USO DE TABELA DE DISPERSÃO NA CORREÇÃO DE GRAFIA INCORRETA: ESTUDO COM APLICAÇÃO DE ALGORITMOS

MACEDO, Paulo Cesar
Faculdade Santa Lúcia
paulouniararas@gmail.com

MORAES, Marcos Roberto de
Faculdade Santa Lúcia
professormoraes@gmail.com

RESUMO

A Internet vem experimentando um crescimento expressivo desde a década de 1990. Além do desenvolvimento tecnológico, vem ocorrendo um amadurecimento desta rede como uma ferramenta de comunicação. Aplicações em comunicação pessoal, comércio, jornalismo e marketing são consideradas essenciais no dia a dia das empresas e instituições. Observando especificamente a área de marketing, é possível definir de maneira muito precisa o perfil de um consumidor analisando-se sua participação na Internet. Na utilização da rede mundial por parte dos usuários, é possível identificar seu comportamento, preferências, mudanças de humor e hábitos através de sua participação em chats, fóruns, comunicadores instantâneos, ferramentas de redes sociais e outras formas de comunicação online. O objetivo deste trabalho é apresentar um algoritmo de sugestão e correção ortográfica de comentários deixados por usuários em redes sociais, como parte de um trabalho maior envolvendo a classificação qualitativa de comentários pelas empresas de monitoramento. Justifica-se para tal, a grande quantidade de erros ortográficos encontrados nesses comentários, sendo este um grande desafio para a área da tecnologia de informação. Como resultado o algoritmo proposto obteve êxito em 85,7% dos erros ortográficos encontrados.

Recebido em 10-02-10 • Aceito em 16-08-10

PALAVRAS-CHAVE: *erro ortográfico; comentários; Internet; índice de produto; algoritmo.*

INTRODUÇÃO

A análise das mídias sociais na *Internet* pode-se tornar um diferencial para as empresas, de forma a colaborar com o relacionamento entre os consumidores e seus produtos. Segundo Gartner (2010), elas podem encontrar formas de potencializar a afinidade do consumidor com suas marcas por meio do que se fala na *Internet*. No cenário em que estamos, é evidente que a palavra do consumidor, em uma mídia pública como a *Internet*, pode afetar o desempenho e o prestígio de uma empresa, ou seja, "quem manda no jogo é o consumidor" (*grifo nosso*). Os diversos tipos de mídias sociais existentes possibilitam aos consumidores, enquanto Internautas, se relacionarem através de diálogos e criarem conteúdos muitas vezes significativos sobre uma marca, serviço, pessoa ou produto. Esses conteúdos são deixados nos *sites* das empresas ou em redes sociais por meio de comentários, que por sua vez são analisados por empresas de monitoramento de forma manual ou utilizando ferramentas com o objetivo de classificá-los quanto ao fato de serem bons, ruins ou neutros.

O grande desafio para essa classificação é justamente a possibilidade de captar erros de ortografia. Essa dificuldade técnica faz com que eles sejam ignorados pelas ferramentas de monitoramento. Levando em conta que os usuários, na maioria dos casos, cometem erros de digitação ou erros ortográficos em seus comentários, centenas deles podem ser descartados de uma análise. A solução proposta é a criação de um algoritmo capaz de reconhecer palavras de um dicionário pré-definido e usar esse reconhecimento em um sistema de classificação. Como experimento serão utilizados conteúdos com erros captados aleatoriamente na *Internet*, com posterior aplicação do algoritmo e apresentação dos resultados obtidos. Na composição da metodologia fixou-se como universo de pesquisa as redes sociais presentes na *Internet;* como método de coleta de dados, a escolha aleatória de comentários de tais redes sociais; como procedimentos, foram utilizados a pesquisa bibliográfica e o levantamento das informações deixadas nos comentários das pessoas nas redes sociais a fim de conhecer seu comportamento.

Recebido em 10-02-10 • Aceito em 16-08-10

2. COMENTÁRIOS DA *INTERNET*

Este artigo é parte integrante de uma pesquisa que aborda o monitoramento de comentários por empresas de marketing na *Internet*. Esses conteúdos, que versam sobre produtos ou serviços nas redes sociais, segundo essas empresas, podem ser classificados como bons, ruins ou neutros (KUKICH, 1992). Neste caso, por exemplo, as ferramentas de monitoramento teriam grande parte de seus comentários coletados descartados por conta de erros de grafia cometidos por mau uso da língua. Esse descarte poderia ser evitado se pelo menos os adjetivos que qualificam os comentários fossem entendidos. Segundo Pollock e Zamora (1984), algumas pesquisas apontam que a estratégia de correção mais adequada para um texto publicado na *Internet*, muitas vezes depende da sua natureza e sua origem. Nesse caso específico, isto é, dos comentários das redes sociais, a ideia é não deixar de qualificá-los apenas por possuir alguns erros.

