PRINCÍPIOS DE INTELIGÊNCIA ARTIFICIAL COM PHP

VILLANI, Leonardo, Mestre*

* Faculdade de Tecnologia de Praia Grande
Departamento de Análise e Desenvolvimento de Sistemas
Praça 19 de Janeiro, 144, Boqueirão, Praia Grande / SP, CEP: 11700-100
Fone (13) 3591-1303
leonardo.villani@fatec.sp.gov.br

RESUMO

Neste artigo são usados princípios de inteligência artificial para implementar em, PHP, um corretor ortográfico. Essa abordagem compara recursos que ditam todos os passos que um sistema precisa percorrer para alcançar uma solução. Os resultados obtidos indicam que a implementação de uma abordagem que usa princípios de inteligência artificial pode ser mais eficiente.

PALAVRAS-CHAVE: Inteligência artificial. PHP. Distância de Levenshtein.

ABSTRACT

In this paper are used artificial intelligence principles to implement in PHP an orthographic corrector. This approach is compared to another that imposes all steps that a system need make to reach a solution. The results obtained show that the approach that uses implementation of artificial intelligence principles can be more efficient.

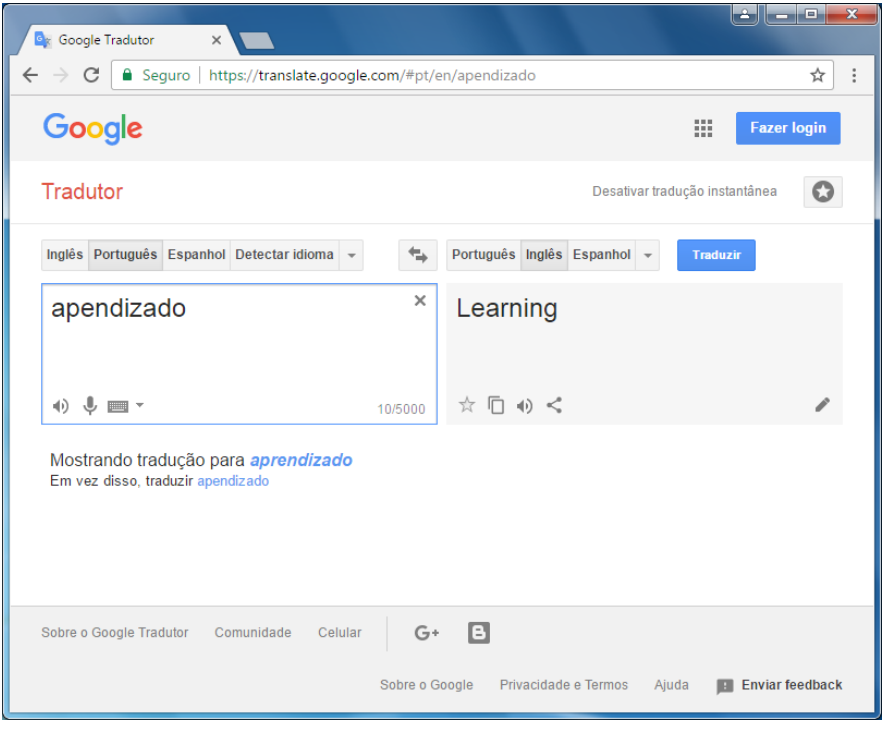
***KEYWORDS**: Artificial intelligence. PHP. Levenshtein distance.*

INTRODUÇÃO

Alguém, em algum momento, já deve ter precisado usar o serviço de tradução do Google. Dentre os diversos recursos que esse

serviço possui, um deles é o recurso ilustrado na Figura 1. Esse serviço permite que seja definido um idioma para a entrada de um texto e outro idioma para a saída do texto correspondente. À esquerda da imagem, há um campo para a entrada; à direita, é apresentada a saída correspondente para o idioma informado. Abaixo do local de entrada do texto, eventualmente, pode ser apresentado o recurso "Mostrando tradução para...". Esse recurso sugere uma alternativa mais adequada ao termo informado no campo de entrada. Trata-se de um recurso útil e inteligente para a tradução. Seria, entretanto, essa a tal inteligência artificial?

Figura 1 - Página do Tradutor do Google sugerindo correção para entrada incorreta de palavra



Fonte: adaptado pelo autor para este trabalho (2018).

