

CIÊNCIAS DE DADOS (BIG DATA PROCESSING AND ANALYTICS)

Arquitetura de *Big Data*





Universidade Presbiteriana
Mackenzie

Modalidade a distância

RECUPERAÇÃO DA INFORMAÇÃO NA WEB E EM REDES SOCIAIS

**Trilha 4 – Processamento de Linguagem Natural:
extração de palavras-chave**

Professor: Luciano Moreira Camilo e Silva

Sumário

1. Introdução à Trilha	4
1.1. Caso prático	4
2. Pré-processamento de textos [Normalização].....	7
2.1. Texto insensível a maiúsculas e minúsculas	7
2.2. Remoção de acentuação e outros caracteres especiais.....	8
2.3. Remoção de plural	9
2.4. Remoção de palavras de parada.....	11
2.5. Remoção de palavras sem valor para o contexto.....	12
2.6. Visão da nuvem de palavras após pré-tratamento	13
3. Vetorização de textos	14
3.1. Codificação de caracteres em sistemas computacionais	15
3.2. Tokenização de palavras	18
3.3. Bigramas	19
4. Síntese.....	21
5. Referências	22

1. Introdução à Trilha

Nas trilhas anteriores, discutimos como as informações podem ser organizadas para fácil recuperação de dados (Trilha 1) e como recuperar e fazer a mineração desses dados¹ quando a fonte é um documento HTML disponível na web.

É possível obter dados estruturados dentro desses documentos. Porém, grande parte das informações obtidas são dados semiestruturados (aqueles que não seguem um rigor formal de modelagem, mas possuem etiquetas para identificação do conteúdo) ou não estruturados, como textos livres.

Há muita informação em dados não estruturados. Há casos clássicos de interpretação do conteúdo gerado por usuários em redes sociais na identificação do sentimento de um cliente em relação a uma marca ou produto (o cliente gostou do produto?). Avalia-se, por meio dos textos gerados pelo usuário em comentários sobre cinema, se a maioria gostou ou não do filme, e o que mais chamou atenção em determinadas cenas. Podemos gerar resumos automáticos de textos, entender o que um cliente digita em um *chatbot*² entre várias outras iniciativas de interpretação de textos.

A disciplina que lida com a interpretação de texto é conhecida como Processamento de Linguagem Natural (PLN, ou NLP em inglês). Nesta e nas próximas três trilhas, exploraremos um pouco sobre como trabalhar com essas informações não estruturadas que aprendemos a recuperar nas trilhas anteriores.

1.1. Caso prático

Ao contrário das trilhas anteriores nas quais os exemplos práticos foram apresentados apenas ao final do e-book, as trilhas de processamentos de linguagem natural apresentarão os exercícios e a teoria de forma conjunta, de modo a facilitar o entendimento dos conceitos.

¹ “Prospecção de dados (português europeu) ou mineração de dados (português brasileiro) (também conhecida pelo termo inglês *data mining*) é o processo de explorar dados à procura de padrões consistentes, como regras de associação ou sequências temporais, para detectar relacionamentos sistemáticos entre variáveis, detectando assim novos subconjuntos de dados.” (Mineração de dados, Wikipédia).

² “Chatbot (ou chatterbot) é um programa de computador que tenta simular um ser humano na conversação com as pessoas. O objetivo é responder as perguntas de tal forma que as pessoas tenham a impressão de estar conversando com outra pessoa e não com um programa de computador. Após o envio de perguntas em linguagem natural, o programa consulta uma base de conhecimento e em seguida fornece uma resposta que tenta imitar o comportamento humano.” (Chatterbot, Wikipédia).

Alguns dos conceitos utilizados nos processamentos de linguagem natural têm, por vezes, uma escrita matemática rebuscada para aqueles que não são versados em escrita técnica, assim como têm como objetivo descrever, com precisão, conceitos simples do dia a dia, como contagem e proporção.

Destarte, os conceitos apresentados em processamento de linguagem natural neste componente se darão a partir de exemplos práticos, claros, seguidos do necessário teor matemático para a real compreensão e expansão dos ensinamentos feitos. Ademais, aqueles que, porventura, queiram se aprofundar no tema são convidados a destrinchar as referências bibliográficas da seção de Referências.

Sem mais delongas, ao caso de uso!

A internet digitalizou muito da nossa vida moderna. Um dos primeiros casos a migrar para o mundo digital foram os anúncios classificados. Inicialmente uma lista de e-mails criados por Craig Newmark, o cariglist.org cresceu para ser um dos maiores sites do planeta.³

O site craigslist.org nunca fez muito sucesso no resto do mundo. E, em particular, no Brasil, o site fundado na argentina para ser uma cópia do craigslist.org tornou-se líder de mercado: **OnLine eXchange**, ou OLX (DALMAZO, 2011)

Os sites de anúncios classificados online, assim como suas contrapartes populares em jornais de grande circulação do século passado, são organizados por categorias: emprego, imóveis, automóveis, entre outros.

Utilizando os conhecimentos da trilha anterior, foram obtidas, por raspagem, informações acerca dos anúncios da categoria de animais, especificamente de cachorros. Foram coletados, ao todo, mais de 75 mil anúncios classificados.

