UNIVERSIDADE DO VALE DO TAQUARI
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE ENGENHARIA DA COMPUTAÇÃO

IDENTIFICAÇÃO COMPUTACIONAL DE EMOÇÕES NEGATIVAS EM TEXTOS DA INTERNET: PROPOSTA DE APOIO AO USUÁRIO PARA CONTROLE EMOTIVO

Maiquel Jardel Ludwig

Lajeado, junho de 2018.

Maiquel Jardel Ludwig

IDENTIFICAÇÃO COMPUTACIONAL DE EMOÇÕES NEGATIVAS EM TEXTOS DA INTERNET: PROPOSTA DE APOIO AO USUÁRIO PARA CONTROLE EMOTIVO

Monografia apresentada ao Centro de Ciências Exatas e Tecnológicas da Universidade do Vale do Taquari, como parte dos requisitos para a obtenção do título de bacharel em Engenharia da Computação. Área de concentração: Inteligência Artificial e Processamento de Linguagem Natural.

ORIENTADOR: Me. Juliano Dertzbacher

Lajeado, junho de 2018.

AGRADECIMENTOS

À minha esposa Jaine, pelo suporte constante na trajetória de minha formação, pelas iniciativas e ideias de implementação, pelas revisões e sugestões de melhoria e, principalmente, pelo apoio incondicional neste e outros desafios.

Aos meus pais Araci e Sérgio e ao meu irmão Marlon, pela sólida estrutura familiar que sempre representaram e que foi fundamental para minha evolução acadêmica e profissional.

Ao professor e orientador Me. Juliano Dertzbacher pelos conselhos e incentivos, pelas críticas e sugestões de melhoria que foram essenciais para a construção deste trabalho, pelo tempo dedicado nas revisões e encontros e, por fim, pelo laço de amizade criado.

Aos envolvidos nas etapas de testes, pelas sugestões de melhoria e pela amizade cultivada: Jaine Elise Gräf Ludwig, Joice Elis Gräf, Marlon Vinícius Ludwig, Marcos Hilara, Pablo Petzen, Ivan Lampert, Alex Carvalho, Mateus Silva e Matheus Netto.

A todos os professores, colegas e coordenadores que participaram de minha jornada acadêmica.

RESUMO

Com a internet cada vez mais presente no cotidiano das pessoas, alguns usuários utilizam do anonimato e da inconsequência para expressar sua raiva, preconceito e intolerância. Esta monografia apresenta a pesquisa e o desenvolvimento de uma solução em software capaz de detectar aspectos emotivos nos textos inseridos por usuários da internet, visando a redução dos conteúdos agressivos, principalmente nas redes sociais utilizadas amplamente pelos jovens. O software considera a utilização de vários segmentos da Inteligência Artificial como o Processamento de Linguagem Natural e o Aprendizado de Máquina, utilizando de textos da internet para a composição das bases de treinamento dos algoritmos de classificação. Os resultados foram obtidos a partir de três rodadas de testes com usuários, sendo necessária a calibragem dos algoritmos a cada iteração. Uma breve comparação entre os algoritmos Naive Bayes e Regressão Logística foi realizada sem diferenças de efetividade relevantes, concluindo-se que a qualidade do grupo de treinamento possui maior relevância na efetividade das classificações de textos agressivos do que a escolha entre esses dois algoritmos.

Palavras-chave: Processamento de Linguagem Natural (PLN), Inteligência Artificial, Computação Afetiva.

ABSTRACT

With the internet increasingly present in people's daily lives, some users take advantage of the anonymity to do inconsequent acts expressing their anger, prejudgement and intolerance. This monograph presents the research and development of a software solution capable of detecting emotive aspects in the texts inserted by Internet users, aiming at the reduction of aggressive content, mainly in the social networks widely used by young people. The software considers the use of several segments of Artificial Intelligence such as Natural Language Processing and Machine Learning. It uses internet texts for the composition of the training bases of the classification algorithms. The results were obtained from three rounds of tests with early adopter users, being necessary the calibration of the algorithms at each iteration. A brief comparison between the Naive Bayes and Logistic Regression algorithms was performed without any relevant differences in effectiveness, and it was concluded that the quality of the training group is more relevant to the effectiveness of aggressive text classifications than the choice between these two algorithms.