Um recurso similar há em alguns editores de texto, como o Microsoft Word ou o Libre Office Writer. Quando algum texto apresenta

erro ortográfico, editores como esses acrescentam um sublinhado vermelho abaixo do texto indicando que existe erro. Acionando o menu de contexto (botão direito do mouse) sobre as palavras com esse tipo de sublinhado, uma lista com os termos que mais se aproximam dessa palavra errada é apresentada. Esse recurso é tão inteligente quanto o recurso do Tradutor do Google, mas seria esse recurso também uma inteligência artificial? Seria possível reproduzi-los apenas com os recursos de estruturas comuns de linguagens de programação? O que é necessário para implementar tais recursos com a linguagem de programação PHP?

Sendo assim, com base nos fundamentos da inteligência artificial, o presente estudo buscou analisar uma solução mais eficiente de corretor ortográfico, utilizando um sistema de sugestão mais autônoma que não exija um tratamento exaustivo de entrada de dados. Para tanto foi empregado os princípios dos algoritmos de aprendizado de máquina, com funções de distância, o denominado "Função de Distância de Levenshtein".

1 PROGRAMAÇÃO COM PHP

O PHP (Personal Home Page) é uma linguagem de programação que possibilita o desenvolvimento de programas para a Web. Um programa escrito em PHP pode conter apenas código PHP ou estar entre marcações *HyperText Markup Language* (HTML). (PREFÁCIO, 2017). O HTML é uma linguagem usada para estruturar um documento de hipertexto e não possui recursos para programação como o PHP (WHAT IS, 2017). Por isso é comum o desenvolvimento de aplicações para a Web que utilizam ambas as linguagens. No Código 1, é apresentado um trecho de código de ambas as linguagens.

Código 1 - Exemplo de código PHP

```
1. <!DOCTYPE html>
2.  <html lang="pt-BR">
3.  <head>
4.    <meta charset="UTF-8">
5.    <title>Título da página</title>
6.  </head>
7.  <body>
8.    <h1>Título principal</h1>
9.    <?php
10.   $nome = "Leonardo";
11.   print "Meu nome é $nome";
12.   ?>
13. </body>
14. </html>
```

Fonte: elaborado pelo autor para este trabalho (2018).

Somente na Linha 9 desse código há uma instrução PHP. As instruções PHP se diferem das marcações HTML por meio do sinal de interrogação seguido da palavra “php”. Quando inseridas junto às marcações HTML, as instruções PHP devem conter a marcação de fechamento que indica o fim de uma instrução PHP. As marcações de fechamento não possuem a palavra “php”. Com as marcações de abertura “<?php” e fechamento “?>” é possível indicar onde começa e onde termina uma instrução PHP. No código apresentado, o comando “print” escreve na página o texto “Script em PHP”.

Variáveis são recursos que linguagens de programação possuem para armazenar um valor e permitir que o valor armazenado seja recuperado depois. As variáveis em PHP possuem um cifrão antes do seu identificador, como na Linha 10 do Código 2.

Código 2 - Declaração de variáveis em PHP

```
1. <!DOCTYPE html>
2.  <html lang="pt-BR">
3.  <head>
4.    <meta charset="UTF-8">
5.    <title>Título da página</title>
6.  </head>
7.  <body>
8.    <h1>Título principal</h1>
9.    <?php
10.   $nome = "Leonardo";
11.   print "Meu nome é $nome";
12.   ?>
13. </body>
14. </html>
```

Fonte: elaborado pelo autor para este trabalho (2018).

Na Linha 10 do Código 2, uma variável “nome” é declarada e armazena o conteúdo “Leonardo”. O conteúdo dessa variável é recuperado na linha seguinte. Desse modo, na Linha 11 do Código 2, o comando “*print*” deverá escrever na página o texto “Meu nome é Leonardo”.

Um dos recursos que programas Web possuem para entrada de dados é o *Uniform Resource Locator* (URL), traduzido como Localizador Padrão de Recursos. Informado na barra de endereços de um navegador (eg. Chrome), um URL contém o nome do computador que tem o recurso e o nome da página. Eventualmente, também pode conter parâmetros fornecidos para a página (BERNERS-LEE, 2017), com nesse URL:

http://www.meusite.com.br/arquivo.php?**nome=Leonardo**.