Em sua maioria, são anúncios de venda de cachorro de raças. Uma nuvem de palavras, sem nenhum tratamento, pode ser vista na figura abaixo.

³ Conforme fonte on-line, como o <https://www.similarweb.com/top-websites/united-states/> de 1º de agosto de 2021. Acessado em 25 de setembro de 2021.

Figura 1 – Principais palavras dos anúncios da OLX na categoria cachorros



Fonte: Elaborada pelo autor.

Observa-se que, na imagem, das principais 30 palavras do anúncio classificado, as que mais aparecem são “filhotes” e sua variação “filhote”).

Ao longo dessa trilha, veremos como identificar e efetuar a contagem dessas palavras. Entenderemos o porquê da importância de se separar o texto em palavras no processamento de linguagem natural, assim como estudaremos quais tratamentos utilizar no texto para identificar as principais raças anunciadas.

2. Pré-processamento de textos [Normalização]

Uma das dificuldades de extrair valor em processamento de linguagem natural é a miríade de variações encontradas para expressar o mesmo tema, ou as pequenas nuances de variação entre eles.

Analisando os corpora ¹ de anúncios do site OLX, mesmo pela frequência de palavras observada na nuvem da Figura 1, notamos uma série de palavras que, apesar de compor os escritos originais, pode dificultar a interpretação por um sistema computacional, que é baseado com um conjunto de regras.

Desta forma, é comum e recomendado que se faça um pré-tratamento do texto, ou como é referenciado no meio de estudo da língua, um corpus linguístico.

Abaixo, temos alguns dos tratamentos mais comuns utilizados em pré-processamento de textos. Devemos, contudo, examinar o objetivo pretendido com o processamento da linguagem natural, a fim de evitar que o pré-processamento remova informações cruciais para a análise.

2.1. Texto insensível a maiúsculas e minúsculas

Como será apresentado mais a frente, aplicativos computacionais interpretam comandos de forma explícita. Por exemplo, pense em duas variáveis declaradas conforme abaixo:

```
1. Palavra1 = "Husky"
2. Palavra2 = "husky"
3.
4. Palavra1 == Palavra2
5. # >> False
```

Semanticamente, as duas palavras são idênticas, diferenciadas por uma ser início de frase ou pela preferência do autor do texto em destacar a raça em letra capital. Porém, devido à codificação e à lógica booleana de comparação de *strings*, as duas palavras

¹ "Corpus linguístico é o conjunto de textos escritos e registros orais em uma determinada língua e que serve como base de análise. Em vez de consultar nossas intuições, ou de 'extrair' informações dos falantes, penosamente, uma a uma, podemos examinar um vasto material que foi produzido espontaneamente na fala ou na escrita das pessoas proporcionando informações altamente confiáveis e isentas de opiniões e de julgamentos prévios, sobre os fatos de uma língua." (Corpus linguístico, Wikipédia).

serão tratadas como distintas.

Um tratamento bastante usado para evitar essa diferença nas comparações é converter todos os caracteres em seu correspondente em letra minúscula. Para quem está usando *Python*, o mesmo pode ser obtido pelo método `.lower()`.

```
1. Palavra1 = "Husky".lower()
2. Palavra2 = "husky".lower()
3.
4. Palavra1 == Palavra2
5. # >> True
```

2.2. Remoção de acentuação e outros caracteres especiais

Outro procedimento utilizado em boa parte dos tutoriais e livros para tratamento de textos é a remoção de acentos e outros caracteres especiais, como, no caso do português, o cedilha.

Principalmente em textos mais recentes e escritos por pessoas habituadas à escrita na internet, pode ser comum evitarem o uso desses artefatos linguísticos (MANNING; RAGHAVAN; SCHÜTZE, 2008).

Deve-se atentar ao caso de uso. Um idioma como o português (ou espanhol, ou francês) possui palavras com grafias semelhantes que se diferenciam apenas pelo uso do acento. Dependendo da análise que se deseja fazer, normalizar as palavras para remover tais caracteres pode comprometer boa parte da análise.

No caso em questão, como o objetivo é identificar as raças de cachorro anunciadas na OLX, entendemos que essa normalização simplificará a análise, sem comprometer nenhum valor semântico nos textos.

Outro tratamento, também descrito no livro de Manning, Raghavan e Schütze (2008) e amplamente divulgado nos tutoriais de processamento de linguagem natural, é a remoção de números e outros símbolos como cifrão, parênteses, arrobas, *hashtags* ou até mesmo emojis² e emoticons.³ Tal procedimento é obtido restringindo o texto aos caracteres básicos do alfabeto latino.

Novamente, deve-se observar a intenção do uso na análise do processamento do texto. Emoticons podem ser muito úteis na avaliação de sentimentos em redes sociais, bem como

² “Emoji é uma palavra derivada da junção dos seguintes termos: e (“imagem”? em japonês) + moji (“letra”? em japonês). (Emoji, Wikipédia).

³ “Emoticon é uma forma de comunicação paralinguística. Palavra derivada da junção dos termos em inglês emotion (emoção) + icon (ícone) (Emoticon, Wikipédia).

excluir números em análise de textos esportivos pode remover bastante da informação fornecida pelos autores.