Keywords: Natural Language Processing (NLP), Artificial Intelligence, Affective Computing.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API	*Application Programming Interface*
CSS	*Cascading Style Sheets*
FGV	Fundação Getúlio Vargas
HTML	*HyperText Markup Language*
IA	Inteligência Artificial
IP	*Internet Protocol*
LN	Locução Nominal
LV	Locução Verbal
MVC	*Model View Controller*
NLTK	*Natural Language Toolkit*
PLN	Processamento de Linguagem Natural
WEB	*World Wide Web*

SUMÁRIO

1 INTRODUÇÃO

A linguagem por nós representada através da fala e da escrita, apesar de trivial, contém características e nuances que somente são percebidas quando inseridas em um contexto. Esse contexto é essencial para que nosso cérebro consiga formar a semântica correta, que segundo Aurélio (2017A), é o ramo da linguística que estuda o significado das palavras. Na verdade, nosso cérebro é extremamente eficiente em consertar problemas de semântica, quando a mensagem inclui a informação de um contexto.

Tal fato foi comprovado pelo estudo realizado por Van Berkum *et al*. (2008), onde um grupo de voluntários recebeu frases com falhas semânticas como "eu lavo minhas mãos com cavalo e sabão", já outro grupo recebeu frases com falhas de contexto como “tenho uma tatuagem aqui atrás”. Em ambos os casos, as atividades cerebrais monitoradas foram muito similares e todos os voluntários compreenderam as frases, confirmando que o cérebro humano é capaz de corrigir as lacunas de contexto e semântica quase em tempo real, na verdade, são necessários entre duzentos e trezentos milissegundos após ouvir a palavra fora de contexto ou semântica.

Já em termos computacionais esse entendimento é bem mais complicado. Os computadores contam com uma linguagem própria, afinal, um computador é composto basicamente por um conjunto de caminhos e “interruptores” por onde trafegam os elétrons. A única forma de comunicação constitui-se de manipular tais interruptores a fim de impedir ou não que os elétrons passem por ali. Os códigos que controlam esses

interruptores são chamados de programas, os quais são escritos em linguagens de alto nível, compostas por suas próprias sintaxes. Segundo Aurélio (2017B), a sintaxe pode ser descrita como “parte da linguística que se dedica ao estudo das regras e dos princípios que regem a organização dos constituintes das frases” ou ainda, um “conjunto de regras que regem a escrita de uma linguagem de programação”. Essas linguagens são traduzidas e compiladas para outras linguagens cada vez menos abstratas até atingir a linguagem que efetivamente a máquina é capaz de compreender (e obedecer).

As linguagens de alto nível atuais já oferecem uma forma fácil de comunicação com os computadores mas, mesmo assim, estão longe de oferecer uma conversa casual com o programador. O conceito criado para lidar com esse problema é denominado Processamento de Linguagem Natural (PLN) e será amplamente utilizado nesta monografia. Conforme Goodfellow *et al*. (2016), o PLN é o uso de linguagens humanas, como o português ou o inglês, por um computador.

Ao entendermos, de forma computacional, como é feita a comunicação entre humanos, temos a oportunidade de criar programas e ferramentas capazes de nos auxiliar na classificação, agrupamento, tradução e tomada de decisão sobre essas informações. Não são poucas as novas informações disponibilizadas na internet. A cada segundo que você investe lendo esta monografia, segundo o Youtube (2017), são assistidas mais de onze mil horas de vídeo pelos usuários da plataforma. Já o Facebook (2017) registrou em junho de 2017 um total de 2,01 bilhões de usuários ativos em sua rede social.