Esses comentários da *Internet*, na maioria das vezes, ocorrem utilizando uma linguagem especial, cheia de abreviações e combinadas a um conjunto enorme de caracteres especiais disponíveis nos computadores. De acordo com Ringlstetter, Schulz e Mihov (2006), os conjuntos de caracteres utilizados para escrever páginas ou comentários na *Web* muitas vezes não são totalmente adequados para o alfabeto de uma determinada língua, o que representa outra fonte sistemática de imprecisões. Para os utilizadores, isso não é considerado um problema, pois quando leem um texto na *Internet*, seja em forma de uma notícia ou um simples comentário, o usuário consegue processar e entender o que está escrito sem maiores problemas. Por exemplo, algumas palavras como: Beleza abrevia-se **blz**; com abrevia-se **c** ou **c/**; como torna-se **cmo** ou **cm**; comigo abrevia-se **cmg**; imagem abrevia-se **img**; firmeza abrevia-se **fmz**; falou abrevia-se **flw** .”Falow” é uma gíria para “**tchau**”; mais torna-se **+**; menos torna-se **-**; não se abrevia **ñ, n, non, naum, num ou nao**; para se abreviar **p**/, **pra**, **pa** ou **p**; também se abrevia **tbm**; **tb** ou “**tmb**”; teclar abrevia-se **tc**, que é uma gíria para “conversar usando o teclado”; quando se abrevia **kdo**, **qdo**, **qd** ou **qnd** (“quando” não é usado como **kd**, pois é uma forma abreviada de cadê); quanto se abrevia **kto**, **qto** ou **qnt**; tchau abrevia-se **xau**, **tiaw** ou **tiau**; vezes abrevia-se **vzes**; ver abrevia-se **v** (RECUERO, 2008, grifos do autor).

Esses usuários criaram o que alguns chamam de “Internetês” (*grifo nosso*), que é outro agravante para ferramentas de análise textual. Trata-se de um idioma exclusivo para *Internet*, composto de abreviações e sinais que formam uma gama de novas palavras. Pelo grande número de palavras

Recebido em 10-02-10 • Aceito em 16-08-10

utilizadas, seria necessária a criação de um dicionário exclusivo para *Internet* que descrevesse todas essas variações. Além das abreviações, o usuário comete alguns erros de digitação, podendo dessa forma escrever palavras com letras trocadas, letras demais, letras de menos, ou ainda cometer erros totais de ortografia do idioma.

Um dos problemas encontrados quanto à utilização desse tipo de idioma é que as pessoas ou ferramentas que buscam o entendimento desses textos têm dificuldade em entendê-los. Para Damerau (1964)[1], mesmo quando se utiliza de ferramentas, dificilmente podemos evitar uma possível correção manual. Algumas técnicas de correção de ortografia são limitadas no seu alcance e precisão (KUKICH , 1992).

3. ESTADO DA ARTE

Desde o surgimento da *Internet*, nos meados da década de 1960 até 2010, várias ferramentas foram concebidas apresentando ótimas performances. Naturalmente, os verificadores gramaticais melhoraram as correções de textos, no entanto, esse processo requer grandes recursos. Algumas empresas na grande rede como o *Google*, por exemplo, mantêm uma base de dados e conhecimento sobre todas as pesquisas solicitadas em seu mecanismo de busca, fazendo sua correção com um algoritmo que utiliza o reconhecimento de frases comparando com ocorrências dos termos pesquisados em seu site de busca (JACQUEMONT; JACQUENET; SEBBAN, 2007).

Alguns algoritmos que utilizamos como referência aplicam técnicas de correção através de analisadores morfológicos que abordam problemas de inflexão do texto e de frases compostas por gírias e costumes de uma região (CHAUDHURI, 2002).

Outros algoritmos propostos herdam técnicas da distância Levenshtein (1966)[2], que é uma métrica para medir a quantidade de diferenças entre duas sequências, ou seja, o chamado editar distância. A distância entre duas sequências é dada pelo número mínimo de operações necessárias para transformar uma string em outra, quando uma operação é uma inserção, exclusão ou substituição de um único caractere (LEVENSHTEIN, 1966).

É de conhecimento notório que ferramentas na área de monitora-

[1] Este autor trata de uma referência clássica e básica das ciências da computação para esse assunto.
[2] A distância de Levenshtein é parte de um artigo clássico das ciências da computação para esse assunto.

Recebido em 10-02-10 • Aceito em 16-08-10

mento *Web* utilizam PLN (Processamento de Linguagem Natural)[3] para resolver seus problemas de correções de textos. É importante ressaltar a questão léxica dos elementos de linguagem principalmente na forma de concordância verbal que ainda permeia uma série de discussões sobre a implementação. Existem diversas propostas de uso de técnicas de PLN para colaborar com esse processo que prometem tornar todo processo de correção e entendimento mais rápido.

