# UNIDADE DE APRENDIZAGEM: A base do processamento de textos

## Apresentação

O processamento de linguagem natural, ou PLN, é amplamente definido como a manipulação automática de linguagem natural, como fala e texto, por meio de *software*. Em outras palavras, o PLN tem como objetivo fazer com que as máquinas entendam a linguagem natural dos seres humanos. O estudo do processamento de linguagem natural existe há mais de 70 anos e cresceu fora do campo da linguística com o surgimento dos computadores. Por exemplo, é possível usar PLN para criar sistemas como reconhecimento de fala, resumo de documentos, tradução automática, detecção de *spam*, sistemas de resposta a perguntas, e assim por diante.

Para facilitar a leitura e o entendimento de texto pelas máquinas, surgiu o conceito de pré-processamento de texto, que, em na sua essência, ajuda a deixar determinado documento de texto mais fácil de ser entendido por uma máquina. Nesse contexto, é importante que você consiga entender a base do processamento de texto, pois ela irá ajudá-lo a explorar de maneira mais adequada esse enorme mundo que é o PLN.

Nesta Unidade de Aprendizagem, você vai entender métodos de processamento de linguagem natural, vai conhecer processos de pré-processamento de texto e vai desenvolver soluções de PLN utilizando ferramentas que facilitam esse processo.

Bons estudos.

**Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:**

- Identificar métodos de processamento de linguagem natural.
- Descrever processos de pré-processamento de textos.
- Desenvolver soluções de processamento de linguagem natural empregando ferramentas.

# Desafio

Para se sustentarem e permanecerem no topo do mercado e proporcionarem conforto absoluto aos consumidores, as organizações empresariais estão usando diferentes estratégias e tecnologias. O PLN é uma dessas tecnologias que atinge profundamente e amplamente o mercado, independentemente da indústria e dos domínios. Hoje, é muito aplicado nas empresas e é a palavra da moda na vida de profissionais que trabalham com Inteligência Artificial (IA). Em resumo, o PLN está em todo lugar.

Nesse contexto, considerando que existe um grande número de problemas que o PLN pode ajudar a resolver, veja a situação a seguir:

Imagine que você tenha sido contratado por uma escola particular para implementar um sistema de bate-papo que detecta se as conversas dos alunos no horário da aula fogem do assunto escolar.

O diretor dessa escola estava tendo muitos problemas para controlar as conversas dos alunos por meio de redes sociais. Assim, ele teve a ideia de criar um sistema de bate-papo escolar no qual só poderão ser conversados assuntos pertinentes à escola e às disciplinas. Estima-se que a escola tenha em torno de 700 alunos em cada turno: manhã e tarde.

No entanto, percebeu-se o seguinte:

**I -** Como os alunos são crianças e adolescentes, os alunos conversam muito por meio de abreviações e gírias.

**II -** As crianças, por terem vocabulário um pouco menos diversificado, geralmente repetem muito as mesmas palavras, como, por exemplo: "de", "da", "que", "ao", "é", entre outras.

De acordo com os dados fornecidos, responda às seguintes questões:

a) No contexto do bate-papo escolar, é necessário utilizar pré-processamento de texto nessa aplicação? Justifique sua resposta.

b) Como você poderia tratar o desafio relacionado à manutenção da privacidade dos alunos?

# Infográfico

Os computadores são ótimos em trabalhar com dados estruturados, como planilhas e tabelas de banco de dados. No entanto, humanos geralmente se comunicam por palavras, e não por tabelas. Assim, faz-se necessário que computadores e máquinas entendam a linguagem natural do ser humano.

Muitas informações no mundo não são estruturadas, ou seja, existe muito texto bruto em português ou em outro idioma humano. Como se pode fazer um computador entender o texto não estruturado e extrair dados dele? Em outras palavras, como fazer o computador interpretar determinado texto? Umas das atividades necessárias para melhor interpretação de texto pela máquina é realizar o pré-processamento de texto, pois ele auxilia no entendimento da linguagem natural por máquinas, seja por meio de texto, seja por meio da fala (pois a fala é convertida em texto). Nesse contexto, é fundamental entender como funciona o processo de "limpeza" do texto bruto para um texto que possa ser entendido pela máquina.

Neste Infográfico, você vai ver e entender as formas de transformar texto bruto em informação relevante para a máquina.

# PROCESSO DE TRANSFORMAÇÃO DE TEXTO BRUTO EM INFORMAÇÃO RELEVANTE

Diariamente, milhares de textos são gerados na Internet, por meio de redes sociais, *blogs*, *sites*, entre outros meios. Por exemplo, o buscador Google precisa "entender" milhares de páginas até encontrar resultados relevantes para uma pesquisa feita. Mas como é feito processo de transformação de texto bruto em informação relevante?

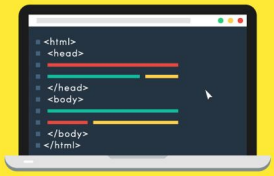
Podem ser aplicados vários processos e métodos no texto bruto com abreviações e gírias para deixá-lo mais fácil de ser entendido por uma máquina:

## REMOÇÃO DE *STOP-WORDS*

Remoção de palavras muito frequentes e insignificantes, como "de", "não", "é","o", "a", etc. Isso ajuda a diminuir o tamanho do texto sem perder a informação relevante.

## REMOÇÃO DE CÓDIGO HTML/CSS

Remoção de código Web, ou seja, código que não vai acrescentar ao entendimento do conteúdo de um texto.