Abaixo, um código em Python para fazer essa etapa da normalização:

```
1. from unicode import unicode
2. import string
3.
4. #Transformação em minúscula e Remoção de acentos
5. titles['normalized'] = titles['title'].str.lower().apply(lambda x: unicode(x))
6.
7. #Remoção de pontuação
8. titles['normalized'] = titles['normalized'].str.replace('{}'.format(string.punctuation), "")
9.
10. #Remoção de números
11. titles['normalized'] = titles['normalized'].str.replace('{}'.format(string.digits), "")
```

2.3. Remoção de plural

Assim como nos casos anteriores, o plural de uma palavra é um item a ser analisado se agrega ou não valor no contexto a ser analisado. Veja novamente a Figura 1. Nela, é possível observar a aparição das palavras “filhotes” e “filhote”. Em nosso objetivo de identificar as raças que estão anunciadas, talvez nenhuma das duas palavras agreguem e devam ser removidas como será mostrado no item 2.5.

De toda forma, assim como converter todas as palavras para minúscula pode ajudar a diminuir o corpus que será tratado, transformar todas os substantivos em seu equivalente singular resulta em uma quantidade menor de palavras a ser tratada.

O português é um idioma com um conjunto grande de regras para o tratamento do plural como pode ser visto em Araújo ([s.d.]). Inclusive, existem algumas regras com variações, como no caso do plural de palavras terminadas em “ÃO”. Temos aviões para avião, cães para cão e cristãos para cristão.

Por sorte, o que se quer fazer é o inverso: a partir do plural, queremos singularizar uma palavra, o que torna o exercício mais simples. Abaixo, vemos as regras, conforme explicitado por Araújo ([s.d.]):

- **Regra básica:** às palavras terminadas em vogal, ditongo oral e consoante “N”, acrescentamos a consoante “S”.
- **Palavras terminadas em “M”:** troca-se a consoante “M” por “NS”.
- **Palavras terminadas em “R” e “Z”:** às palavras terminadas em consoante “R” e “z” acrescentamos “ES”:
- **Palavras terminadas em “AI”, “EL”, “OL”, “UL”:** para esses casos, trocamos a

consoante “L” por “IS”, como em pardais, coronéis e faróis. São exceções:

- o mal – males;
- o cônsul – cônsules;
- o mel – mele ou méis;
- o fel – feles ou féis;
- o cal – cales ou cais;
- o aval – avals ou avais;
- o mol – moles ou móis;
- o real – réis (antiga moeda).

• **Palavras terminadas em “IL”:** há dois casos.

- o Palavras oxítonas, substituímos por “IS” como em cantil – cantis, canil – canis.
- o Palavras paroxítonas, substituímos por “EIS” como em míssil – misseis, fácil – fáceis, fóssil – fosseis.

• **Palavras terminadas em consoante “S”:**

- o Às monossílabas e oxítonas, acrescenta-se “ES” como em português – portugueses, inglês – ingleses, gás – gases.
- o As não-oxítonas ficam inalteráveis como pires, lápis, ônibus.

• **Palavras terminadas em “ÃO”:** existem três formas de pluralizar uma palavra terminada em “ÃO”, e a flexão depende do caso particular de cada palavra e consiste na substituição do “ÃO” por uma das três variações:

- o **“ÕES”:** corresponde há mais de 95% dos casos, como visto em Huback (2017), e tem como exemplos avião – aviões, balão – balões, nação – nações.
- o **“ÃES”:** é uma forma alternativa de formação de plural para palavras terminadas em “ÃO”. Exemplos: pão – pães, cão – cães, alemão – alemães.
- o **“ÃOS”:** é uma forma alternativa de formação de plural para palavras terminadas em “ÃO”. Exemplos: cidadão – cidadãos, irmão – irmãos, cristão – cristãos.

• **Palavras terminas em “X”:** estes casos permanecem inalterados, como em durex, tórax, córtex.

Assim, a flexão singular-plural no idioma português possui um conjunto de regras que dependem da entonação e possui, ainda, exceções, o que torna o exercício não trivial. Porém, para a grande maioria dos casos, é possível aplicar algumas regras para obtermos a flexão.

No idioma inglês, é possível encontrar bibliotecas em Python que auxiliam nessa transformação. Para o idioma português, o autor deste e-book iniciou um código que pode ser encontrado no link a seguir <<https://brasilecola.uol.com.br/gramatica/singular-plural.htm>> e convida os leitores a contribuir para a formação de uma biblioteca mais rica

para a língua deste texto.

2.4. Remoção de palavras de parada

Algumas palavras são utilizadas em um grande conjunto de frases, no entanto, apesar da correção gramatical, são termos que não agregam muito conteúdo semântico. Exemplos dessas palavras são os artigos “o”, “a”, “um” e “uma”. É comum, então, remover essas palavras da análise do processamento de linguagem natural.

Uma forma de gerar uma lista de palavras de parada é identificar em corpora quais são as palavras mais comuns, as que aparecem em praticamente todo corpus e que, por conseguinte, agregam pouco ou nenhum valor semântico para a análise. Métodos estatísticos que permitiram essa conclusão serão discutidos na próxima trilha.

Também encontramos trabalhos, como o apresentado por Junior, Vieira e Gonçalves (2008), que utilizam uma construção semântica, por exemplo, uma lista constituída de artigos, preposições e advérbios. O texto ainda discute outras formas de construção dessa lista.