4. METODOLOGIA

Neste trabalho foi proposta uma forma de classificar as frases dos comentários feitos por usuários na *Internet*, de acordo com a **Tabela 1.**

Tabela 1 – Classificação das frases dos comentários

CLASSIFICAÇÃO	DESCRIÇÃO
Frase positiva	Trata-se de um comentário com adjetivos que comprovam a satisfação do usuário com o contexto analisado. Retrata ainda a ideia de que o produto ou serviço atendeu suas expectativas plenamente.
Frase negativa	É um comentário que contém adjetivos que demonstrem a insatisfação do usuário com um serviço ou produto. Normalmente usando palavras que agridem e/ou criticam a qualidade do que se oferece.
Frase neutra	Entende-se por um comentário que não oferece elementos de qualidade ao contexto, ou seja, não critica e nem elogia o produto ou serviço analisado. Muitas vezes, não expressa através de adjetivos sua satisfação ou insatisfação.
Frase nula	É um comentário que não pode ser classificado como positivo, negativo ou neutro devido seus erros de grafia. Esse tipo de comentário não permite a compreensão pelas ferramentas de monitoramento fazendo com que a falta de um padrão e o não entendimento das palavras no idioma analisado descartem o mesmo.

Fonte: Adaptado de Kukich (1992, p. 377)

As frases positivas, negativas e neutras são reconhecidas e classifi-

[3] O PLN pode ser definido como o processo de utilização de conhecimentos sobre a língua e a comunicação humana, tanto para a comunicação com sistemas computacionais como para melhorar a comunicação entre os seres humanos (BARROS, 2004).

Recebido em 10-02-10 • Aceito em 16-08-10

cadas por ferramentas de monitoramento, devido a seus adjetivos estarem, na maioria das vezes, sem erros de ortografia, o que não acontece com as frases nulas.

Este artigo atua sobre as frases classificadas como nulas, tentando assim diminuir a perda de comentários causados por erros principalmente em seus adjetivos.

O que há de mais importante para o monitoramento dos comentários da *Internet*, e o que precisamos para classificá-los são os adjetivos, ou seja, são eles que indicam se o comentário sobre um produto ou serviço é bom, neutro ou ruim.

Para resolver o problema mais específico de não descartar o comentário com erros, este estudo propõe a utilização de um algoritmo que funcionará da seguinte forma: o comentário é submetido a um teste que irá isolar palavra por palavra, e em seguida confrontá-las com uma lista de palavras do idioma escolhido (dicionário). O resultado é um grupo de palavras candidatas a estarem erradas, pois não constam da lista de palavras do idioma, sendo que, palavras com menos de quatro letras ou símbolos (imagens) colados no comentário serão ignorados pelo algoritmo. Essas palavras são adicionadas a uma coleção conhecida como *HashTable* que segundo Boswell (2004) é uma coleção onde os elementos possuem chave/valor, e onde cada elemento é indexado usando uma chave alfanumérica. Ao trabalhar com a classe *HashTable* deve se ter em mente que a ordenação dos elementos desta coleção independe da ordem na qual foi adicionado na tabela. A classe emprega seu próprio algoritmo *hash* para ordenar de forma eficiente os pares chave/valor deste tipo de coleção. A classificação não irá alterar a integridade do comentário e será utilizada apenas para contribuir para a sua identificação, podendo ser um comentário bom, ruim ou neutro. A **Figura 1** apresenta sucintamente como se dará o processo de correção.

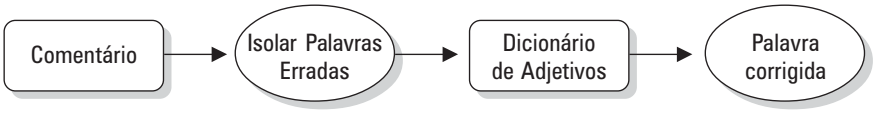


Figura 1 - Processo de correção

Explicando mais detalhadamente, tem-se um algoritmo onde primeiramente cada comentário é analisado de forma a converter todo texto em letras maiúsculas, em seguida cada palavra é submetida a um dicionário normal da língua desejada, posteriormente as palavras não encontradas

Recebido em 10-02-10 • Aceito em 16-08-10

nessa lista serão segmentadas e colocadas em uma *HashTable* de palavras candidatas a correção.

Vamos tomar como exemplo uma palavra errada:

"Palaivra" - A segmentação consiste na retirada da primeira letra para filtragem da palavra candidata da lista e o restante divide-se em triplas a serem armazenadas no *hash* conforme a **Figura 2,** obtendo as seguintes combinações como resultado: 1."ala", 2."lai", 3."aiv", 4."ivr", 5. "vra" (grifos nossos).

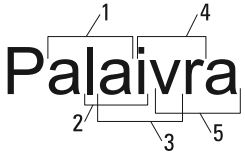


Figura 2 - Palavra segmentada

A letra inicial irá ser usada para filtrar uma lista de adjetivos relevantes à classificação do comentário, esse filtro irá trazer apenas as palavras iniciada pela letra especificada, deixando o processo mais rápido. Finalmente cada palavra será segmentada em triplas, no caso do exemplo em cinco triplas e confrontada com as triplas da palavra errada armazenadas na *HashTable*. A palavra do dicionário que obtiver o maior número de casamentos de triplas será a mais indicada à substituição.

