpus

O corpus utilizado pertence ao portal de noticias da Globo.com o G1 e possui um total de 492 mil matérias no período de fevereiro de 2010 a outubro de 2011 divididas em 162 editorias.

Deste corpus, 72 mil matérias possuem o componente de matéria relacionada. Este componente matém os links das matérias que foram relacionadas pelo editor.

Para diminuir o tempo de execução dos algorítmos reduzimos o corpus para um total 1440 matérias já relacionadas o que representa a 2% do total de matérias relacionadas manualmente. Também reduzimos as representação das editorias ficando apenas com as 15 principais editorias de acordo com o seu número de matérias publicadas.

Decidimos distribuír de forma igualitária as matérias em suas editoria de modo a promover uma visão das editorias com maior percentual de acertos, assim temos a seguinte distribuição:

Table 7 - Representação do Corpus

|  |  |  |
| --- | --- | --- |
| 1 | Brasil | 96 |
| 2 | São Paulo | 96 |
| 3 | Rio de Janeiro | 96 |
| 4 | Minas Gerais | 96 |
| 5 | Economia | 96 |
| 6 | Política | 96 |
| 7 | Mundo | 96 |
| 8 | Espírito Santo | 96 |
| 9 | Pop & Arte | 96 |
| 10 | Auto Esporte | 96 |
| 11 | Concursos e Emprego | 96 |
| 12 | Ciência e Saúde | 96 |
| 13 | Música | 96 |
| 14 | Mercados | 96 |
| 15 | Tecnologia e Games | 96 |
|  | TOTAL | 1440 |

Foram necessárias etapas de limpeza do corpus para garantir que todos os links de matérias apontavam para matérias existentes na nossa base de dados. Isso se fez necessário porque o cadastro de materias relacionadas é aberto e permite a ligação à matérias externas. Assim sendo, matérias que possuiam pelo menos um link externo, foram descartadas durante a preparação do corpus.

Ao final desta etapa de limpeza, temos então um corpus de matérias relacionadas manualmente com links válidos e que corresponde a 2% do corpus total de materias relacionadas manualmente.

Durante a etapa de limpeza, as matérias são indexadas no servidor de busca Solr que dá suporte ao sistema GRNews.

## Critério para avaliação do sistema

Para avaliar o sistema foi necessária a construção de um módulo de combinação das features extraídas pelo extrator de features como dito no capítulo anterior.

Para cada combinação de features os sistema percorre o corpus separado com matérias previamente relacionadas e realiza para cada matéria uma recomendação de n matérias onde n é o número de matérias que serão recomendadas. Após a recomendação o sistema compara as matérias que foram relacionadas manualmente com as que foram recomendadas pelo GRNews e em seguida, extrai a interseção. Se a interscão não for nula, consideramos que a recomendação foi válida.



pseudo-código 6 – Módulo para combinação e validação de Features

## Resultados obtidos

A primeira tentativa foi descobrir a melhor combinação de critério sobre todo o corpus de teste utilizando como base um total de 5 recomendações e ainda dispensando o fator de similaridade das matérias uma vez que ele não é determinante para encontrarmos a melhor relação, com isso obtivemos a tabela abaixo:

Table 8 – Percentual de acerto por Features

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| uthce | 0.447533009 |  | ue | 0.431549687 |  | tc | 0.403752606 |
| ubthce | 0.447533009 |  | bhe | 0.430159833 |  | he | 0.401667825 |
| ubhce | 0.447533009 |  | be | 0.429464906 |  | uc | 0.401667825 |
| ubce | 0.446838082 |  | bthe | 0.429464906 |  | e | 0.395413482 |
| utce | 0.445448228 |  | bte | 0.428769979 |  | ubh | 0.390548992 |
| ubtce | 0.445448228 |  | te | 0.428769979 |  | ub | 0.390548992 |
| bthce | 0.444753301 |  | the | 0.428769979 |  | ut | 0.389159138 |
| uce | 0.444058374 |  | hce | 0.41695622 |  | uth | 0.389159138 |
| thce | 0.444058374 |  | ubhc | 0.412786657 |  | ubt | 0.388464211 |
| bhce | 0.444058374 |  | ce | 0.41209173 |  | ubth | 0.388464211 |
| uhce | 0.443363447 |  | ubthc | 0.411396803 |  | uh | 0.38637943 |
| bce | 0.443363447 |  | uthc | 0.410701876 |  | th | 0.384294649 |
| tce | 0.44266852 |  | ubc | 0.408617095 |  | bh | 0.384294649 |
| btce | 0.44266852 |  | ubtc | 0.407227241 |  | u | 0.384294649 |
| uthe | 0.435024322 |  | utc | 0.406532314 |  | bth | 0.383599722 |
| ubhe | 0.435024322 |  | uhc | 0.405837387 |  | t | 0.383599722 |
| ubthe | 0.434329395 |  | bhc | 0.40514246 |  | b | 0.382904795 |
| ube | 0.433634468 |  | bthc | 0.40514246 |  | bt | 0.382209868 |
| ute | 0.433634468 |  | bc | 0.404447533 |  | hc | 0.212647672 |
| uhe | 0.432939541 |  | thc | 0.404447533 |  | c | 0.17790132 |
| ubte | 0.432939541 |  | btc | 0.403752606 |  | h | 0.061153579 |

|  |  |
| --- | --- |
| **LEGENDA:** | |
| u | Unigrams |
| b | Bigrams |
| t | Trigrams |
| h | Html Tags |
| c | Captions |
| e | Entiddade |

A segunda tentativa, foi em razão da dúvida sobre o fator de similaridade, ou seja, apenas habilitamos o fator de similaridade para ajustar o score das matérias com maior similaridade textual. Com isso pudemos observar quanto o fator de similaridade entre as matérias influencia no nosso algoritimo. O novo score é dado pela multipicação simple entre os score anterior e o vsm entre as duas matérias.

Até este ponto estávamos realizando apenas 5 recomendações por matéria. Executamos então os dois procedimentos anteriores porém alterando o número de recomendações para 10.

Um útimo teste realizado foi a observação do nosso modelo de recomendação dentro de uma editoria específica, ou seja, estamos aqui querendo verificar que feature é mais relevante dentro de uma determinada categoria. Esta é apena uma hipótese diferente que vamos avaliar também variando número de recomendações e a utilização ou não do fator de similaridade.

Algumas conclusões tiradas a partir dos dados levantados:

Execução de combinacoes com 5 recomendacoes

Execucao de combinacoes com 5 recomendacoes com similaridade

Execução de combinacoes com 10 recomendacoes

Execucao de combinacoes com 10 recomendacoes com similaridade

Execução de combinacoes com 5 recomendacoes por editoria

Execucao de combinacoes com 5 recomendacoes com similaridade por editoria

Execução de combinacoes com 10 recomendacoes por editoria

Execucao de combinacoes com 10 recomendacoes com similaridade por editoria

# Conclusões e trabalhos futuros

ESCREVER

# Bibliografia

1. [Yuanhua Lv, Taesup Moon, Pranam Kolari, Zhaohui Zheng, Xuanhui Wang, Yi Chang – 2011] **Learning to model relatedness for news recommendation**, Yahoo! Labs, Sunnyvale, CA, USA.
2. [World Wide Web Size – 2011] **Daily Estimated Size**, disponivel em: <http://www.worldwidewebsize.com/>. Acesso em 08 Jul 2011.
3. [Miniwatts Marketing Group – 2011] **Internet World Stats**, disponivel em: <http://www.internetworldstats.com/stats.htm>. Acesso em 19 Jan 2011.
4. [James Davidson, Benjamin Liebald, Junning Liu – 2010] **The YouTube Video Recommendation System**, Google Inc.
5. [Francesco Ricci, Lior Rokach, Bracha Shapira - 2010] **Introduction to Recommender Systems Handbook**.
6. [Bracha Shapira, Lior Rokach – 2010] **Recommender Systems and Search Engines – Two sides of the same Coin!?**, Department of Information Systems Engineering, Ben-Gurion University.
7. [Jacob Perkins – 2010]. **Python Text Processing with NLTK 2.0 Cookbook**.
8. [Bruno Magalhães Nogueira – 2009] **Avaliação de métodos não-supervisionados de seleção de atributos para Mineração de Textos**, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, ICMC-USP.
9. [Fekade Getahun, Joe Tekli, Chbeir Richard, Marco Viviani, Kokou Yetongnon – 2009] **Relating RSS News/Items**, Laboratoire Electronique, Informatique et Image (LE2I), UMR-CNRS Université de BourgogneSciences et Techniques
10. [Yin Yang, Panagiotis Ipeirotis, Wisam Dakka, Dimitris Papadias, Nilesh Bansal, Nick Koudas – 2009] **Query by Document,** Computer Science University of Toronto, koudas@cs.toronto.edu.
11. [Arni Darliani Asy’arie, Adi Wahyu Pribadi – 2009] **Automatic News Articles Classification in Indonesian Language by Using Naive Bayes Classifier Method**, Universitas Islam Negeri Syarif Hidayatullah Jakarta, Universitas Pancasila Jl. Srengseng Sawah, Jagakarsa Jakarta.
12. [Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze – 2008] **Introduction to Information Retrieval**, Cambridge University Press.
13. [Luiz Claudio Gomes Maia – 2008] **Uso de Sintagmas Nominais na Classificação Automática de Documentos Eletrônicos**, Universidade Federal de Minas Gerais, UFMG.
14. [Alexander E. Richman, Department of Defense – 2008] **Mining Wiki Resources for Multilingual Named Entity Recognition,** Departamento de Defesa Americano.
15. [Takashi Yoneya, Hiroshi Mamitsuka – 2007] **PURE: A Pubmed Article Rcommendation System Based on Content-Based Filtering,** Bioinformatics Center, Kyoto University, Gokasho Uji, Japan and Discovery Research Laboratories, Kirin Pharma, Miyahara, Takasaki, Japan
16. [Abhinandan Das, Mayur Datar Google – 2007] **Google News Personalization: Scalable Online Collaborative Filtering**, Google Inc
17. [Dipanjan Das, Andre F.T. Martins – 2007] **A Survey on Automatic Text Summarization,** Language Technologies Institute Carnegie Mellon University.
18. [Hiroyuki Toda, Patrick Schone – 2005] **A Clustering Method for News Articles Retrieval System**, NTT Cyber Solutions Laboratories, NTT Corporation
19. [Greg Linden, Brent Smith, Jeremy York – 2003] **Amazon.com Recommendations Item-to-Item Collaborative Filtering**, IEEE Internet Computing Magazine.
20. [Ying Huang – 2001] **An Intelligent adaptative news filtering system**, University of Science & Technology of China.
21. [Chin-Yew Lin, Eduard Hovy – 1997] **Identifying Topics by Position,** Information Sciences Institute of the University of Southern California
22. [Apache Lucene – 2011] **Lucene Documentation**, disponivel em: <http://wiki.apache.org/lucene-java> . Acesso em 20 Set 2011.
23. [Min-Ling Zhang, Jose M. Penã and Victor Robles – 2009] **Feature Selection for Multi-Label Naive Bayes Classification**, College of Computer and Infor- mation Engineering, Hohai University, Nanjing 210098, China.

# APÊNDICE A

## Classificador de Entidades Nomeadas

Durante a etapa de levantamento de features, observamos a necessidade de extraír do texto informações que representassem entidades nomeadas, a hipótese levandada era que essas informações poderiam ajudar no processo de recomendação de matérias. Contúdo, identificar entidades nomeadas segundo [14], é uma das tarefas mais importantes e complexas do processamento de linguagem natural.

A dificuldade aumenta nesta área por conta da falta de corpus para algumas línguas o que é o caso da língua portuguesa. Alguns trabalhos em lingua portuguesa foram observados como por exemplo, o trabalho de [13], utiliza sintagmas nominais para classificação automática de documentos.