Para este trabalho, utilizaremos a lista proveniente pela biblioteca NLTK, a qual pode ser conhecida pelo livro escrito por Bird, Klein e Loper (2009).

Tabela 1 – Conjunto de palavras de parada do idioma português da biblioteca NLTK

a	já	seu	hão	esse	está	pelas	serei	aquele	aquelas	estivera	estávamos
o	eu	sua	sou	eles	haja	você	serão	aquela	estamos	houvemos	estejamos
e	só	nos	são	você	eram	nosso	seria	aquilo	estavam	houveram	estivesse
é	me	até	era	essa	fora	nossa	tenho	estive	estejam	houvesse	estiverem
à	às	ela	fui	suas	seja	delas	temos	esteve	estiver	houverem	houvessem
de	tu	sem	foi	numa	será	estes	tinha	estava	havemos	houverei	houvermos
do	te	aos	for	elas	tive	estas	tenha	esteja	houvera	houverão	houveriam
da	há	nas	tem	qual	teve	estou	tiver	houver	hajamos	houveria	tivéríamos
em	que	num	tém	este	terá	estão	terei	éramos	houverá	fôssemos	estivessem
um	com	nem	para	dele	muito	houve	terão	fossem	fôramos	seríamos	estivermos
os	não	meu	mais	lhes	entre	hajam	teria	formos	sejamos	tínhamos	houvéramos
no	uma	nós	como	meus	mesmo	somos	quando	seriam	seremos	tenhamos	houveremos
se	por	lhe	pelo	teus	minha	fomos	também	tinham	tivemos	tivessem	tivéssemos
na	dos	vos	pela	tuas	pelos	foram	depois	tivera	tiveram	tivermos	estivéríamos
as	mas	teu	isso	dela	deles	sejam	minhas	tenham	tivesse	teríamos	houvéssemos
ao	ele	tua	seus	esta	essas	fosse	nossos	teriam	tiverem	estivemos	houveríamos
ou	das	hei	quem	isto	esses	forem	nossas	aqueles	teremos	estiveram	estivéssemos

Deve-se tomar cuidado ao utilizar a remoção de palavras de paradas. Alguns autores, como Manning, Raghavan e Schütze (2008) explicam que, em alguns casos, estas palavras podem mudar o conceito semântico. “presidente Brasil” traz menos significado semântico do que “presidente do brasil”.

Em nosso caso, como o objetivo é identificar nome de raças de cachorro anunciados no OLX, a remoção destas palavras deve resultar em um conjunto de palavras mais adequados à finalidade (BRAGA, 2009).

2.5. Remoção de palavras sem valor para o contexto

Após todos os tratamentos feitos do item 2.1 ao 2.4, o texto remanescente do exercício proposto está melhor para ser consumido e interpretado computacionalmente.

Porém, ao analisar as palavras remanescente, devidamente despluralizadas, convertidas todas em iniciais minúsculas e considerando o objetivo de encontrar as principais raças de cachorro anunciadas no site OLX, deparamo-nos com um conjunto de palavras que não agregam valor, tais como “lindo” e “imperdível” ou ainda palavras relacionadas à cor do animal, como “branco”, “preto” e “caramelo”.

Neste caso, e somente por termos certeza do objetivo, cria-se uma lista de palavras a serem ignoradas, tal como foi feita nas palavras de parada. Para este exercício, fizemos uma lista inicial com as palavras demonstradas na tabela abaixo.

Tabela 2 – Lista de palavras a serem ignoradas para o exercício proposto

c	dia	hora	total	ideal	pronta	fisica	vendese	adquirir	incríveis	lindissimo	veterinaria
r	via	leia	todas	reais	pronto	cabeca	reserva	vacinada	garantias	disponivel	assistencia
x	vai	veja	todos	puros	lindas	adulto	reserve	adoravel	microchip	entregamos	vermifugado
ja	chip	veja	todas	sonho	lindos	vacina	pelagem	condicao	vacinados	entregamos	apartamento
so	hoje	casa	venha	preco	padrao	azuis]	tamanho	amorosos	saudaveis	disposicao	companheiro
sp	raca	apto	belos	unico	compro	prontos	conosco	imediate	vacinados	parcelamos	filhotinhos
hs	loja	belo	lojas	unica	compra	entrega	filhote	conhecer	adoraveis	belissimos	maravilhosos
cm	dias	azul	porte	sonho	chamar	suporte	amarelo	namorada	seguranca	felicidade	oportunidade
ate	voce	macho	amigo	otimo	melhor	confira	machinho	conferir	condicoes	transporte	parcelamento
pra	todo	femea	super	otima	cartao	adquira	pedigree	proprias	polegadas	oferecemos	vermifugados
vet	toda	lindo	hiper	chama	varias	tamanho	lindinho	servicos	perfeitos	brincalhao	olhocaramelo
vet	vida	linda	meses	cinza	visita	vacinas	lindinha	retirada	descricao	exclusivas	exclusividade
top	pura	fofos	vezes	preto	unicos	garanta	fofinhos	cachorro	alexandre	exclusivos	exclusividades
bem	casa	saude	docil	preta	unicas	amoroso	garantia	legitimo	companhia	beneficios	brancochocolate
vez	capa	racas	bebes	canil	otimos	contato	contrato	tamanhos	qualidade	filhotinho	vacinado
sim	amor	vendo	chame	olhos	otimas	contato	promocao	linhagem	companhia	merlepreto	maravilhoso
nao	face	vende	horas	white	gratis	procuro	whatsapp	filhotes	excelente	lindissimos	
hrs	aqui	venda	ainda	black	fofura	alegria	gratuito	vermelho	exclusiva	disponiveis	
ver	novo	juros	unica	merle	branco	procura	saudavel	merlered	exclusivo	veterinario	
lar	bebe	whats	feliz	machos	cabeca	visitar	clinicas	lindinhos	pagamento	informacoes	
apt	info	ligue	fotos	femeas	ultimo	conheca	melhores	lindinhas	chocolate	procedencia	