Comparando, temos uma lista com as palavras candidatas a ser a palavra correta de acordo com algumas regras de desempate:

a) A quantidade de triplas iguais encontradas no *HashTable*;
b) O tamanho da palavra, ou seja, a quantidade de letras correspondentes mais aproximadas;
c) A ordem alfabética das palavras encontradas.

A **Figura 3**, apresenta uma lista das possibilidades, e mostra também a quantidade de combinações encontradas na *HashTable* que, nesse caso, são as triplas "ala" e "vra" totalizando 8 pontos, porém a ordem de tamanho indica a melhor candidata (grifos nossos).

A *HashTable* possui recursos para que as triplas sejam adicionadas, consultadas ou removidas se necessário. Nesse caso, a ordem de entrada na *HashTable* foi o critério adotado para pontuar a palavra adequada, ou seja, a candidata a substituição.

Recebido em 10-02-10 • Aceito em 16-08-10

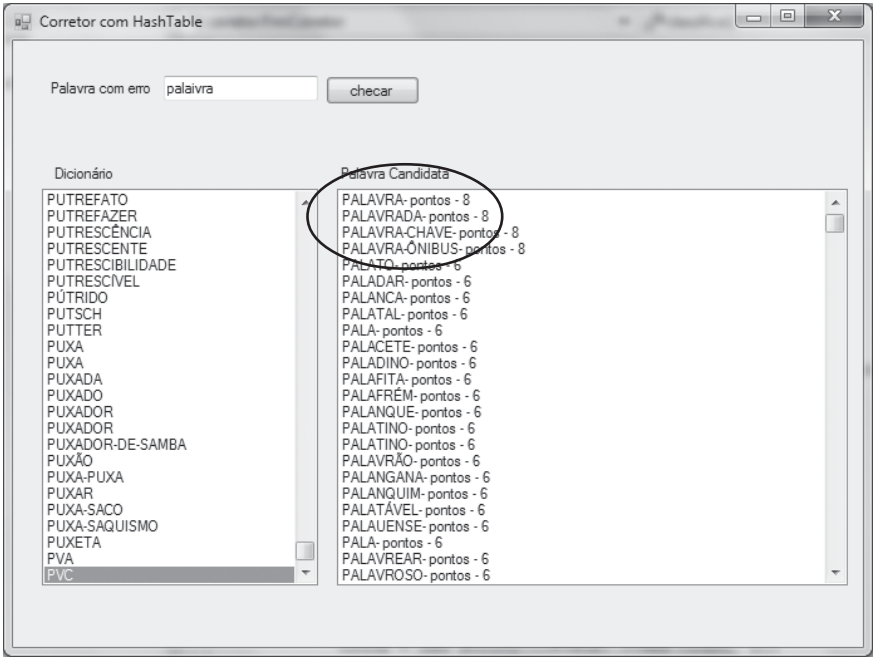


Figura 3 - Exemplo da correção

A tripla da primeira posição deve ter melhor pontuação que uma encontrada no meio da palavra, pois através dela se caracteriza melhor a palavra candidata. Como as palavras possuem tamanhos diferentes, não podemos fazer uma escala de pontos concreta. Sendo assim, adotamos pontuações de acordo com a quantidade de triplas encontradas, multiplicando pela posição da tripla na *HashTable*.

Para exemplificar e aplicar esse experimento vamos entender como o protótipo criado funciona:

- Passo I: criou-se uma base de dados para comportar o dicionário. A **Figura 4** mostra a base criada.

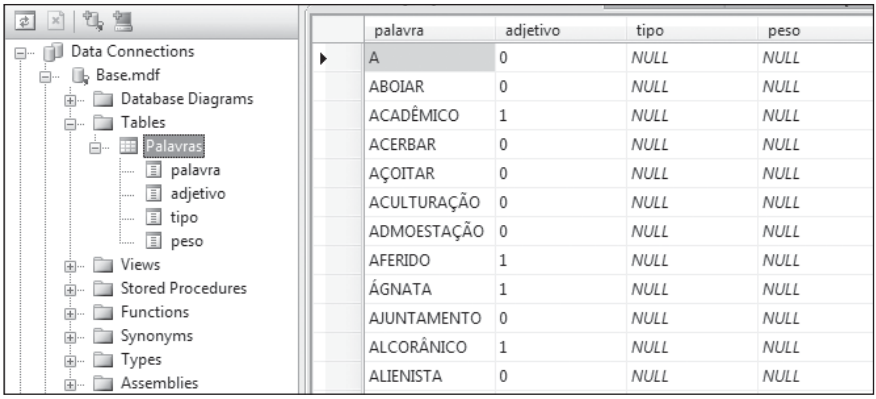
Esse dicionário de palavras foi elaborado no idioma Português, utilizando o *SQLExpress* da empresa *Microsoft*. Após todas as palavras acomodadas em nossa tabela, criou-se uma lógica para utilizá-lo através da linguagem *C#*. A **Figura 5** ilustra o código necessário.

- Passo II: acionamento do corretor.