Essas informações são criadas por pessoas, distribuídas por computadores e consumidas por outras pessoas novamente. A fim de otimizar e qualificar todos esses dados, atualmente há um forte investimento por parte de grandes companhias tecnológicas, tanto em soluções envolvendo o PLN como o IBM Watson (IBM, 2017), o Apple Siri (APPLE, 2017) e o Microsoft Cortana (MICROSOFT, 2017), quanto com analisadores e classificadores de texto como os serviços *antispam* dos grandes provedores de e-mail e o Google Translator (GOOGLE, 2017).

Tais tecnologias fazem parte de um ramo mais amplo da computação, conhecido como Aprendizado de Máquina, que basicamente utiliza um grande volume de dados históricos como exemplo para a resolução de problemas posteriores, gerando um verdadeiro conhecimento computacional adquirido a partir de dados passados (ALPAYDIN, 2010). Ainda referente ao volume de dados e o Aprendizado de máquina, Domingos (2015, p. 30) defende:

> Quando consideramos o machine learning dessa forma, duas coisas se destacam imediatamente. A primeira é que quanto mais dados temos, mais aprendemos. Se não houver dados? Não há nada a aprender. Muitos dados? Muito a aprender. É por isso que o machine learning está surgindo em todos os lugares, conduzido pela quantidade exponencialmente crescente de dados [...]. A segunda é que o machine learning é uma arma com a qual podemos derrotar o monstro da complexidade [...].

Infelizmente nem todo o conteúdo criado, pode ser considerado de qualidade. A Brand24 é uma ferramenta digital para monitoramento de mídias (BRAND24, 2017) muito utilizada na área de marketing digital. Ela permite rastrear a utilização de termos nas principais redes sociais (Twitter, Facebook, Instagram e outras) com a utilização de palavras-chave específicas. Com ela foi possível descobrir que entre a segunda metade de julho e a primeira metade de agosto de 2017, os termos “filho da puta”, “caralho”, “merda” “bosta” e “porra” foram mencionados pelos usuários mais de setenta e duas mil vezes e visualizadas por outros usuários quase vinte milhões de vezes.

A área da computação que engloba o entendimento da relação entre emoções e os meios tecnológicos, é conhecida como Computação Afetiva. Segundo Picard (1997), a Computação Afetiva é a “Computação que está relacionada com, que surge de ou deliberadamente influencia emoções”. Portanto, todo esse volume de informações repleto de conteúdo de baixa qualidade, influencia emocional e negativamente os usuários leitores.

Para intensificar ainda mais os impactos, a TIC Kids Online Brasil (CGI.BR, 2015, p. 170) revela que:

> [...] as redes sociais on-line estão bastante disseminadas entre os jovens usuários da rede (87%), tanto entre os residentes de áreas urbanas (87%) como de áreas rurais (83%), e nas diferentes classes sociais (93% entre as crianças usuárias de Internet das classes AB, 85% na classe C e 80% nas classes DE).

> No entanto, é preciso ressaltar que esse uso é maior entre os mais velhos: 96% dos adolescentes de 15 a 17 anos e 93% dos de 13 a 14 anos reportaram ter perfis em redes sociais, percentual que cai para 63% entre os que possuem de 9 a 10 anos e 79% entre as crianças de 11 a 12 anos de idade [...].

Considerando esse elevado número de crianças e adolescentes utilizando as redes sociais, sujeitas a visualizar e reproduzir conteúdos textuais grosseiros e negativos, detectou-se a possibilidade de aliar os conceitos psicológicos que compõem as emoções, com os avanços tecnológicos da área da computação, para a elaboração de um software detector de emoções negativas a partir de pequenos textos inseridos pelos usuários na internet.

1.1 Tema

O tema central deste trabalho refere-se à identificação computacional de emoções negativas e comportamento agressivo em textos da internet, utilizando o Processamento de Linguagem Natural e o Aprendizado de Máquina para auxiliar o usuário no autocontrole emotivo.

Como aplicação prática do tema a ser explorado, tem-se por meta a elaboração de um software capaz de compreender a representação de emoções em textos utilizando o Processamento de Linguagem Natural, passando pela formatação, classificação e treinamento de algoritmos específicos. Esse desenvolvimento auxiliará a compreender os princípios da Computação Afetiva e sua aplicabilidade em um cenário real.