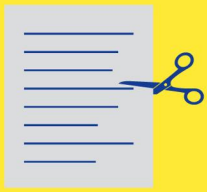



## TOKENIZAÇÃO DE PALAVRAS

Separação de palavras em unidades para facilitar o entendimento da máquina.

## SEGMENTAÇÃO DE SENTENÇAS

Útil para separar grandes parágrafos ou grandes textos em frases, além de ajudar na facilitação do entendimento da máquina.

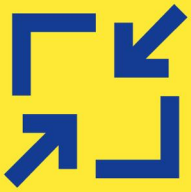


## NORMALIZAÇÃO DAS PALAVRAS

Normalizar gírias e abreviações para palavras na forma padrão. Dessa maneira, fica mais fácil o entendimento de um texto por uma máquina.

## LEMATIZAÇÃO E *STEMMING*

O tamanho de um texto será diminuído, e o vocabulário será reduzido. Isso deixa o texto mais simples.



Todos esses processos têm um objetivo em comum: deixar mais fácil o entendimento de texto bruto pela máquina, transformando esse texto em informação relevante que será entendida e interpretada.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

# Conteúdo do Livro

O PLN refere-se ao método de IA de comunicação com sistemas inteligentes usando um idioma natural, como o português. O PLN é necessário quando se deseja que um sistema inteligente, como um robô, execute algo conforme as instruções. *Chatbots* e assistentes virtuais são exemplos de aplicações que utilizam conceitos de PLN. Por exemplo, imagine um assistente virtual para clientes de determinada empresa de produtos que realiza o atendimento de questões simples, como verificar débito, consultar data de pagamento, solicitar cartão de crédito, entre outras. Esse assistente virtual deverá entender, no momento do atendimento, qual o tipo de solicitação do cliente e, assim, responder de forma precisa.

O processamento de texto permite que aplicações como a de *chatbots* e de assistentes virtuais funcionem de maneira precisa e correta, pois geralmente o canal de comunicação dessas aplições se dá por meio de texto. Dessa forma, é necessário realizar algumas atividades, como pré-processamento de texto, para permitir que a máquina entenda a linguagem natural do ser humano. Nesse contexto, é possível perceber a importância do processamento de texto na área de PLN.

No capítulo A base do processamento de textos, da obra *Processamento de linguagem natural*, base teórica desta Unidade de Aprendizagem, você vai conhecer e aprender a aplicar os principais métodos de PLN, vai estudar os processos responsáveis pelo pré-processamento de textos e vai construir soluções de PLN utilizando as principais ferramentas disponíveis para o desenvolvimento de aplicação.

Boa leitura.

# PROCESSAMENTO DE LINGUAGEM NATURAL

Júlio Serafim Martins

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

# A base do processamento de textos

## Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Identificar métodos de processamento de linguagem natural.
- Descrever processos de pré-processamento de textos.
- Desenvolver soluções de processamento de linguagem natural empregando ferramentas.

## Introdução

Um dos objetivos do processamento de linguagem natural (PLN), uma subárea da inteligência artificial e voltada à maneira como as máquinas e computadores podem interpretar e responder utilizando a linguagem natural, consiste em fazer a máquina entender a linguagem dos seres humanos.

Nesse contexto, o processamento de texto é tradicionalmente um importante passo para a execução de atividades de PLN, transformando o texto bruto em um formato mais fácil para que as máquinas consigam entender, interpretar e trabalhar da melhor maneira possível.

Neste capítulo, você verá métodos de PLN, conhecerá processos de pré-processamento de textos e entenderá como são desenvolvidas soluções de PNL utilizando ferramentas bem conhecidas no mercado de trabalho e na comunidade científica.

# 1 Métodos de processamento de linguagem natural

Segundo Indurkhya e Damerau (2010), na análise linguística de um texto digital de linguagem natural, ou seja, de algum idioma falado pelo ser humano, é necessário definir claramente os caracteres, as palavras e as frases em qualquer documento, tarefa que apresenta desafios diferentes, conforme o idioma processado e a fonte dos documentos, especialmente quando se considera a variedade de idiomas humanos e sistemas de escrita. As línguas naturais contêm ambiguidades inerentes, geralmente amplificadas pelos sistemas de escrita, além de promoverem ambiguidades adicionais. Os autores apontam que o pré-processamento de texto constitui uma parte essencial de qualquer sistema que utiliza o processamento de linguagem natural (PLN), pois os caracteres, as palavras e as frases identificados nessa etapa são as unidades fundamentais para todas as etapas de processamento e de entendimento da máquina. Dessa maneira, o PLN representa uma área que se preocupa com o entendimento das máquinas da linguagem natural do ser humano, como o buscador Google, já que este, quando usuário digita o texto de pesquisa, deve "entender" o que precisa ser buscado para trazer os resultados mais relevantes. Nesse contexto, é possível perceber a importância do processamento de texto na área de PLN.

Jurafsky e Martin (2009) afirmam que, antes de praticamente qualquer processamento de texto em linguagem natural, o texto deve ser normalizado. Mas o que é e para que serve a normalização de um texto? Esse processo consiste em um conjunto de tarefas realizadas para facilitar o entendimento de determinado texto para a máquina, por exemplo, quando da possibilidade de remover caracteres especiais de um texto. Segundo os autores, pelo menos três tarefas são comumente aplicadas como parte de qualquer processo de normalização: **tokenização de palavras**, **normalização do formato das palavras** e **segmentação de sentenças**.