Nesse URL, parâmetros são todos aqueles que aparecem após o sinal de interrogação. Um parâmetro possui um identificador e um valor atribuído a ele. Quando mais de um parâmetro necessita ser fornecido, o caractere “&” é utilizado para separar os parâmetros em um URL.

No código PHP, o vetor $_GET é usado para recuperar o

valor de um parâmetro fornecido pelo URL. Um vetor é um conjunto de variáveis. Por meio do vetor $_GET é possível recuperar os valores de todos os parâmetros fornecidos por meio do URL. Basta usar como índice desse vetor o nome usado para identificar o parâmetro informado (Código 3).

Código 3 - Obtendo valores do URL

```
1. <?php
2.  print "Valor obtido do URL: " . $_GET['nome'];
3.  ?>
```

Fonte: elaborado pelo autor para este trabalho (2018).

Na Linha 2 do Código 3, o operador "." (ponto) é usado para concatenar o texto "Valor obtido do URL" com o valor do parâmetro $_GET['nome']. Essa instrução escreve na página o texto "Valor obtido do URL: Leonardo".

Os valores do vetor $_GET são sempre definidos por meio do URL. Se precisar usar um vetor e atribuir os valores desse vetor direto no código PHP, poderá fazer uso de uma função "array". Na Linha 2 do Código 4, é criado um vetor "times" que deve possuir dois elementos: Palmeiras e Santos.

Código 4 - Declarando vetores em PHP

```
1. <?php
2.  $times = array('Palmeiras', 'Santos');
3.  print "Time 1: " . $time[0] . "<br>";
4.  print "Time 2: " . $time[1] . "<br>";
5.  ?>
```

Fonte: Elaborado pelo autor para este trabalho.

Todos os argumentos fornecidos para a função "array" tornam-se elementos de um vetor. Como visto para o vetor $_GET, cada elemento pode ser acessado por meio de um índice. No caso de um vetor criado pela função "array", todos os índices desse vetor são numéricos. O valor do índice do primeiro elemento de um vetor é zero.

2 IMPLEMENTANDO O CORRETOR ORTOGRÁFICO

Para o funcionamento do corretor ortográfico proposto, um termo deverá ser fornecido pelo URL. Caso o termo informado não exista no dicionário interno, o programa deverá sugerir a palavra do dicionário interno que é "mais próxima" do termo informado. O dicionário interno será um vetor que possui um número limitado de palavras. Nesse sentido, é proposto o código 5 a seguir:

Código 5 - Proposta inicial de corretor ortográfico

```
1. <?php
2.  $busca = $_GET['busca']; // curintia
3.  $times = array('Corinthians','Santos','...');
4.  // Tratamento da entrada
5.  if ($busca == 'curintia')
6.    $busca = 'Corinthians';
7.  elseif ($busca == 'curintians')
8.    $busca = 'Corinthians';
9.  elseif ($busca == 'Corinthias')
10.   $busca = 'Corinthians';
11. elseif (...) ...
```

Fonte: Elaborado pelo autor para este trabalho

O comando "if", usado na Linha 5 do Código 5, e os comandos "elseif", que aparecem a seguir, são comandos da linguagem PHP usados em estruturas condicionais, ou seja, quando se deseja criar condições. Na Linha 5 começaria uma tentativa incessante de tratar todos os possíveis termos que um usuário poderia querer ter informado como entrada. Um extenso código poderia ser produzido apenas para tratar a entrada "curintia". Um único termo. E para os outros termos que poderiam ser informados? E para os termos que poderiam indicar outra equipe? Existiria alguma solução alternativa que exigisse menos código? Que pudesse "entender" o que o usuário poderia querer ter informado? Uma solução mais inteligente? Inteligência Artificial?

3 O QUE É INTELIGÊNCIA ARTIFICIAL

Inteligência pode ser definida como a habilidade de "aprender" algo ou "lidar" com algo (INTELIGÊNCIA, 2017). Nesse sentido, aprender está relacionado com a aquisição de alguma informação que possa ser usada para realizar uma atividade ou melhorar a realização de uma atividade.