1.2 Problema

A internet é um conglomerado de usuários gerando informações de todos os tipos. Nessas informações são incluídas mensagens na forma de ódio, preconceito ou

até mesmo as tão comuns “zueiras”, contendo palavras de baixo calão embutidas em piadas de mau gosto. O anonimato e o distanciamento físico dão incentivo a esse comportamento, principalmente nas redes sociais onde as mensagens podem ser direcionadas diretamente a outras pessoas ou empresas, muitas vezes sem consequências reais.

É possível lidar com toda essa massa de informações negativas de forma eficiente e sem prejudicar os direitos de liberdade de expressão dos usuários na internet? Como a computação, em conjunto com os mecanismos de aprendizado de máquina, pode auxiliar a mapear e a reduzir esses comportamentos agressivos na internet?

1.3 Objetivo Geral

Identificar o contexto emotivo de pequenos textos da internet para, desta forma, efetuar o treinamento de um algoritmo capaz de detectar comportamentos agressivos de usuários de internet e, assim, permitir que esse perfil de usuário possa ser conscientizado.

1.3.1 Objetivos Específicos

- Demonstrar como os comportamentos emotivos são representados em textos da internet e como eles podem ser processados e detectados de maneira computacional.
- Com o apoio de mecanismos de mineração de dados, construir uma base de dados baseada em comentários raivosos ou preconceituosos da internet, para que sejam utilizados como meios supervisores de treinamento de algoritmos para o Processamento de Linguagem Natural.

- Com a utilização de algoritmos de classificação em conjunto com o Processamento de Linguagem Natural, propor uma ferramenta capaz de auxiliar o usuário a controlar suas emoções negativas na internet.

1.4 Justificativa

O ser humano, como um ser social, é capaz de danificar seus laços afetivos com as outras pessoas, principalmente quando a comunicação entre elas é feita de forma agressiva. Esse comportamento é intensificado quando realizado pela internet, onde o sentimento de responsabilidade é protegido pela distância e os usuários sentem-se mais à vontade para reproduzir termos injuriosos.

Com os avanços da tecnologia, principalmente em termos de capacidade de processamento, tornou-se possível o manejo de um grande volume de informações para o aperfeiçoamento de mecanismos de captação, mineração e classificação de dados. Com ferramentas e algoritmos específicos, é possível tirar proveito de toda essa quantidade de informações utilizando do Aprendizado de Máquina para mapear, compreender e prever comportamentos.

Esses mecanismos podem ser incorporados em um software específico, capaz de coletar, formatar e aplicar algoritmos de classificação e probabilidades. Com base no comportamento atual de usuários com perfil violento na internet, é possível compreender alguns desses padrões textuais, gerando processos iterativos de treinamento dos algoritmos, que tendem a ficar mais otimizados e precisos ao longo do tempo.

Esta pesquisa, e o produto resultante terão condições de auxiliar na prevenção desses comportamentos nocivos à comunidade conectada à internet, além de averiguar se o estado da arte atual permite a aplicação dos métodos de classificação na seleção e no entendimento de emoções por um computador. Uma vez validado o processo em

nível de usuário doméstico, será possível ampliar a escala de utilização da ferramenta para um âmbito mais corporativo, onde há uma grande demanda para o controle de qualidade no atendimento a clientes de forma online, seja por e-mail ou seja por *chats* online. A ferramenta aqui proposta poderá auxiliar a garantir que a qualidade das comunicações seja mantida.

Outro motivo para a escolha deste tema está relacionado ao fato de que, tanto o Processamento de Linguagem Natural quanto a aplicação na classificação emotiva de textos em português, ainda são pouco explorados em nossa instituição de ensino e em boa parte das academias de ensino nacionais. Mas há fortes indícios de que estes assuntos se tornem cada vez mais presentes em nosso cotidiano. Atualmente já é possível contar com vários serviços inteligentes, desenvolvidos em sua maioria, por grandes empresas de fora do país. Entre esses serviços, podemos destacar os *chatterbots,* os assistentes pessoais acionados por voz disponíveis nos *smartphones* modernos, os serviços classificadores de conteúdo e de auto-resumo e por fim, os serviços para sugestões de produtos baseado em comportamentos do usuário utilizados em grandes varejistas da internet.