De acordo com Sun *et al.* (2014), o processo de tokenização de palavras busca separá-las em *tokens*, devendo-se remover elementos que não ajudam na semântica do texto. Refere-se, portanto, ao problema de dividir um fragmento de texto de determinada linguagem em palavras, ou seja, tem como finalidade separar as palavras em unidades.

Por exemplo, imagine o seguinte texto:

Texto 1 — "Eu gosto de estudar na minha casa"

Como é feito o processo de tokenização de palavras no Texto 1? Esse texto pode ser representado da seguinte maneira:

['Eu', 'gosto', 'de', 'estudar', 'na', 'minha', 'casa'].

Perceba que as palavras do Texto 1 foram separadas em unidades, facilitando o seu entendimento pelas máquinas. Outro exemplo seria:

Texto 2 — "Processamento de Linguagem Natural é uma disciplina bem interessante"

Como é possível fazer a tokenização de palavras do Texto 2? Novamente, esse texto será separado em *tokens*:

['Processamento', 'de', 'Linguagem', 'Natural', 'é', 'uma', 'disciplina', 'bem', 'interessante']

Assim, a tokenização de palavras faz a marcação de cada palavra como se fosse um *token* no texto.

A normalização de palavras consiste na tarefa de colocar palavras ou *tokens* em um formato-padrão (JURAFSKY;MARTIN, 2009), por exemplo, as palavras "beeeeem" e "bemmmm" podem ser normalizadas para a palavra "bem". Esse processo é importante em casos de textos com muitas gírias e abreviações, como comentários em mídias sociais, mensagens de texto e escrita de textos informais. Por exemplo:

Texto 3 — "Olá td beeeem? Quais são as 9vidades? Vc tem se divertido mt?"

O Texto 3 é um exemplo claro de escrita abreviada e utilização de gírias da internet. No quadro a seguir, podemos perceber algumas palavras do texto e suas respectivas normalizações.

| Palavra antes do processo de normalização | Palavra após o processo de normalização |
|---|---|
| td | tudo |
| beeeem | bem |
| 9vidades | novidades |
| Vc | Você |
| mt | muito |

Como seria precisar ensinar a uma máquina a ler e entender todas as gírias e abreviações? Seria um trabalho bem complicado, não acha? Nesse sentido, a normalização constitui um aspecto importante no contexto de processamento de texto e na área de PLN. Um de seus grandes desafios é conseguir mapear determinada palavra abreviada ou determinar a correspondência para uma gíria na forma-padrão, visto o surgimento crescente de gírias e abreviações na internet, o que dificulta esse mapeamento.

A segmentação de sentenças corresponde ao problema de dividir um fragmento de texto com um parágrafo em componentes de frases, uma ideia que parece muito simples. Em português e alguns outros idiomas, podemos dividir as frases sempre que aparecer um sinal de pontuação, como no seguinte exemplo:

> Texto 4 — "O futebol é um dos esportes mais praticados no Brasil. No Brasil o número de campos de futebol vem diminuindo com o passar do tempo. O futebol brasileiro ainda é considerado o melhor futebol do mundo."

Aplicando o processo de segmentação, as sentenças desse texto ficarão da seguinte forma:

> Sentença 1 — "O futebol é um dos esportes mais praticados no Brasil"
> Sentença 2 — No Brasil o número de campos de futebol vem diminuindo com o passar do tempo"
> Sentença 3 — "O futebol brasileiro ainda é considerado o melhor futebol do mundo"

O Texto 4 foi separado em três sentenças a partir do sinal de pontuação (ponto final).

A segmentação de sentenças representa outro passo importante no processamento de textos, sobretudo a partir da pontuação, como pontos final, de interrogação e de exclamação — os dois últimos, inclusive, geralmente marcam o final de uma frase e não são ambíguos (JURAFSKY; MARTIN, 2009).

Conforme Jurafsky e Martin (2009), um dos grandes problemas da segmentação de sentenças refere-se à ambiguidade do caractere '.', pois ele pode representar o ponto final ou um marcador de abreviações. Imagine os seguintes textos:

> Texto 5 — "É importante praticar atividade física. Beber água também é importante."
> Texto 6 — "O aluno tirou '8.7' em matemática."

No Texto 5, é possível perceber com facilidade a separação em dois segmentos pela observação do ponto final ('.'). O problema reside no fato de que, no Texto 6, o elemento "8.7" tem o caractere '.', que, nessa situação, não tem o objetivo de definir o final de uma frase, e sim ajudar na representação de um número. Então, no Texto 6, se fizermos a segmentação até encontrar a primeira ocorrência do caractere '.', acontecerá o problema conhecido como incorreta previsão de final da frase. Mas o que seria correto fazer nessa situação? Separar esse texto em uma sentença pela ocorrência do ponto final depois da palavra "matemática". E, na prática, como isso é tratado? Em muitos casos, podemos utilizar ferramentas e bibliotecas que detectam esse tipo de problema, como descrito a seguir.

## Ferramentas e bibliotecas que ajudam no processamento de linguagem natural

Uma das linguagens mais utilizadas nos contextos de PLN e processamento de texto é a Python, uma linguagem de programação interpretada, orientada a objetos e de alto nível. A sintaxe simples e de fácil aprendizado da linguagem Python enfatiza sua legibilidade, reduzindo, assim, o custo de manutenção do programa. Ainda, suporta módulos e pacotes, o que incentiva a modularidade do programa e a reutilização de código.

**Saiba mais**

Para aprender mais sobre a linguagem Python e sobre a sua documentação, é possível acessar o *site* do programa em português.