Fonte: Elaborada pelo autor.

2.6. Visão da nuvem de palavras após pré-tratamento

A Figura 2 apresenta o resultado do pré-tratamento. Apesar de enxergarmos uma evolução em relação à Figura 1 na identificação das raças, ainda há processamentos necessários, os quais serão abordados na seção 3.

Figura 2 – Nuvem de palavras resultante do pré-tratamento

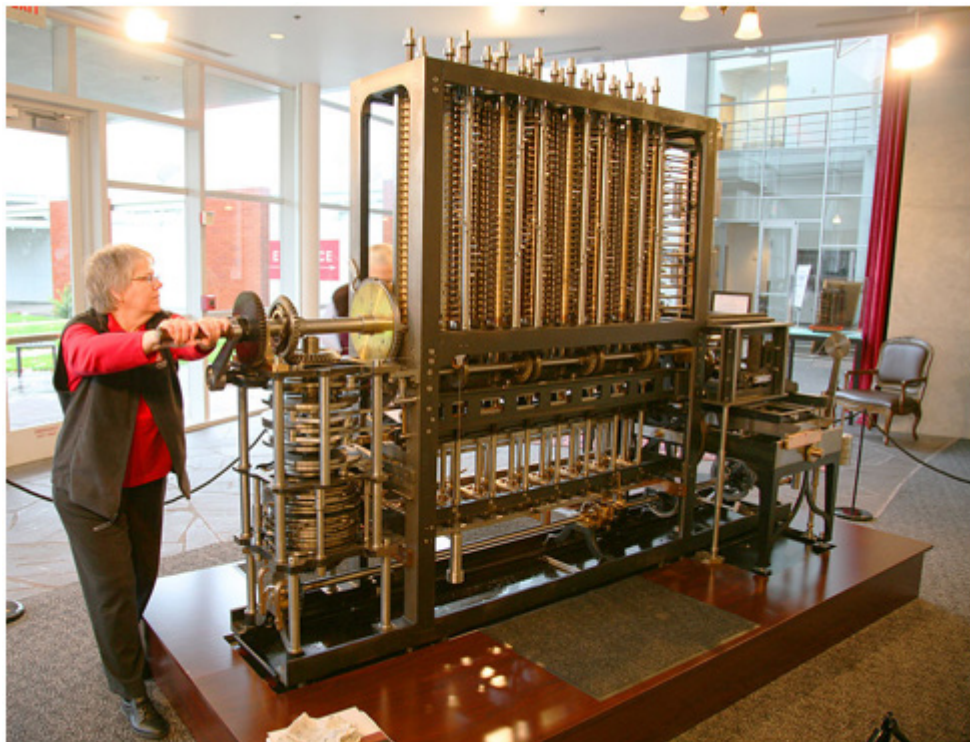


Fonte: Elaborada pelo autor.

3. Vetorização de textos

Charles Babbage, no início do século XIX, descreveu os computadores modernos como máquinas analíticas. Porém, ao invés de mecânicas, elas seriam máquinas eletrônicas (GRAHAM-CUMMING, 2010). Essas máquinas já se baseavam em uma unidade de processamento central ou, em inglês, CPU.¹

Figura 3 – Máquina Analítica, desenhada por Charles Babbage e exposta Museu de História da Computação em Mountain View, CA



Fonte: Wikimedia Commons.²

Anos mais tarde, em 1937, com seu seminal artigo “On Computable Numbers, with an Application to the Entscheidungsproblem”, Alan Turing descreveu o que seria a máquina computacional universal, também baseada em memória e unidades de processamento.

¹ “A unidade central de processamento ou CPU (Central Processing Unit), também conhecida como processador, é a parte de um sistema computacional, que realiza as instruções de um programa de computador, para executar a aritmética básica, lógica, e a entrada e saída de dados. O papel da CPU pode ser comparado ao papel de um cérebro no funcionamento de um computador. Isto é, realiza operações lógicas, cálculos e processamento de dados.” (Unidade central de processamento, Wikipédia).

² Veja mais fotos interessantes sobre máquinas históricas no endereço <<https://www.flickr.com/photos/40648743@N00/sets/72157623284316506/>>.

A máquina de Turing seria capaz de resolver qualquer problema analítico e, inclusive, emular qualquer outra máquina computacional imaginada pelo homem. Porém, as unidades de processamento são otimizadas para o processamento aritmético e dependem de que as entradas tenham significados que possam ser interpretados pelos programas.