Com relação ao termo artificial, é atribuído àquilo que é produzido não pela natureza, mas por meio de uma técnica (ARTIFICIAL, 2017). Por exemplo, uma flor artificial que parece ser natural e pode até perfumar o ambiente como a natural, mas é criada por meio de uma técnica para imitar aquela flor natural. Como outro exemplo, pode-se citar a luz artificial que é usada para iluminar ambientes na ausência da luz natural. Dessa forma, permite que os seres humanos enxerguem ambientes, objetos, uns aos outros etc., tanto quanto a luz do Sol permite. No entanto, tal iluminação não é produzida a partir de explosões tais como as de estrelas.

Em quaisquer exemplos mencionados, o objetivo do artificial é obter os mesmos resultados dos respectivos recursos naturais. Nesse contexto, o importante é o "quê obter" e não o "como obter". Mais importante que surgir das explosões solares é a iluminação que vem delas.

Do mesmo modo, pode-se concluir que inteligência artificial é um recurso que, por meio de uma técnica, reproduz a habilidade natural de aprender algo ou lidar com algo. No entanto, dizer que computadores podem "aprender" não significa que "aprendem" da mesma maneira que os seres humanos (ARTERO, 2009).

Os fracassos dos primeiros projetos de aviões se davam porque o projeto partia do objetivo de imitar os pássaros apenas: locomover-se de um ponto a outro pelo ar, mas batendo asas (HARRINGTON; GALVEZ, 1953). Os projetos começaram a ter sucesso quando o foco passou a ser o "quê obter" (locomover-se de um ponto a outro pelo ar) e deixou de ser em "como obter" (batendo asas). Isso permitiu que se pensasse em uma maneira diferente a fim de obter o que se pretendia.

Por meio de computadores, é possível imitar certos aspectos do pensamento e comportamento considerado inteligente, realizar ações ou atingir metas, questões que estariam essencialmente relacionadas

aos humanos e requereriam inteligência a fim de se chegar a resultados satisfatórios.

Quando surgiu a inteligência artificial, tinha-se como objetivo desenvolver sistemas que fossem mais autônomos ou necessitassem de menos intervenção humana para a realização de uma atividade. De maneira mais ambiciosa, a ideia era reproduzir todas as características cognitivas do ser humano em uma máquina.

Na década de 1950, o cientista Alan Turing propôs um teste para avaliar se um sistema possuía ou não inteligência artificial. O teste era uma adaptação do "Jogo de Imitação". O jogo de imitação era jogado com três pessoas: um homem, uma mulher e um juiz. O juiz fazia perguntas para os demais jogadores, visando identificar qual deles era o homem e qual era a mulher. O papel de um dos jogadores era responder com o objetivo de confundir o juiz e o papel do outro jogador era responder com o objetivo de auxiliar.

A adaptação proposta era que, no lugar do homem e da mulher, existisse um computador/sistema que fizesse uso de inteligência artificial para responder as questões. O juiz deveria manter um diálogo com a máquina usando o seu idioma natural, o inglês por exemplo. O sistema passava no teste se o juiz não tivesse como dizer se quem estava dando as respostas era uma máquina ou um ser humano (RUSSEL; NORVIG, 2013).

Para passar nesse teste era necessário que o sistema tivesse ao menos quatro habilidades cognitivas. Uma dessas habilidades tratava-se da capacidade de processamento de linguagem natural. Tal habilidade permitiria que o juiz fizesse as perguntas e obtivesse as respostas em seu idioma nativo. Outra habilidade era a de representar de maneira eficiente o conhecimento obtido. Essa habilidade possibilitaria o armazenamento e recuperação de informações em grandes quantidades. A terceira habilidade era o raciocínio automatizado para que o conhecimento adquirido durante o diálogo pudesse ser utilizado para fornecer novas respostas. E a quarta e última habilidade, porém não menos importante, era o aprendizado de máquina. Essa habilidade permitia que o sistema pudesse responder ou melhorar a sua resposta a partir do conhecimento obtido (RUSSEL; NORVIG, 2013).

Atualmente, cada uma dessas habilidades virou grandes subáreas de pesquisa dentro da área de Inteligência Artificial.

Diversos sistemas, hoje, possuem recursos importantes empregando uma ou mais técnicas dessas subáreas. Tal como, por exemplo, o reconhecimento facial do sistema de marcações de fotos do *Facebook*, o recurso de pronúncia de um termo em outro idioma do serviço de tradução do Google, os sistemas de tele atendimento para suporte de algumas operadoras de internet e TV a cabo entre outros.