Criou-se um botão que, após seu acionamento, cria uma *HashTable* para ser utilizada. Em seguida filtra na base de dados (dicionário) as pala-

Recebido em 10-02-10 • Aceito em 16-08-10

vras começadas pela mesma letra da palavra errada. Finalmente submetemos a palavra para classificar a que melhor se encaixa.
- Passo III : foi criado então o filtro destacado na **Figura 6** para separar as palavras candidatas no dicionário começado pela mesma letra.



palavra	adjetivo	tipo	peso
A	0	NULL	NULL
ABOIAR	0	NULL	NULL
ACADÊMICO	1	NULL	NULL
ACERBAR	0	NULL	NULL
AÇOITAR	0	NULL	NULL
ACULTURAÇÃO	0	NULL	NULL
ADMOESTAÇÃO	0	NULL	NULL
AFERIDO	1	NULL	NULL
ÁGNATA	1	NULL	NULL
AJUNTAMENTO	0	NULL	NULL
ALCORÂNICO	1	NULL	NULL
ALIENISTA	0	NULL	NULL

Figura 4 - Base de dados criada

```
private void button1_Click(object sender, EventArgs e)
        {
                //Constrói o Hashtable
                meuhash= new Hashtable();
                //pega palavra errada
                palavra = textBox1.Text.ToUpper();
                //filtra o listbox1 com palavras
                //oriundas do dicionário começadas
                //pela primeira letra da palavra
                //errada
                filtrarGrid();
                //método para classificação das
                //sugestões de palavras
                 classifica();
        }
```

Figura 5 - Botão para acionar a correção

Recebido em 10-02-10 • Aceito em 16-08-10

```
private void filtrarGrid()
        {
                listBox1.DataSource = null;
                listBox1.Items.Clear();
                char[] letra = palavra.Substring(0, 1).ToCharArray();
                clsPalavras dados = new clsPalavras();
                listBox1.DisplayMember = "palavra";
                listBox1.DataSource = dados.BuscarLetra(letra[0]).Tables[0];
        }
```

Figura 6 - Filtro de palavras na *ListBox*

Para esse filtro foi utilizado uma classe de palavras (*clsPalavras*) que consome uma Classe de Dados (*clsDados*) para trazer as informações do banco de dados.

- Passo IV – A classificação da palavra mostrada na **Figura 7** apresenta a lógica mais importante para solução do algoritmo.

```
private void classifica()
{
    if (palavra.Length > 3)
        {
        string primeiraletra = palavra.Substring(0, 1);
        int size = palavra.Length;
        int vetor;
        vetor = size - 3;
        string resto = palavra.Substring(1, size - 1);
        for (int i = 0; i < vetor; i++)
            {
                meuhash.Add(i, resto.Substring(i, 3));
            }
        array = new string[listBox1.Items.Count, 2];
         for (int j = 0; j < listBox1.Items.Count; j++)
            {
                listBox1.SelectedIndex = j;
                palavra_list = listBox1.Text;
                int sizej = palavra_list.Length;
                string restoj = palavra_list.Substring(1, sizej - 1);
                vezes = sizej - 3;
                pontos = 1;
```

Figura 7 - Classificação das palavras candidatas (Parte I)

Recebido em 10-02-10 • Aceito em 16-08-10

```
for (int k = 0; k < vezes; k++) {
    if (meuhash.ContainsValue(restoj.Substring(k, 3)))
        {
            if (palavra.Length == palavra_list.Length){
            if (k == 0)pontos += (pontos * meuhash.Count);
            if (k == 1)pontos += (pontos * (meuhash.Count - 1));
            if (k == 2)pontos += (pontos * meuhash.Count - 2);
            if (k == 3)pontos += (pontos * (meuhash.Count - 3));
            }else{
            if (k == 0)pontos += meuhash.Count;
            if (k == 1)pontos += (meuhash.Count - 1);
            if (k == 2)pontos += (meuhash.Count - 2);
            if (k == 3)pontos += (meuhash.Count - 3);
                }
            }
        }
        array[j, 0] = palavra_list;
        array[j, 1] = Convert.ToString(pontos);
        pontos = 0;
            }
        }
```

Figura 7 - Classificação das palavras candidatas (Parte II)

Esse passo é o mais importante de todo processo de correção. Após a palavra errada ser digitada (captada do comentário), ela é dividida em triplas e colocada na *HashTable*, em seguida cada palavra filtrada do dicionário é submetida a um processo de localização da tripla e comparada com a mesma no intuito de efetuar a pontuação adequada. Essa pontuação, como já foi dito, consiste em analisar a posição da tripla encontrada e multiplicar por sua posição na *HashTable*. Dependendo da posição, a tripla pode ser melhor ou pior pontuada.

No final todos os pontos são classificados bem como uma ordenação por tamanho, onde é considerado o tamanho da palavra certa com relação à palavra errada.

Portanto, este entendimento complementa ferramentas de monitoramento e qualificação dos comentários, evitando o seu descarte devido a erros ortográficos.