Computadores modernos, sem os programas, não conseguem entender nada diferente de zeros e uns. E, para interpretar letras, palavras, frases e textos inteiros é necessário transformá-los em algum valor que os programas possam trabalhar em cima. É aí que entra a transformação desses símbolos em números, ou como é chamado em um estrangeirismo difundido entre os estudantes de processamento de linguagem natural, a *tokenização*.

3.1. Codificação de caracteres em sistemas computacionais

Os sistemas computacionais modernos são máquinas que trabalham com o sistema de numeração binário.³ Fortemente influenciado por outro gênio da matemática, John von Neumann⁴ e sua arquitetura de computadores (presente nos computadores até os dias de hoje), estabelecem que os computadores, como máquinas de processamento, recebem uma entrada de dados e uma entrada de programa (na mesma fita imaginada por Alan Turing) e produzem uma saída.

Os processos aritméticos que ocorrem dentro da unidade central de processamento são codificados no sistema binário, ou seja, em zeros e uns. Números decimais, como os que estamos acostumados a utilizar no dia a dia, são transformações da base binária por um programa de computador, o que facilita a comunicação com os seres humanos.

A apresentação de textos, como este que você está lendo e que foi produzido utilizando um computador, só é possível pela codificação dos zeros e uns em letras entendíveis pelo computador. Ao salvar uma letra “A” no computador, muito provavelmente o arquivo no disco gravará o byte **01000001**.

A necessidade de representação dos caracteres latinos era essencial para a integração das grandes máquinas computacionais da década de 1960 com as impressoras que

³ “Os computadores digitais trabalham internamente com dois níveis de tensão, pelo que o seu sistema de numeração natural é o sistema binário. Com efeito, num sistema simples como este é possível simplificar o cálculo, com o auxílio da lógica booliana. Em computação, chama-se um dígito binário (0 ou 1) de bit, que vem do inglês Binary Digit. Um agrupamento de 8 bits corresponde a um byte (Binary Term).” (Sistema de numeração binário, Wikipédia).


⁴ “John von Neumann, nascido Margittai Neumann János Lajos (Budapeste, 28 de dezembro de 1903 — Washington, D.C., 8 de fevereiro de 1957) foi um matemático húngaro de origem judaica, naturalizado estadunidense. Foi professor na Universidade de Princeton e um dos construtores do ENIAC. Entre os anos de 1946 e 1953, von Neumann integrou o grupo reunido sob o nome de Macy Conferences, contribuindo para a consolidação da teoria cibernética. De fato, é considerado um dos mais importantes matemáticos do século XX.” (John von Neumann, Wikipédia).

eram as principais saídas dessas máquinas. Imprimir o código em binário, ou mesmo em sua representação decimal (65) ou hexadecimal (40) não era muito prático.

Em 1963 foi, então, publicado um documento técnico padronizando, a princípio, apenas para o mercado estadunidense, a tabela de codificação do alfabeto latino usado em inglês, utilizando um total de 7 bits (AMERICAN Standards Association, 1963). Como pode ser visto na figura abaixo, o padrão ASCII era composto de 128 combinações, reservando as 32 primeiras para comandos, como retorno de linha. E não previa a utilização de acentos ou outros caracteres de alfabetos não latinos, como o alfabeto grego ou cirílico.⁵

Figura 4 – Tabela ASCII original repicado do manual de impressora da década de 70

USASCII code chart



Column Row	0	1	2	3	4	5	6	7
0	0 0 0 0 0	0 0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0	NUL	DLE	SP	0	@	P	\	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(8	H	X	h	x
9	HT	EM)	9	I	Y	i	y
10	LF	SUB	*	:	J	Z	j	z
11	VT	ESC	+	;	K	[k	{
12	FF	FS	,	<	L	\	l	
13	CR	GS	-	=	M]	m	}
14	SO	RS	.	>	N	^	n	~
15	SI	US	/	?	O	_	o	DEL

Com o passar do tempo, foram criadas diversas outras formas de codificação de caracteres, o que deixou a troca de arquivos mais complexa. Cada sistema operacional e/ou programa de computador poderia escolher um formato de codificação diferente.

O Windows, em português, utilizava-se principalmente do padrão Windows 1252 ou ISO-8859-1. Porém não havia um padrão na indústria. Países ocidentais elaboravam seu padrão de codificação estendendo o padrão ASCII, aproveitando do fato de ASCII ser apenas 7 bits e utilizando o bit remanescente para codificar 128 novas combinações de

⁵ “O alfabeto cirílico, também conhecido como azbuka, é um alfabeto cujas variantes são utilizadas para a grafia de seis línguas nacionais eslavas e é composto de 33 letras. (Alfabeto cirílico, Wikipédia).

palavras sem perder a compatibilidade com o ASCII original. Países orientais precisavam de bem mais do que 8 bits.