Nesse contexto, foi criada em 2001 a Natural Language Toolkit (NLTK), como parte de um curso de linguística computacional no Departamento de Ciência da Computação e Informação da Universidade da Pensilvânia (BIRD; KLEIN; LOPER, 2009). Desde então, foi desenvolvida e ampliada com a ajuda de dezenas de colaboradores, adotada em cursos em dezenas de universidades e servido de base para muitos projetos de pesquisa. Esse tipo de linguagem NLTK foi projetado para ser oferecido como uma plataforma intuitiva, oferecendo aos usuários uma ferramenta prática para trabalhar com PLN e processamento de texto.

Dessa maneira, o NLTK representa uma ferramenta utilizada por diversos profissionais da área de PLN, com a qual se pode escrever programas em Python capazes de, por exemplo, fazer tokenização de palavras, calcular a similaridade de dois textos, verificar os conjuntos de sinônimos de uma palavra, entre outras atividades relacionadas ao processamento de texto.

**Saiba mais**

Para aprender mais sobre NLTK e acessar sua documentação, visite o *site* oficial.

Outra ferramenta interessante para trabalhar com PLN com Python é a spaCy, uma biblioteca de código aberto gratuita para processamento avançado. Ao trabalharmos com uma quantidade enorme de texto, eventualmente desejaremos "saber mais" a seu respeito. Por exemplo: sobre o que é? O que as palavras significam no contexto? Quem está fazendo o que com quem? Quais empresas e produtos são mencionados? Quais textos são similares entre si? Nesse contexto, a spaCy foi projetada especificamente para ser usada em produção e ajudar a criar aplicativos que processam e "compreendem"

grandes volumes de texto, podendo ser usada para criar sistemas de extração de informações ou de compreensão de linguagem natural, bem como para pré-processar texto. Os principais desenvolvedores da biblioteca, publicada sob a licença do Massachusetts Institute of Technology (MIT), são Matthew Honnibal e Ines Montani, os fundadores da empresa de *software* Explosion.

**Saiba mais**

Para aprender mais sobre a biblioteca spaCy e acessar sua documentação, visite o *site* oficial.

Tanto a NLTK quanto a spaCy são excelentes ferramentas para trabalhar na área de PLN, dispondo e suportando exemplos em português.

**Saiba mais**

Para visualizar e aprender utilizando exemplos na língua portuguesa das ferramentas apresentadas aqui, busque na internet: "Examples for Portuguese Processing NLTK" e "Portuguese ·spaCy Models Documentation".

Outra ferramenta capaz de auxiliar no desenvolvimento de aplicações utilizando Python é a Google Colab, um serviço gratuito na nuvem que pode rodar suas aplicações escritas em Python, possibilitando melhorar as habilidades de programação com essa linguagem, desenvolver aplicações de *deep learning* (aprendizado profundo) usando bibliotecas populares, como Keras, TensorFlow e PyTorch, e utilizar bibliotecas da área de PLN, como a NLTK. A funcionalidade mais importante que distingue o Google Colab de outros serviços na nuvem refere-se ao fato de essa plataforma dispor de uma unidade de processamento gráfico totalmente de forma gratuita, acessada por um navegador comum, como o Google Chrome.

### Saiba mais

Para mais informações sobre a plataforma Google Colab, busque na internet "What is Colaboratory?".

Para rodar o código Python no Google Colab, é necessário criar um arquivo para execução de código. Com o arquivo criado, você pode nomeá-lo e executar seus próprios códigos, como exemplificado na Figura 1.

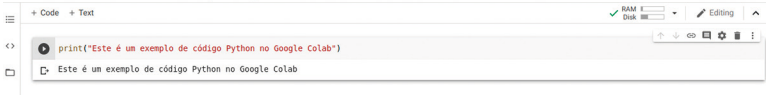



**Figura 1.** Código de exemplo na plataforma Google Colab.

Primeiro, escreve-se o código na parte cinza — o editor de código —, depois clica-se no botão em forma de "play" no lado esquerdo do editor de código, e, por fim, consegue-se visualizar o resultado do programa na parte inferior (parte branca), como mostra a Figura 1.

É visível a quantidade de boas ferramentas que auxiliam na área de PLN, processamento de texto e suas aplicações, as quais você precisa conhecer e se familiarizar, pois que farão parte do seu cotidiano como profissional dessa área que só tende a crescer.

## Abertura de arquivos de texto em Python

No Python, não há necessidade de importar biblioteca externa para ler e gravar arquivos, já que o programa fornece uma função embutida para criar, escrever e ler arquivos. Em Python, a sintaxe é relativamente simples para ler e imprimir o texto de um arquivo, como exemplificado na Figura 2.

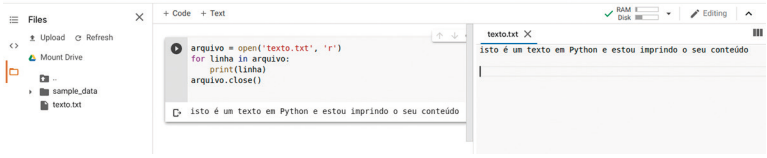



**Figura 2.** Abertura de arquivo em Python.

No lado esquerdo, é possível perceber com o arquivo `texto.txt` que um arquivo que foi upado na plataforma Google Colab, e, no canto direito, visualizar o conteúdo desse arquivo (o texto do arquivo). No meio, localiza-se o código para promover a leitura do arquivo `texto.txt`. Para isso, torna-se necessário trazer o método `open()`, que retorna o objeto do arquivo para ser manipulável. Esse método recebe dois parâmetros: o nome do arquivo e a maneira de abri-lo (por leitura ou por escrita). Ao utilizarmos a permissão de escrita, podemos modificar o conteúdo de um arquivo de texto (Figura 3).