4.1 EXPERIMENTO

Para comprovar o funcionamento do algoritmo proposto, utilizou-

Recebido em 10-02-10 • Aceito em 16-08-10

se um método onde cada palavra significativa de um comentário foi isolada das demais e comparada à lista de adjetivos conforme descrito na proposta. Essas palavras, que segundo Hu e Wong (2003), são conhecidas como *tokens*, e que em computação significa um segmento de texto ou símbolo que pode ser manipulado por um *parser*. Em outras palavras, *token* é um conjunto de caracteres (de um alfabeto, por exemplo) com um significado coletivo. Ainda para Hu e Wong (2003) o *parser* é um programa de computador (ou apenas um componente de um programa) que serve para retirar cada *token* para analisar a estrutura gramatical de uma entrada, manipulando-o. Neste artigo foram utilizados esses conceitos de forma a exemplificar sua utilização no aproveitamento e entendimento de textos extraídos aleatoriamente da *Internet*, mais especificamente, das redes sociais.

A **Tabela 2** apresenta alguns fragmentos de textos encontrados aleatoriamente através de uma pesquisa na *Internet*.

Tabela 2 - Comentários da *Internet* (grifos nossos)

COMENTÁRIO	FONTE/SITE (MÊS – ANO)
Excelente vendedor, **porduto** muito bom, negociação rápida e tranquila, obrigado... Indico a todos do ML....	Mercado livre (Março – 2010)
Jogo Ps3 NFS Pro Street **Exelente** estado Europeu Completo	Mercado Livre (Março – 2010)
Muito bom tem um manual fácil uso bom **porduto**	Bom de Faro (Março – 2010)
Anyway…um **porduto** muito bom, **soh** o preço que realmente **eh** salgado	Brasil Hawaii (Março – 2010)
Esselente Produtudo marca **rementada** garantia boa,**costei** de compra **besta** loja	Buscapé (Março – 2010)
NÃO COMPREM - PRODUTO É **POCARIA**. E já soube que - depois de voltar da **asistêencia**, terei inúmeros problemas com ela.	Reclame aqui (Março – 2010)
Com 60% da população brasileira, é o **comprexo** regional que concentra as maiores indústrias e maiores cidades.	Blogspot: Assuncaoturma221 (Março – 2010)

Recebido em 10-02-10 • Aceito em 16-08-10

COMENTÁRIO	FONTE/SITE (MÊS – ANO)
Isso me deixou **flustado** pois esperava que essa placa rodasse pelo menos em 800x600 no high pro evolution soccer 5...	Fórum: Clube do hardware (Março – 2010)
O preço está bom, vem bastante produto nas embalagens. Me **decepisionei** um pouco com a demora para chegar os produtos.	Blogspot: Beaultiful (Março – 2010)
Foi feita para spyder nitro, porem a litragem vai servir **prefeitamenta** pra vários outros sub s.	Mercado Livre (Março – 2010)

As palavras destacadas na **Tabela 2** possuem erros ortográficos da língua portuguesa e foram submetidas ao algoritmo. Segundo Ringlstetter, Schulz e Mihov (2007), a maioria das ferramentas de correção são construídas sobre dicionários estáticos que representam, de maneira fixa, coleções de expressões de uma determinada língua. Partindo dessa premissa, nosso dicionário estático irá conter apenas adjetivos, fundamentais para entender o que se pretende.

Todas as palavras serão submetidas a correção, e serão retiradas do texto aquelas que apresentarem erros de grafia. Algumas dessas palavras com erros não irão prejudicar a classificação do comentário, as ferramentas podem usar apenas os adjetivos para identificar e classificar o mesmo. As palavras contendo erros que teoricamente não venham a fazer diferença devem ser levantadas, pois podem mudar o sentido de uma frase se não forem devidamente corrigidas. Na impossibilidade de correção, essas palavras serão ignoradas pela ferramenta no intuito de não prejudicar um possível entendimento do restante do texto.

No código dessa checagem serão cortados possíveis espaços existentes no início e no final de toda a frase e dividir o texto, ou seja, um conjunto de palavras individuais que usaremos para compará-las com o arquivo de palavras (utilizado neste estudo). Cada palavra da frase deve filtrar as palavras existentes na lista que comecem com a mesma letra. Caso encontre a palavra completa, pode-se passar para a próxima; caso contrário, deve-se seguir com o algoritmo até o final, isolando as palavras erradas.

Para esse experimento foram utilizados os comentários da **Tabela 2**. Cada palavra dos comentários foi colocada em minúsculo para reduzir o tempo de procura bem como o de substituição. Foram obedecidas regras de

Recebido em 10-02-10 • Aceito em 16-08-10

probabilidade para verificar se uma palavra sugerida é a mais correta, mas não há nenhuma maneira de saber ao certo, ou seja, não se pode garantir 100% das substituições corretas, mesmo porque o comentário do internauta não será mudado, mas compreendido.