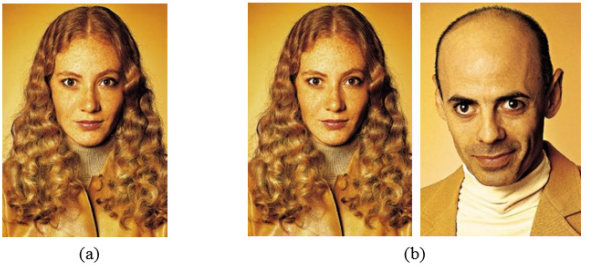
Vários estudos na busca por inserir todas as características de um ser humano em uma máquina continuam, mas importantes sistemas para o dia a dia dos seres humanos já foram desenvolvidos, inserindo apenas algumas dessas características.

4 O APRENDIZADO DE MÁQUINA

O aprendizado de máquina é uma subárea da Inteligência Artificial que visa estudar maneiras para que computadores aprendam a resolver um problema. Nesse contexto, aprender significa obter habilidade para realizar uma atividade ou melhorar o desempenho a partir de uma experiência adquirida (MICHALSKI; CARBONELL; MITCHELL, 2013).

Desse modo, o objetivo dos algoritmos de aprendizado de máquina não é "ditar" tudo que um sistema deve fazer para realizar uma tarefa, mas definir e usar parâmetros para que, com base nesses parâmetros, o sistema realize uma tarefa de modo autônomo. Um exemplo de aplicação para o tipo de abordagem que "dita" o que a máquina precisa fazer é ilustrado na Figura 2.

Figura 2 - Busca de imagens. Parâmetro da busca (a) e imagens disponíveis na base (b)



(a) (b)

Fonte: adaptado pelo autor para este trabalho (2018).

Abordagens de busca de imagens que “ditam” o que máquina deve fazer possibilitam que imagens iguais sejam recuperadas de uma base de imagens. Mas somente se as imagens, a imagem de parâmetro e a imagem recuperada, forem iguais.

Por outro lado, abordagens de busca de imagens que “dão autonomia” às máquinas podem recuperar imagens que são parecidas e não somente iguais (Figura 3).

Figura 3 - Busca de imagens semelhantes.
Parâmetro da busca (a) e imagens disponíveis na base (b)

(a) (b)

Fonte: Adaptado pelo autor para este trabalho

Alguns algoritmos de aprendizado de máquina utilizam funções de distâncias como base para sua autonomia. A seguir serão apresentadas algumas aplicações que podem fazer uso de funções de distâncias.

4.1 DISTÂNCIA EUCLIDIANA: IDENTIFICANDO MENOR DISTÂNCIA ENTRE SANTOS E PRAIA GRANDE

Existem diversas funções de distância, cada uma é considerada ideal para certo tipo de aplicação. A distância euclidiana, por exemplo, é uma função que trabalha com valores do tipo contínuo. Valores contínuos são aqueles em que se pode estabelecer uma relação de ordem. Por exemplo, se for obtido o peso de dez pessoas que estão em uma sala, será possível organizar esses valores do mais leve até o mais pesado.

Outros exemplos para valores contínuos que podem ser citados são: idade, altura e preço. A cor de cabelo, por exemplo, trata-se de um exemplo de outro tipo de valor: valor nominal. A única relação de ordem que é possível estabelecer entre valores nominais é a ordem alfabética e esse tipo de ordem não há como ser utilizado em uma função de distância como a euclidiana.

Como exemplo de aplicação da função de distância euclidiana, será descrita a questão da distância entre Santos e Praia Grande. Considera-se que a localização dessas cidades possa ser representada como pontos em um plano cartesiano (Figura 4). Para facilitar o exemplo matemático, considera-se apenas os valores do eixo X. Desse modo, Praia Grande estaria localizada no ponto 2 do eixo X e Santos no ponto 5 do mesmo eixo.