A computação neste caso, é apenas uma ferramenta de apoio ao desenvolvimento dos processos automatizados que facilitam nossa vida e das empresas. No escopo desta pesquisa, esses processos envolvem aspectos linguísticos, psicológicos e comportamentais. Ou seja, será uma mescla do estudo de características humanas com mecanismos e aplicações computacionais, reforçando a ideia de que a computação pode oferecer suporte para a elaboração de soluções em qualquer outra área de estudo.

Este trabalho está estruturado a partir da apresentação de conceitos psicológicos das emoções e como elas podem ser expressas através de textos inseridos em um ambiente de internet. Em seguida, são apresentados os referenciais necessários para a aplicação da computação afetiva e dos mecanismos essenciais para o processamento de linguagem natural. Esses estudos e conceitos são utilizados na construção da

metodologia para o alcance dos objetivos e são aplicados em no desenvolvimento de um software para operacionalizar todos esses processos.

Há um capítulo dedicado ao detalhamento dos conceitos, tecnologias e ferramentas envolvidas na construção da ferramenta e que, após esse processo, é submetida a várias rodadas de testes com usuários. Os resultados preliminares resultaram em ajustes no software, aumentando o seu percentual de efetividade ao longo da execução dos testes.

Por fim, esta monografia é finalizada com o capítulo de conclusões, onde há um mescla da análise dos resultados obtidos nos testes com as opiniões pessoais e aprendizados adquiridos ao longo do processo de escrita e desenvolvimento da ferramenta. Como era de se esperar, mais questionamentos e problemas surgem a partir deste estudo, elencando novos temas para pesquisa e desenvolvimento futuros. O tema é bastante amplo e ao mesmo tempo pouco explorado, evidenciando que a área de Aprendizado de Máquina e a Computação afetiva possuem um potencial de exploração muito grande.

2 REFERENCIAL TEÓRICO

A união entre o estudo computacional da linguagem e a compreensão de emoções, requer um conhecimento teórico extremamente amplo. É impossível falar do entendimento emotivo sem envolver questões psicológicas, assim como é impossível discutir o comportamento inteligente de um computador sem envolver assuntos filosóficos.

As seções deste capítulo servirão de embasamento para a aplicação metodológica em um experimento prático que envolve a detecção de emoções. Serão abordados assuntos voltados à psicologia comportamental no estudo das emoções, como elas influenciam no comportamento agressivo dos humanos e como esses comportamentos são comunicados através de textos na internet. Em seguida, veremos como a computação vem evoluindo no entendimento de linguagens humanas (computação afetiva) e como a inteligência artificial aborda esse tema. Nesse processo será necessário compreender as estruturas textuais, bem como suas composições morfológicas para que possam ser aplicados os algoritmos de aprendizado de máquina e classificação de textos.

2.1 Emoções

O conceito de emoções não é unanimidade na literatura. Segundo Bradley e Lang (2006), “há quase mais definições do que há investigadores”. Essas definições variam de acordo com a área onde são estudadas, portanto, esta monografia irá ater-se somente nas definições do ponto de vista da psicologia e da computação. Por se tratar de um assunto muito diverso, porém nem um pouco unânime, deve-se canalizar o estudo de emoções nos tópicos que permitirão a melhor abordagem do tema em questão.

Para Darwin (1872), as emoções são expressas por humanos e animais de forma bastante similar, sendo extremamente importantes para os comportamentos instintivos (principalmente o medo) e além disso, as emoções desempenham um papel muito importante na comunicação entre os seres de mesma espécie, uma vez que as expressões corporais transmitem informações sutis, mas perceptíveis aos seres semelhantes. Um sorriso no rosto de um amigo pode indicar que ele está em um bom dia, já uma testa enrugada pode demonstrar uma apreensão da parte dele. Ou seja, podemos comprovar facilmente as palavras do autor, apenas observando o comportamento das pessoas ao nosso redor.