Algumas palavras do texto poderiam ter sugestões que não satisfazem corretamente o que se propõe no comentário. Por exemplo, na frase: "A região sudeste com 60% da população brasileira, é o **comprexo** regional que concentra as maiores indústrias e maiores cidades." (*grifo nosso*).

A palavra "comprexo" poderia ter como primeira sugestão a palavra "completo" e mudar o sentido da frase, quando deveria ter a palavra "complexo" como melhor sugestão.

Obtendo suas triplas "T" e usando a *HashTable* temos:

"Comprexo" - T1= "omp", T2= "mpr", T3= "pre", T4= "rex" e T5= "exo"
Como palavras candidatas temos:
"Complexo" - T1= "omp", T2= "mpl", T3= "ple", T4= "lex" e T5= "exo"
"Completo" - T1= "omp", T2= "mpl", T3= "ple", T4= "let" e T5= "eto"
Quantidades de combinações iguais:
"Complexo" – 2
"Completo" – 1 (grifos nossos).

Tomando-se como exemplo a ferramenta de correção ortográfica do editor de textos *Word* da empresa *Microsoft*© na **Figura 6**, teríamos a palavra correta apenas como terceira opção de substituição, enquanto que

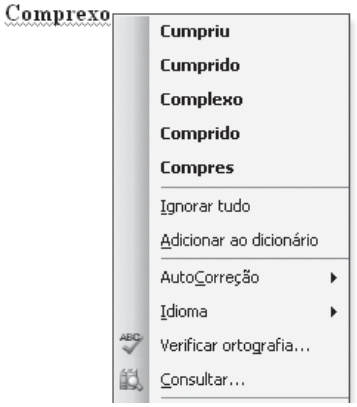


Figura 6 - Corretor ortográfico

Recebido em 10-02-10 • Aceito em 16-08-10

com a solução proposta seria a primeira opção seguramente, pois como podemos observar a palavra "complexo" obteve maior pontuação na comparação da *HashTable* (*grifo nosso*).

Algumas soluções apontam que um algoritmo deve considerar a distância (*string* métrica) entre duas sequências, ou seja, finito de símbolos, determinada pela contagem do número mínimo de operações necessárias para transformar uma string em outro, quando uma operação é definida como uma inserção, exclusão ou substituição de um único caractere, ou uma transposição de dois ou mais caracteres (LEVENSHTEIN, 1966; ULLMANN, 1977).

Observa-se que nem todas as palavras serão atendidas, mas uma grande parte dos erros será coberta. Este algoritmo visa atender expectativas em torno de 80% das palavras erradas, podendo assim ser considerado eficaz em sua proposta. Para construir um modelo que atingisse porcentagens maiores talvez teríamos que analisar também as sequências de letras, usando modelos tradicionais com frases prontas armazenadas e combinar soluções. Esses tipos de soluções são atualmente utilizados por *sites* de busca como o Google que mantém enormes bancos de dados com combinações de frases (JACQUEMONT; JACQUENET; SEBBAN, 2007).

A **Tabela 3,** apresenta as palavras com erros encontradas nos comentários e sua classificação de acordo com o método adotado (*HashTable*), lembrando que as palavras que não eram adjetivos foram ignoradas pelo algoritmo:

Tabela 3 - Classificação das palavras

PALAVRA ENCONTRADA	PALAVRA CORRETA	COMBINAÇÕES DE *HASH*
Esselente	Excelente	4 (ele,len,ent,nte)
Rementada	———	———
Costei	———	———
Pocaria	Porcaria	3 (car,ari,ria)
Asisteencia	———	———
Comprexo	Complexo	2 (omp,exo)
Flustado	Frustrado	2 (ust,ado)
Decepsionei	Decepcionado	3 (ece,cep,ion)
Prefeitamenta	Perfeitamente	7 (fei,eit,ita,tam,ame,men,ent)

Após o uso do algoritmo implementado nas palavras encontradas, pode-se concluir que o algoritmo apresentou taxas de acertos e de classificação satisfatórios, conforme **Tabela 4.**

Recebido em 10-02-10 • Aceito em 16-08-10

Tabela 4 - Resultados dos erros encontrados no estudo

Totalizando erros encontrados	Quantidade	Porcentagem
Adjetivos com erros	7	100,0%
Correções efetuadas	**6**	**85,7%**
Sem correção, ignorados	3	42,8%
Não adjetivos	2	28,5%

Ao se analisar os resultados obtidos, percebe-se que os erros tabulados mostram que algumas palavras possuem a necessidade de se aplicar mais de um método de correção, ou seja, algumas precisaram de inserção e/ou substituição de caracteres.

Ao fim do experimento, obteve-se de forma positiva 85,7% (em destaque na **Tabela 4**) de correções efetuadas com sucesso. É importante observar ainda que algumas palavras foram consideradas "Sem correção", pois não foram encontradas no dicionário de adjetivos, portanto não substituídas (*grifo nosso*).