Figura 4 - Representação da localização de Santos e Praia Grande em plano cartesiano

Fonte: Adaptado pelo autor para este trabalho

Partindo de Santos para a Praia Grande, poder-se-ia considerar que a distância a ser percorrida é de 3 pontos, demonstrado na equação 1 a seguir:

Equação 1: $Distância(5, 3) = 5 - 2 = 3$

Desse modo, fazendo o caminho inverso, de Praia Grande a Santos, é necessário tomar cuidado para não cair no seguinte erro lógico resultando na distância de -3 pontos (equação 2 abaixo):

Equação 2: $Distância(3, 5) = 2 - 5 = -3$

A função da distância euclidiana cuida para que a ordem dos valores inseridos como entrada da função não influencie no resultado (Equação 3):

Equação 3: $$Euclidiana(a, b) = \sqrt{(a - b)^2} = |a - b|$$

A referida função para mais de uma dimensão (eixo) é definida como (Equação 4):

Equação 4:

$$Euclidiana(a, b) = \sqrt{(a_1 - b_1)^2 + (a_2 - b_2)^2 + \cdots + (a_n - b_n)^2} = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

No exemplo fornecido foi considerado apenas um caminho entre as duas cidades, uma reta. Em um caso real, a distância euclidiana poderia ser usada para estimar as distâncias entre todos os pontos que definem cada caminho para a cidade destino. Em seguida, o sistema poderia indicar o caminho que possuísse a menor distância.

4.2 DISTÂNCIA DE LEVENSHTEIN: IDENTIFICANDO QUÃO PARECIDAS SÃO DUAS PALAVRAS

A função de distância de Levenshtein é uma função que trabalha com valores do tipo nominal, como a cor de cabelo citado como exemplo na seção anterior. Essa função é usada, em especial,

para estimar a distância entre duas palavras. O operador de igualdade usado em expressões relacionais permite avaliar apenas se o texto de uma variável é ou não igual ao texto de outra variável.

A função de distância de Levenshtein, além de avaliar se os textos de ambas as variáveis são iguais, ela também permite estimar quão parecidos são os textos de duas variáveis.

O uso de três operações é considerado para estimar a referida distância. Tais operações são substituição, inserção e deleção de caractere. Para cada operação que tiver que ser realizada para transformar uma palavra em outra, é contabilizado o custo um (LEVENSHTEIN, 1966). Assim, a similaridade entre palavras pode ser estimada como aquela que possui o custo mais barato para transformação.

Suponha, como exemplo, a comparação entre duas variáveis. Para uma variável foi atribuído o texto "Piotr" e para a outra variável, o texto "Peter". A distância entre ambas as variáveis poderia ser calculada como:

1. P**i**otr – Início da transformação;
2. P**eo**tr – substituir "i" por "e" (custo 1). Total = 1;
3. Petr – apagar "o" (custo 1). Total = 2;
4. Pet**e**r – inserir "e" entre "t" e "r" (custo 1). Total = 3.

Para o caso da aplicação do corretor ortográfico apresentado na Seção 2, esse tipo de função parece ser mais adequado, pois bastaria retornar a palavra do dicionário que fosse mais próxima ao termo digitado, ou seja, aquela que possuir o custo mais barato de transformação.

5 IMPLEMENTANDO A SOLUÇÃO

Usando os princípios empregados em algoritmos de aprendizado de máquina, como funções de distância, propõe-se uma solução mais eficiente do que aquela iniciada na Seção 2. A distância entre as palavras pode ser usada como parâmetro para que o sistema sugira com maior autonomia uma palavra do dicionário, evitando a sequência de condições iniciada na solução proposta. Um exemplo com essa nova solução é proposto por meio do Código 6.