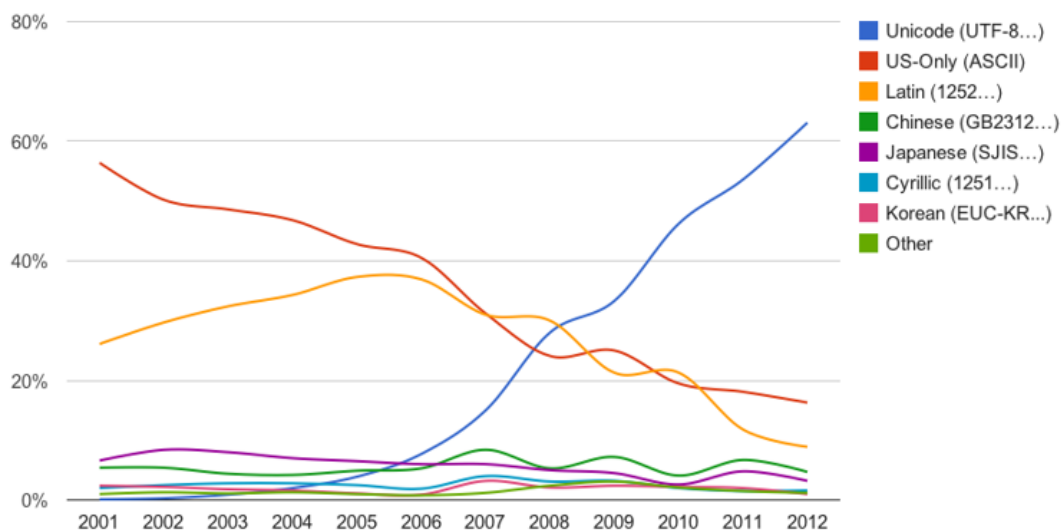
Não era incomum, nessa época, encontrar documentos em sites ou e-mails nos quais apareciam o sinal “?” no lugar de um caractere legível. Isso se devia à incapacidade do programa que tentava interpretar o arquivo de entender a codificação utilizada.

E, no mundo atual, com as centenas de emojis e emoticons sendo trocadas por uma miríade de aparelhos diferentes também foi demandada a padronização desses “caracteres”. Na visão do equipamento, seja um desktop Windows, um celular iOS ou Android, ou qualquer outro, os emojis são tratados como meros caracteres, 🤖?

De forma a resolver a complexidade, um esforço conjunto foi iniciado com a publicação de um livro em dois volumes, *The Unicode Standard* (1992). Foram criados alguns padrões, sendo o mais popular o UTF-8, que é um padrão de codificação de caracteres de tamanho variável (1 a 4 bytes), retrocompatível com o ASCII original, capaz de identificar mais de um milhão de caracteres distintos.

Abaixo, temos um gráfico que avalia a migração dos documentos web para o novo padrão e a representatividade dos padrões antigos.

Figura 5 – Adoção do padrão Unicode



Fonte: UNICODE (2012).

3.2. Tokenização de palavras

Trabalhar com palavras em sistemas de processamento de linguagem natural é semelhante ao relato de codificação explicado na seção anterior. Isso posto, assim como o UTF-8 codifica cada letra em seu respectivo código binário, a ideia com a *tokenização* de palavras é criar um código único para cada palavra (BIRD; KLEIN; LOPER, 2009).

Tentar interpretar uma palavra pela combinação de suas letras aumenta a complexidade computacional, pois um mesmo conjunto de caracteres pode formar um conjunto de palavras diferentes, com semânticas variadas, fenômeno conhecido como anagrama.

Em vista disso, é comum criar-se uma codificação própria para cada palavra dos corpora. Cada palavra do texto analisado é substituída por um valor em uma lista de palavras criada para o exercício. Com o exemplo do poema de Carlos Drummond de Andrade da Trilha 1 teríamos a seguinte tabela de codificação:

Tabela 3 – Tabela com a construção da codificação do texto do poema

Code	Word
1	acontecimento
2	caminho
3	esquecerei
4	fatigados
5	meio
6	nunca
7	pedra
8	retinas
9	tinha
10	vida

Fonte: Elaborada pelo autor.

Essa tabela permite que o sistema computacional que interpretará o texto o represente de forma mais suscinta e, conseqüentemente, o manipule de forma mais eficaz.

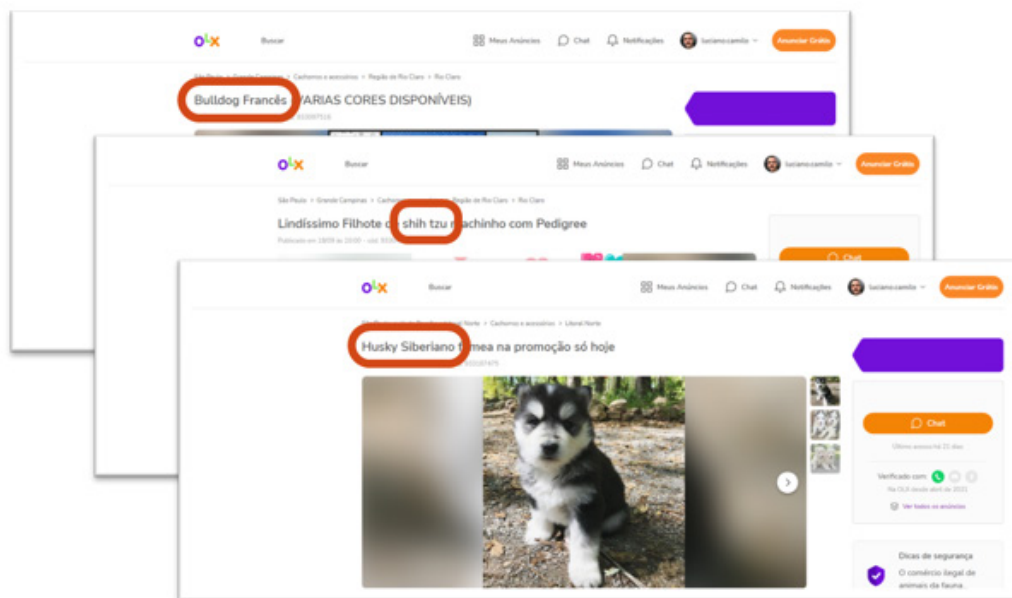
Antes de aplicar a codificação, é comum aplicar todos os pré-tratamentos de textos, vistos na seção 2 deste documento.