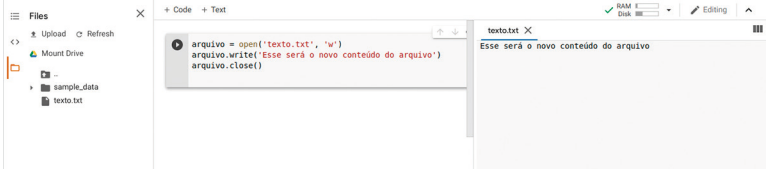



**Figura 3.** Escrita de arquivo em Python.

Em vez de colocarmos a letrar `r` ("*read*", que significa ler em português) no segundo parâmetro, é necessário colocar a letra `w` ("*write*", escrever em português) — assim, o objeto do arquivo será enviado para a variável "arquivo". Nesse momento, podemos chamar o método `write()`, que recebe o conteúdo que será inserido no arquivo. Por fim, é chamado o método `close()`, que fecha o arquivo aberto, representando sempre uma boa prática de fechar arquivos, tanto para ler quanto escrever.

Ao executar esse código, podemos perceber que o conteúdo do arquivo foi modificado para o texto definido no método `write()`.

Um dos grandes desafios ao trabalharmos com abertura de arquivos de texto em Python reside no tamanho desses arquivos: quanto maior o arquivo, maior a complexidade em trabalhar com ele. Porém, a linguagem Python tem uma linguagem bem simples e fornece maneiras intuitivas de trabalhar com os arquivos.

**Saiba mais**

Para saber mais sobre como manipular arquivos em Python, busque na internet "Refinando a formatação de saída".

## POS-Tagging no NLTK

Conforme Jurafsky e Martin (2009), as partes do discurso (também conhecidas como POS-Tagging) são úteis porque revelam muito sobre uma palavra e suas palavras vizinhas: saber se uma palavra é um substantivo ou um verbo nos ajuda a compreender sobre possíveis palavras vizinhas (os substantivos são precedidos adjetivos, verbos por substantivos) e a estrutura sintática (os substantivos geralmente fazem parte das frases substantivas), facilitando a análise (Figura 4).


**Figura 4.** POS-Tagging utilizando NLTK.

A Figura 4 mostra um exemplo de POS-Tagging empregando a plataforma NTLK com um texto em língua portuguesa, já que "MAC-MORPHO" apresenta trechos de notícias do jornal *Folha de S.Paulo*. O método `tagged_words()` mostra palavras e a sua classe gramatical, por exemplo, as palavras "Jersei" e "média" são representadas pela letra `N`, que significa substantivo, e palavra "atinge" pela letra `V`, um verbo.

Nesse contexto, podemos observar o poder da ferramenta NLTK, que dispõe de um conjunto de métodos nativos para trabalhar com PLN e processamento de texto.

## 2 Processos de pré-processamento de texto

Haddi, Liu e Shi (2013) afirmam que pré-processar o texto consiste no processo de limpeza e preparação do texto para classificação. Os textos *on-line* geralmente contêm muita "sujeira" e partes pouco informativas, como *tags* HTML, *scripts* e anúncios. Além disso, no nível das palavras, muitas palavras no texto não impactam a orientação geral do texto, e mantê-las dificulta tanto o seu entendimento quanto a sua classificação. Assim, uma das soluções para realizar um correto pré-processamento de texto consiste em reduzir a "sujeira" no texto, para ajudar a melhorar o desempenho do classificador e acelerar o processo de classificação, auxiliando na análise de sentimentos em tempo real. Todo processo de pré-processamento de texto apresenta passos-chave — limpeza de caracteres especiais e código HTML/CSS/JS, retirada de *stop-words*, *stemming*, normalização e lematização —, alguns dos quais detalhados a seguir.

## Remoção de *stop-words*

Na computação, *stop-words* são palavras filtradas antes ou depois do processamento de dados em linguagem natural (texto), geralmente as mais comuns em um idioma. Segundo Jurafsky e Martin (2009), as *stop-words* mais frequentes são: 'a', 'o', 'um' e 'uma'.

Uma das principais tarefas no contexto de pré-processamento de textos refere-se à retirada de *stop-words*, mas ela é realizada? Esse método consiste na remoção de palavras muito frequentes, como "de", "da", "a", "o", "que", "e", "do", etc., pois, na maior parte das vezes, não compreendem informações muito relevantes para o entendimento do texto. Aqui, vale ressaltar que a remoção de *stop-words* somente se dá quando essas palavras não forem relevantes para o entendimento do texto, por exemplo, no contexto de análise de sentimentos, em que se verifica se o "sentimento" sobre um assunto é positivo ou negativo, não é uma boa ideia remover a *stop-word* "não", pois traz uma indicação de negatividade para a frase, apontando justamente o sentimento transmitido.

Existem várias listas de *stop-words* da língua portuguesa disponíveis na internet.

**Saiba mais**

Para acessar duas listas de *stop-words* da língua portuguesa, busque na internet "alopes/stopwords.txt" e "Lista de StopWords Virtuati".

## *Stemming* e lematização

Segundo Teixeira e Brum (2018), há uma diferença bem sutil entre os conceitos de *stemming* e lematização. No primeiro, promove-se um processo de reduzir a palavra para o seu radical, como as palavras "meninas" e "menino", que se reduziriam ao radical "menin". Outro exemplo pode ser visualizado na Figura 5.