Código 6 - Algoritmo alternativo para a proposta do corretor ortográfico

```
1. ALGORITMO
2.    DECLARE busca, times[4], sugestao LITERAL
3.    DECLARE i, distancia, menor_d NUMÉERICO
4.    sugestao <- NULO
5.    menor_d <- NULO
6.    busca <- OBTEM PARÂMETRO DO URL
7.    times <- {"Corinthians", "Santos", "Palmeiras", "São Paulo"}
8.    PARA i <- 0 ATÉ 3 FAÇA
9.      distancia <- LEVENSHTEIN(busca, times[i])
10.     SE distancia < menor_d OU menor_d = NULO ENTÃO
11.       menor_d <- distancia
12.       sugestao <- times[i]
```

```
1. <?php
2.  $busca = null;
3.  $mais_provavel = null;
4.  if (isset($_GET['busca'])) {
5.    $busca = $_GET['busca'];
6.    $times = array('Palmeiras', 'Santos', 'São Paulo', 'Corinthians');
7.    $menor_distancia = null;
8.    foreach ($times as $time) {
9.      $distancia = levenshtein($busca, $time);
10.     if ($distancia < $menor_distancia || is_null($menor_distancia)) {
11.       $menor_distancia = $distancia;
12.       $mais_provavel = $time;
13.     }
14.   }
15. }
16. $saida = null;
17. if (is_null($busca)) $saida = 'Informe um parâmetro de busca';
18. else $saida = "Você quis dizer: $mais_provavel";
19. ?>
20. <!DOCTYPE html>
21. <html lang="pt-BR">
22.   <head>
23.     <meta charset="UTF-8">
24.     <title>Sugerindo palavra</title>
25.   </head>
26.   <body>
27.     <p><?=$saida?></p>
28.   </body>
29. </html>
```

Fonte: Adaptado pelo autor para este trabalho

Nesse exemplo também há um "acabamento" na saída apresentada usando HTML. Nesse caso, se nenhum parâmetro for informado é escrito na página 'Informe um parâmetro de busca'. A lógica do algoritmo é executada ao entrar com um parâmetro "busca" no URL.

CONSIDERAÇÕES FINAIS

Um programa de computador pode ser desenvolvido a partir da previsão de todas as possíveis entradas que esse programa pode receber. Uma ação do programa pode ser executada para cada entrada fornecida. Essa abordagem de programação dita todos os passos que o programa deve percorrer para se alcançar uma solução. No entanto, essa abordagem pode não ser a mais eficiente. Um exemplo com essa abordagem foi apresentado como solução inicial para o corretor ortográfico proposto.

A partir do uso de princípios de inteligência artificial foi possível desenvolver uma solução que fosse mais eficiente em relação à solução inicial apresentada, pois a segunda abordagem permitiu que o corretor ortográfico fornecesse sugestões com maior autonomia, não sendo necessário o tratamento exaustivo da entrada fornecida e tornando a tomada de decisão (a sugestão de palavras) baseada em pesos atribuídos a cada uma das possibilidades.

Vale ressaltar que os pesos atribuídos de maneira incorreta a cada uma das possibilidades podem resultar em sugestões equivocadas, assim como acontece quando os seres humanos precisam tomar uma decisão.

REFERÊNCIAS

ARTERO, A. O. **Inteligência artificial**: teórica e prática. São Paulo: Livraria da Física, 2009.

ARTIFICIAL. **Dicionário o*nline* de português**. Disponível em: <https://www.dicio.com.br/ artificial> Acesso em: 21 fev. 2017.

BERNERS-LEE, T. et. al. ***Uniform resource locators***. Disponível em: <http://www.w3.org/ Addressing/URL/url-spec.html>. Acesso em: 21 fev. 2017.

HARRINGTON, E. e GALVEZ, G. ***La aeronáutica.*** In: Historia de los inventos. Revista Sucesos. Espanha: 1953, n. 12, cap. 13. Disponível em: <http://www. librosmaravillosos. com/inventos/capitulo13.html>. Acesso em: 21 fev. 2017.

INTELIGÊNCIA. **Dicionário *online* de português**. Disponível em: <https://www.dicio. com. br/inteligencia>. Acesso em: 21 fev. 2017.

LEVENSHTEIN, V. I. ***Binary codes capable of correcting deletions, insertions, and reversals***. Soviet Physics Doklady, New York, v.10, n 8, p. 707-710, 1966.

LEVENSHTEIN. **Manual do PHP**. Disponível em: <http://php.net/ manual/pt_BR/ function.levenshtein.php>. Acesso em: 21 fev. 2017.

MICHALSKI, R. S.; CARBONELL, J. G.; MITCHELL, T. M. *Machine learning: An artificial intelligence approach.* Springer Science and Business Media, 2013.

RUSSEL, S. e NORVIG, P. **Inteligência artificial**. 3 ed. Rio de Janeiro: Elsevier, 2013.

WHAT IS HTML. **HTML & CSS**. Disponível em: <http://www.w3.org/ standards/ webdesign/htmlcss >. Acesso em: 21 fev. 2017.