CONSIDERAÇÕES FINAIS

Estratégias para desenvolvimento de bons corretores ortográficos sempre têm sido um grande desafio para pesquisas computacionais. As empresas utilizam ferramentas proprietárias e algoritmos que desconhecemos para solucionar esse tipo de problema. Este artigo buscou, em síntese, apoiar desenvolvedores a criar seus algoritmos de forma simples e utilizá-los sem depender de soluções de alto custo. Buscou também explicar o que acontece, como funcionam essas ferramentas que utilizamos em nosso dia a dia além de mostrar como os usuários de *Internet* desprezam a língua em sua forma correta. Os comentários foram testados para comprovar que mesmo com erros poderiam ser classificados e não precisavam ser descartados durante o seu monitoramento. Os resultados mostraram que o algoritmo proposto obteve êxito em 85,7% dos erros, mas como foi dito anteriormente, erros que tiram completamente o sentido do comentário ou palavras inexistentes no dicionário pré-estabelecido, tornam inviável sua análise, por isso alguns foram descartados. Além disso, deve-se levar em consideração questões de como avaliar o número de classificações incorretas, ou seja, o algoritmo reconheceu a palavra, mas substituiu por uma errada e comprometeu o comentário. E o número de perdas, será que o comentário

Recebido em 10-02-10 • Aceito em 16-08-10

perdido faria diferença na análise final? Questões como essas nos fazem refletir o quão longo é o caminho para entendimento definitivo de um texto dentro do que se espera das ferramentas automáticas de monitoramento. Ferramentas como o *Você quis dizer* (grifos nossos) da empresa *Google*, apostam em estatísticas armazenadas em seus bancos de dados para supor o que o usuário pretende quando digita uma palavra/frase em seu campo de pesquisa. Talvez novas estratégias e aplicações de PLN (Processamento de Linguagens Naturais), combinadas com *HashTable* e expressões regulares possam contribuir nesse sentido e fazer com que as porcentagens de acertos sejam maiores.

REFERÊNCIAS BIBLIOGRÁFICAS

BARROS, L.A.. **Curso Básico de Terminologia**, São Paulo: Editora da Universidade de São Paulo, 2004.

BOSWELL D.. **Language models for spelling correction**, CSE 256, Spring 2004

CARLOS, J. L.. Análise das formas de comunicação escrita utilizadas em situações de interação verbal e língua portuguesa pela *Internet*. **Revista INICIE, 2(1)**. Disponível em: http:// www6.mackenzie.br. Acesso em junho de 2008.

CHAUDHURI, B.B.. Towards Indian language spell-checker design. **Language Engineering Conference, 2002.** Proceedings , vol., nº, pp. 139-146, 13-15 Dec. 2002.

DAMERAU F.J.. **A technique for computer detection and correction of spelling errors**. Commun. ACM, Vol. 7, Nº –3, pp. 171-176, 1964.

GARTNER. **Technology Business Research Insight** - Tendências de Ti para os próximos anos. Disponivel em: http://www.metaanalise.com.br/inteligenciademercado/palavra-aberta/analise-setorial/gartner-destaca-as-tendencias-de-ti-para-os-pr-ximos-anos.html Acesso em fevereiro de 2010.

HU, K.; WONG, W. ; S. A.. probabilistic model for intelligent Web crawlers, **Computer Software and Applications Conference, 2003. COMPSAC 2003. Proceedings. 27th Annual International** , vol., nº, pp. 278-282, 3-6 Nov. 2003.

JACQUEMONT, S.; JACQUENET, F. ; SEBBAN, M. . Correct your text with Google. Web Intelligence, IEEE/WIC/ACM. **International Conference on** , vol., nº, pp.170-176, 2-5 Nov. 2007

Recebido em 10-02-10 • Aceito em 16-08-10

KUKICH, K.. **Techniques for automatically correcting words in text**. ACM Comput. Surv. 24, 4 (Dec. 1992), 377-439.

LEVENSHTEIN, V. I.. **Binary Codes Capable of Correcting Deletions, Insertions and Reversals**. Soviet Physics Doklady, vol. 10, February 1966.

POLLOCK, J. J. ; ZAMORA, A.. **Automatic spelling correction in scientific and scholarly text.** Commun. ACM 27, 4 (Apr. 1984) .

RINGLSTETTER, C.; SCHULZ, K. U.; MIHOV, S.. **Orthographic Errors in Web Pages:** Toward Cleaner Web Corpora. Comput. Linguist. 32, 3 (Sep. 2006), 295-340.

RINGLSTETTER, C.; SCHULZ, K. U.; MIHOV, S.. **Adaptive text correction with Web-crawled domain-dependent dictionaries**. ACM Trans. Speech Lang. Process. 4, 4 (Oct. 2007), 9.

RECUERO, R.. **Elementos para a análise da conversação na comunicação mediada pelo computador.** Verso e Reverso (São Leopoldo), v. 2008/3, p. 1-15, 2008.

ULLMANN J.R.. A binary n-gram technique for automatic correction of substitution, deletion, insertion, and reversal errors in words. **Computer Journal.** Vol. 20, Nº 2, 1977, pp.141-147.

Recebido em 10-02-10 • Aceito em 16-08-10