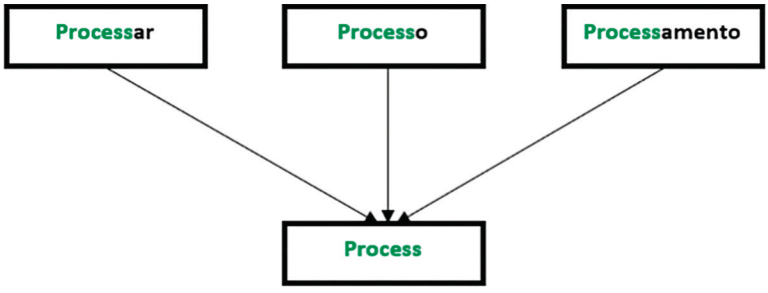



**Figura 5.** Processo de *stemming* .
***Fonte:*** Oliveira (2019, p. 22).

Como observamos na Figura 5, foi aplicado o processo de *stemming* para as palavras "Processar", "Processo" e "Processamento", que apresentam o radical "Process" em comum.

Já a lematização (Figura 6) se preocupa com o lema, reduzindo a palavra para sua origem básica, a forma no masculino e no singular. Por exemplo, as palavras "gato", "gatas", "gata" e "gatos" são reduzidas ao lema "gato". A mesma coisa acontece para verbos, diferenciando pelo fato de que, nesse caso, o lema será o infinitivo, por exemplo, os verbos "responde" e "respondendo" são formas do mesmo lema: "responder".


**Figura 6.** Processo de lematização.
***Fonte:*** Oliveira (2019, p. 23).

Na lematização, o radical "Process" e todas as palavras relacionadas a esse radical tem poderiam assumir a forma infinitiva do verbo "Processar".

Mas qual é a vantagem de aplicar *stemming* e lematização? A vantagem reside na possibilidade de reduzir bastante o vocabulário de um texto e abstrair o significado de forma mais fácil.

# 3 Desenvolvimento de solução de PLN utilizando ferramentas

Como já dito, a NLTK representa uma das ferramentas mais utilizadas no desenvolvimento de aplicações e soluções na área de PLN, tendo sido construída para ser simples, consistente, extensível e modular (BIRD; KLEIN; LOPER, 2009).

Por exemplo, como é feito o processo de tokenização utilizando a NLTK? A Figura 7 mostra um exemplo desse tipo de solução.

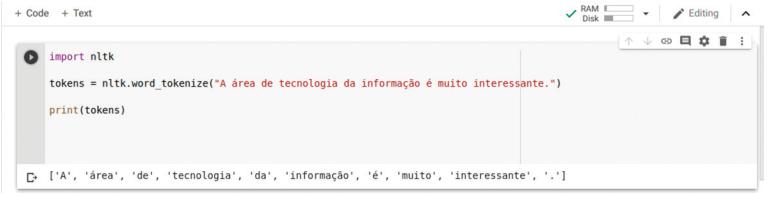



**Figura 7.** Tokenização de texto utilizando Python e NLTK.

O texto para tokenizar foi: "A área de tecnologia da informação é muito interessante". O console mostra o resultado da impressão dos *tokens* dessa frase. Como podemos observar, foi utilizado o método `word_tokenize()` da ferramenta NLTK, que recebe um texto como parâmetro e faz a sua tokenização.

Com Python e NLTK, também podemos fazer a segmentação de sentenças, como mostrado no código da Figura 8.

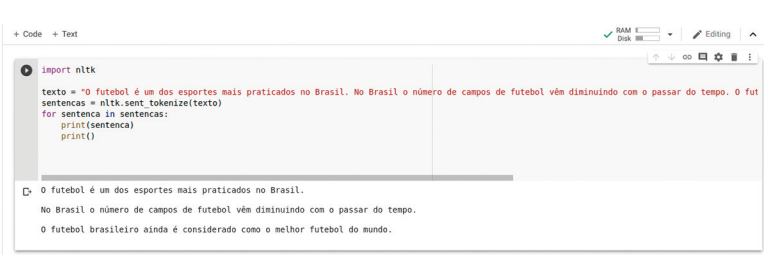



**Figura 8.** Segmentação de sentenças.

A Figura 8 mostra a segmentação de sentenças do Texto 4 exibido anteriormente: criou-se uma variável chamada de "texto", que recebeu o conteúdo do Texto 4; depois, chamou-se o método `sent_tokenize()`, que recebeu a variável texto como parâmetro e retornou uma lista de sentenças segmentadas; por fim, fez-se o percurso imprimindo todas as sentenças segmentadas.

Também podemos imprimir todas as *stop-words* de determinado idioma. Na Figura 9, há uma lista de *stop-words* da língua portuguesa.


**Figura 9.** Listagem de *stop-words* da língua portuguesa.

A partir do *download* do pacote `stopwords` e da utilização do método `stopwords.words()`, é possível retornar as *stop-words* de determinado idioma. No caso da Figura 9, foi passado como parâmetro a palavra `portuguese`, ou seja, são impressas todas as *stop-words* da língua portuguesa de acordo com o pacote *stopwords* da NLTK.

E, se a partir de um texto, fosse possível remover as suas *stop-words*? Também podemos fazer isso com Python e NLTK, como exemplificado na Figura 10.