Dada a dificuldade de encontrar uma ferramenta que pudesse ser usada para apoio a esta atividade, partimos para o desenvolvimento de uma solução caseira utilizando uma abordagem simples e um corpus de entidades mapeadas.

### O classificador

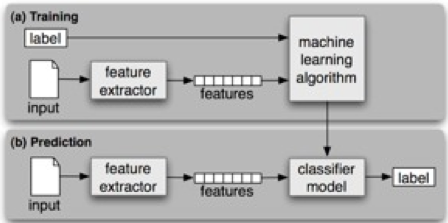
Para nosso trabalho não era necessário conhecer a que categoria a entidade pertencia, se ela representava uma organização ou uma pessoa. Não utilizamos este complemento de informação para melhorar a recomendação. Por este motivo, nos limitamos a descobrir se um determinado n-gram correspodia a uma entidade ou não.

Desta forma a idéia passou a ser a de produzir um classificador binário bayesiano que respondesse a probabilidade de um n-gram representar uma entidade.

Para melhor entender o que é um classificador binário bayesiano, vamos primeiro lembrar o conceito de classificação.

Classificação é uma das técnicas existentes em aprendizado de máquina que segundo [7], é definida como a tarefa de atribuir um determinado rótulo a um dado texto de entrada de acordo com o reconhecimento de um padrão existente. Ou seja, o classificador aprende um padrão e com isso rotúla as instâncias que correspondem a esse padrão.

Table 9 – Cassificador



O padrão é encontrado pelo classificador com o auxilio de features que são passadas para ele. As features são dicas de informações que possibilitem ao classificador o entendimento do corpus. Por exemplo a primeira letra de um n-gram.

De forma simples, um classificador bayesiano é um classificador probabilístico que parte do pressuposto que a presença ou ausência de uma feature de uma classe não esta relacionada a presença ou ausência de outras features da mesma classe.

Desta forma o classificador entende as features isoladamente atribuindo probabilidades para cada feature.

Em nosso caso, a features foram montadas de acordo como a combinação da classe gramatical dos n-gram, do formato da primeira letra do n-gram (se maiuscula ou minuscula) e do número de palavras existentes no n-gram.

Com isso tivemos a seguinte composição de features:



pseudo-código 7 - Definição de Feature

A respeito do apreendizado, dizemos que a nossa classificação é supervisionada onde um conjunto de dados é separado para servir de base de aprendizado pelo classificador. De acordo com o entendimento desta base, o classificador consegue inferir qual a categoria correta para uma nova instancia. Assim sendo, utilizamos um corpus de 31814 entidades entre nomes de pessoas, lugares e organizações extraídas da base semântica da globo.com.

Com essas entidade pudemos percorrer os textos das matérias do corpus e encontrar as sentenças onde elas apareciam para extraír as features do classificador. Todavia, ainda precisavamos desenvolver uma outra parte do problema para treinar o classificador, O **pos-tagger**.

### O pos-tagger

Segundo [7], **part-of-speech tagging** é o processo de converter uma sentença, no formato de uma lista de palavras, para uma lista de tuplas onde cada tupla é composta de uma palavra e uma tag e essa tag se refere a classe gramatical da palavra. Em outras palavras, o pos-tagger é o mecanismo desenvolvido para identificar a classe gramatical de palavras em um determinado texto.

Para treinar o nosso pos-tagger utilizamos o corpus macmorpho disponível no nltk toolkit. O MAC-MORPHO é um corpus em língua portuguesa do brasil que possui milhões de palavras extraídas de artigos jornalísticos divididos em 10 seções diferentes do jornal a folha de são paulo em 1994.

Em seguida, utilizamos o módulo de Tagger da API do nltk para treinar o nosso tagger de modo que obtivemos a acuracia de aproximadamente 90%.



Figure 9 – Acurácia do Pos-Tagger

Treinamento do Classificador Nayve Bayes

Corpus do Classificador

Resultados e Acurácia

Considerações Finais