Uma nota importante: para alguns casos, como na tentativa de sumarização de textos maiores, efetuar a tokenização por frases inteiras, e não por palavras, pode ser mais indicado.

3.3. Bigramas

Ao criar uma lista de palavras pelo processo tradicional de tokenização, muitas vezes, excluimos palavras compostas, como algumas das raças de cachorro que tentamos encontrar nos anúncios classificados do OLX.

Figura 6 – Anúncios classificados do OLX, com destaque para as raças formadas por duas palavras



Fonte: Elaborada pelo autor.

Neste caso, a *tokenização* tradicional não resolve o problema encontrado. Para esses casos, existe o tratamento por bigramas, que é a *tokenização* por um par de palavras subsequentes.

Peguemos como exemplo o terceiro anúncio classificado da Figura 5, no qual se lê “Husky Siberiano fêmea na promoção só hoje”. Ignorando o pré-tratamento, a lista final de tokens seria {“Husky”, “Siberiano”, “fêmea”, “na”, “promoção”, “só”, “hoje”}.

No bigrama, combinando cada uma das duas palavras subsequentes, teríamos os seguintes tokens, formados por palavras duplas: {“Husky Siberiano”, “Siberiano fêmea”, “fêmea na”, “na promoção”, “promoção só”, “só hoje”}.

Considerando a pleora de anúncios diferentes, teremos a repetição de alguns desses tokens (de duplas de palavras). E, estatisticamente, apareceram mais vezes aquelas que são palavras compostas e comuns nos diversos anúncios classificados.

4. Síntese

Esta trilha foi a primeira de uma série que tratará do processamento de linguagem natural em textos recuperados da web.

Nas trilhas anteriores, foi debatido como construir robôs e aranhas que se conectam a diversos sites e extraem informações deles. Muitas dessas informações estão em formato não estruturados, porém com bastante conteúdo.

De forma mais ampla ao tratamento de linguagem natural, nesta trilha, exploramos os formatos de codificação de textos que podem ter surgido enquanto você criava um robô. Apesar da maior parte dos sites atuais usarem a codificação UTF-8, ainda é comum encontrar sites em outras codificações.

De forma mais específica, verificamos como extrair as informações das raças de cachorro mais populares no site de classificados, apenas usando técnicas comuns de processamento de linguagem natural.

Essas técnicas, apesar de simples, formam o alicerce do tratamento de texto em processamento de linguagem natural. Entender a **tokenização** é o primeiro passo para entender processamentos mais avançados, os quais veremos nas próximas trilhas.

5. Referências

AMERICAN Standards Association. *American Standard Code for Information Interchange. ASA X3.4*. New York: American Standards Association, 1963.

ARAÚJO, L. K. *Singular e plural*. Brasil Escola, s.d. Disponível em: <<https://brasilecola.uol.com.br/gramatica/singular-plural.htm>>. Acesso em: 2 out. 2021.

BIRD, S.; KLEIN, E.; LOPER, E. *Natural Language Processing with Python*. Sebastopol: O'Reilly Media, 2009.

BRAGA, Í. A. Avaliação da influência da remoção de stopwords na abordagem estatística de extração automática de termos. *Interinstitutional Center for Computational Linguistics*, 2009.

DALMAZO, L. A guerra dos classificados online. *Revista Exame*, 2011.

UNICODE over 60 percent of the web. *Google Oficial Blog*, 2 fev. 2012. Disponível em: <<https://googleblog.blogspot.com/2012/02/unicode-over-60-percent-of-web.html>>. Acesso em: 20 out. 2021.

GRAHAM-CUMMING, J. Let's build Babbage's ultimate mechanical computer. *Newscientist*, 2010.

HUBACK, A. P. Plurais irregulares do português brasileiro: efeitos de frequência. *Revista da ABRALIN*, v. 9, n. 1, 2017.

RIBEIRO Jr., L. C.; VIEIRA, R; GONÇALVES, P. N. Ontolp: Engenharia de ontologias em língua portuguesa. In: *Anais do XXVIII congresso da SBC-SEMISH*, Seminário Integrado de Software e Hardware, 2008.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. Introduction to information retrieval. *Cambridge University Press*, 2008.

THE UNICODE Consortium. *The Unicode Standard*, v. 1. Menlo Park: Addison-Wesley, 1992.

TURING, A. M. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, v. 2, n. 1, 1937, p. 230-265, 1937.