A frase na qual se removerão *stop-words* é: "Ele é de Curitiba". Nessa frase, é possível identificar duas *stop-words*: 'é' e 'de'. Então, no código da Figura 10, primeiro são colocadas na variável `stop_words` todas as *stop-words* da língua portuguesa. Depois, faz-se a tokenização da frase e, depois, idealiza-se uma estrutura de repetição percorrendo os *tokens* da frase "perguntando": "Essa palavra é uma *stop-word*?". Caso não se trate de uma *stop-word*, essa palavra/*token* é colocada na lista `texto_sem_stop_words`. No final da execução do programa, são impressos os *tokens* não *stop-words*.


**Figura 10.** Remoção de *stop-words* de um texto.

Em resumo, podemos perceber o poder de ferramentas como a NLTK para trabalhar na área de PLN, o que mostra a importância de conhecê-las para garantir cada vez mais chances de obter sucesso profissional.

## Referências

BIRD, S.; KLEIN, E.; LOPER, E. *Natural language processing with Python*: analyzing text with the natural language toolkit. 2009. Disponível em: https://www.nltk.org/book/. Acesso em: 04 maio 2020.

HADDI, E.; LIU, X.; SHI, Y. The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, [*s. l.*], v. 17, p. 26–32, 2013.

INDURKHYA, N.; DAMERAU, F. J. *Handbook of natural language processing*. 2nd ed. Boca Raton: CRC, 2010.

JURAFSKY, D.; MARTIN, J. H. *Speech and language processing*: an introduction to natural language processing, computational linguistics, and speech recognition. 3rd ed. 2009. Disponível em: https://web.stanford.edu/~jurafsky/slp3/ed3book.pdf. Acesso em: 04 maio 2020.

OLIVEIRA, D. C. *Aplicação das técnicas de processamento de linguagem natural Cosine Similarity e Word Mover's Distance na automatização da correção de questões discursivas no sistema tutor inteligente Mazk*. 2019. Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação) – Centro de Ciências, Tecnologias e Saúde, Universidade Federal de Santa Catarina, Araranguá, 2019. Disponível em: https://repositorio.ufsc.br/bitstream/handle/123456789/203096/TCC_Douglas_Camilo_de_Oliveira.pdf?sequence=1&isAllowed=y. Acesso em: 04 maio 2020.

SUN, X. *et al*. Empirical studies on the NLP techniques for source code data preprocessing. *In*: INTERNATIONAL WORKSHOP ON EVIDENTIAL ASSESSMENT OF SOFTWARE TECHNOLOGIES, 3., 2014, [*s. l.*]. *Proceedings* [. . .]. [*S. l.: s. n.*], 2014.

TEIXEIRA, G. M.; BRUM, B. F. *ITeligence*: sistema de apoio à análise de intenções. 2018. Trabalho de Conclusão de Curso (Bacharelado em Sistemas da Informação) – Escola de Informática Aplicada, Centro de Ciências Exatas e Tecnologia, Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro, 2018.

## Fique atento

Os *links* para *sites* da *web* fornecidos neste capítulo foram todos testados, e seu funcionamento foi comprovado no momento da publicação do material. No entanto, a rede é extremamente dinâmica; suas páginas estão constantemente mudando de local e conteúdo. Assim, os editores declaram não ter qualquer responsabilidade sobre qualidade, precisão ou integralidade das informações referidas em tais *links*.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

SAGAH
SOLUÇÕES EDUCACIONAIS INTEGRADAS

# Dica do Professor

O processamento de texto é o processo de analisar e manipular informações textuais. Isso inclui extrair pequenas informações do texto (também conhecido como *extração de texto*), atribuir valores ou *tags* dependendo do seu conteúdo (também conhecido como *classificação de texto*) ou executar cálculos que dependem das informações textuais.

Processamento de texto é uma das atividades mais comuns na área de PLN. Nesse contexto, empresas se beneficiam diariamente do processamento de texto por meio do entendimento das necessidades dos clientes para recomendar novos produtos e das avaliações do clientes para melhorar produtos e serviços.

Nesta Dica do Professor, você irá conhecer o papel do processamento de textos e seu impacto para o mundo dos negócios.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

# Exercícios

1) **O processamento de textos é uma das atividades essenciais no contexto de PLN. É por meio dele que a máquina obtém textos mais padronizados e mais fáceis de ser entendidos. Uma das atividades de pré-processamento de texto consiste em separar frases após a ocorrência de sinais de pontuação. Assinale a alternativa que representa a atividade de pré-processamento descrita:**

A) Segmentação de sentenças.

B) Normalização de palavras.

C) Análise de sentimento.

D) Geração de tokenização.

E) Remoção de código HTML/CSS.

2) **Quase todo o processo de pré-processamento de texto consiste em passos como eliminação de *stop-words*, *stemming*, normalização e lematização. Qual passo se refere à remoção de palavras muito frequentes e que, na maioria das vezes, não são informações relevantes para o texto?**

A) Google Colab.

B) NLTK.

C) Normalização de palavras.

D) Eliminação de *stop-words*.

E) Lematização.

3) **A tokenização de palavras divide um fragmento de texto em palavras, ou seja, tem como finalidade separar as palavras em unidades.**

**Veja a seguinte frase:**

**"Este é um aventureiro de São Paulo.".**

**Assinale a alternativa que apresenta a correta tokenização dessa frase.**

**A)** ['Este', 'é', 'um', 'aventureiro', 'de', 'São', 'Paulo', '.']

**B)** ['Este', 'é', 'um', 'aventureiro', 'de', 'São Paulo', '.']

**C)** ['Este', 'é', 'um', 'aventureiro', 'de', 'São', 'Paulo']

**D)** ['Este', 'aventureiro', 'São', 'Paulo', '.']

**E)** ['Este', 'é', 'um', 'aventureiro', 'de', 'São Paulo']

**4) Um dos métodos que faz a redução do vocabulário de um texto e abstração do significado ajuda na diminuição da complexidade de um texto. Esse conceito reduz a palavra ao seu radical. Por exemplo, a palavra carro tem o radical "carr". Assinale a alternativa que representa esse conceito.**

**A)** NLTK.

**B)** *stopwords.words().*

**C)** *Stemming*.

**D)** Remoção de código HTML/CSS.

**E)** *POS-Tagging*.

**5) Remoção de *stop-words* é um dos métodos de pré-processamento de texto que ajuda a deixar o texto mais limpo e mais fácil de ser entendido pelas máquinas. Cada idioma tem sua lista de *stop-words*. Considere a seguinte frase:**

**"Esse carro não é importante neste novo negócio".**

**Assinale a alternativa que apresenta a correta remoção de *stop-words* dessa frase.**

**A)** ['Esse', 'carro', 'é', 'importante', 'neste', 'novo', 'negócio'].

**B)** ['Esse', 'carro', 'não' , 'importante', 'neste', 'novo', 'negócio'].

**C)** ['não'].

**D)** ['Esse', 'carro', 'importante', 'neste', 'novo', 'negócio'].

**E)** ['Esse', 'carro', 'é'].

# Na prática

O PLN é o relacionamento entre computadores e linguagem humana. Mais especificamente, o PLN é o entendimento, a análise, a manipulação e a geração de linguagem natural do computador.

O PLN refere-se à análise de fala, tanto na fala audível quanto no texto de uma língua. Os sistemas de PLN capturam o significado de uma entrada de palavras (frases, parágrafos, páginas, etc.) na forma de uma saída estruturada. O PLN é um elemento fundamental da IA.

Nesse contexto, o processamento de texto é uma das principais áreas que fornece suporte para a área de PLN. Existem inúmeras técnicas de pré-processamento de texto que auxiliam em diversas aplicações. Assim, você, como profissional dessa área, deve estar atento à correta aplicação prática dos conceitos teóricos.

Neste Na Prática, você vai conhecer um estudo de caso no qual é possível resolver o problema de uma escola de Ensino Fundamental utilizando uma das técnicas de pré-processamento de texto e a ferramenta NLTK com Python.

# ESTUDO DE CASO:

## DESENVOLVIMENTO DE SOLUÇÃO QUE INFORMA O RADICAL DE DETERMINADA PALAVR

Uma das maiores preocupações de professores de Português é conseguir ensinar todos os conteúdos dessa matéria de forma que os alunos consigam entender e aplicar o conhecimento adquirido. No entanto, procurar determinados conteúdos no dicionário de forma manual pode desmotivar os alunos. Neste Na Prática, você vai conhecer um estudo de caso no qual uma escola solicita o desenvolvimento de uma solução computacional que informe o radical de palavras da língua portuguesa de maneira simples e rápida.

A escola Aprenzidado é uma instituição educacional de nível fundamental que precisava urgentemente de uma solução que fornecesse, de maneira rápida, o radical de determinadas palavras. A professora de Português percebeu que as crianças ficavam desmotivadas em ter de procurar o radical de uma palavra no dicionário de maneira manual.

### INFORMAÇÕES DE *SOFTWARE*

A necessidade da escola Aprendizado era encontrar uma forma rápida e simples de identificar o radical das palavras que o usuário digitasse.

Carlos, que é estudante na área de Tecnologia, resolveu aceitar esse desafio. Ele conhece uma maneira de resolver esse problema utilizando Python e a ferramenta NLTK.

### MODELAGEM DO PROBLEMA E DA SOLUÇÃO

Carlos modelou o problema e a solução da seguinte maneira:

1. criar um parâmetro de entrada que será a palavra da qual algum aluno tentará descobrir o radical;
2. retornar na tela o radical de determinada palavra de maneira simples e rápida.

### IMPLEMENTAÇÃO DA SOLUÇÃO

Este é um exemplo de implementação de um método que recebe determinada palavra digitada pelo usuário e imprime o radical dessa palavra utilizando o método *stem*(), que recebe uma palavra como parâmetro. É feito, nesse caso, o processo de *stemming*, que é um dos processos de pré-processamento de texto.

```
+ Code  + Text                                                    RAM  Editing
                                                                  Disk

import nltk

#nltk.download('rslp')

stemmer = nltk.stem.RSLPStemmer()

palavra = input("Digite palavra que quer obter o radical:")

#Imprindo o radical das palavras
print("O radical da palavra digitada é:")
print(stemmer.stem(palavra))

Digite palavra que quer obter o radical:carro
O radical da palavra digitada é:
carr
```

Conclui-se que, conhecendo bem a linguagem Python, a ferramenta NLTK e alguns métodos de pré-processamento de texto, é possível resolver diversos problemas do mundo real.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

# Saiba mais

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

## Funções de processamento de textos com a biblioteca NLTK em Python

Neste vídeo, são mostradas e explicadas algumas funções de processamento de texto utilizando Python e NLTK.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

## Introdução ao processamento de linguagem natural usando Python

Neste trabalho, é feita uma introdução do PLN utilizando a linguagem Python, e, na página 13, é explicada a ferramenta NLTK.

Aponte a câmera para o código e acesse o link do conteúdo ou clique no código para acessar.

## Business intelligence e análise de dados para gestão do negócio.

Na seção 5.2 na página 297 deste livro, são apresentadas as definições e a importância da análise e da mineração de textos na era da informação.

Conteúdo interativo disponível na plataforma de ensino!