Dentre as diversas divisões nos conceitos de emoção encontradas na literatura, as perspectivas que mais se encaixam com os objetivos desta monografia são a Cognitiva e a Construção Social. A primeira, considera que toda emoção está relacionada a um comportamento, e alterando-se um comportamento, altera-se a emoção. Tais comportamentos estão vinculados diretamente a aspectos biológicos do ser humano (CORNELIUS, 2000). Já a segunda perspectiva considera que, independentemente da formação biológica do ser humano, as emoções são construídas a partir da exposição dos seres humanos aos meios sociais e culturais (CORNELIUS, 2000). Considerando que o tema desta monografia se resume em avaliar o comportamento emotivo dos usuários que estão contextualizados em um ambiente

social (redes sociais e internet), ambas as perspectivas dos autores vão de encontro ao que é proposto no presente.

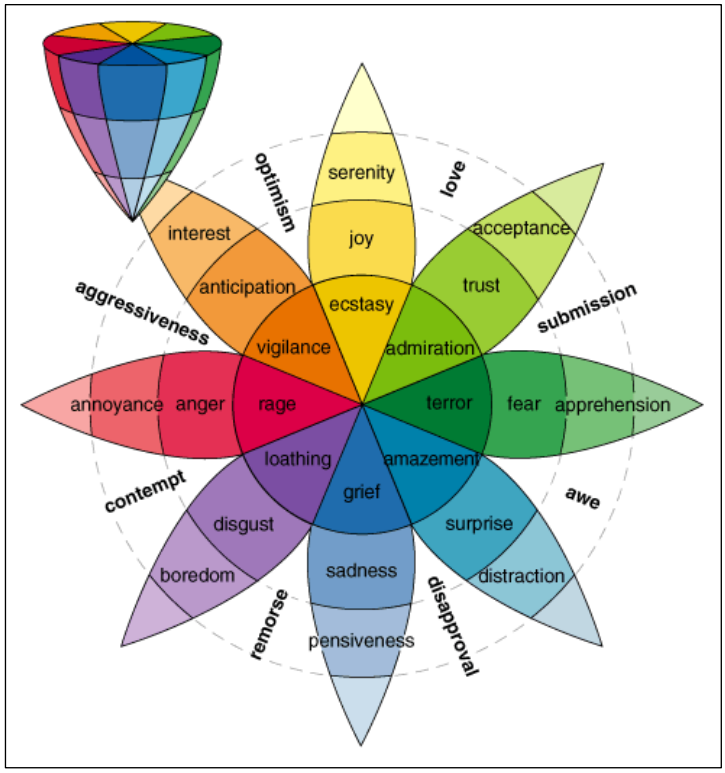
Podemos facilmente lembrar de muitas palavras que representam diferentes emoções. Muitas vezes essas palavras possuem significado similar, mas são importantes para auxiliar a transmitir a intensidade que um interlocutor deseja transparecer. Por exemplo, nas palavras “bem”, “animado”, “alegre”, “contente”, “feliz” e “extasiado”, podemos perceber que todas elas remetem a um sentido positivo, diferenciando-se apenas na intensidade que representam.

Percebendo esse fenômeno, Plutchik (2001, p 349, tradução livre) resolveu combinar as diferentes intensidades de emoções de forma análoga às diferentes intensidades de cores:

> Em inglês, existem algumas centenas de palavras que remetem a emoção e elas podem ser agrupadas em famílias com base na semelhança. Descobri que as emoções primárias podem ser conceitualizadas de uma forma análoga a uma roda de cores [...]. Outras emoções são combinações das emoções primárias e podem ser agrupadas de acordo com a sua intensidade [...].

O objeto resultante do agrupamento de emoções em famílias de cores, desenvolvido por Plutchik, pode ser representado de forma tridimensional, onde, as emoções primárias formam a base do circunflexo e as emoções secundárias podem ser combinadas em diferentes intensidades. Há ainda uma divisão central de 180º separando as emoções predominantemente negativas das predominantemente positivas.

Figura 1 – Roda de Plutchik



Fonte: Plutchik (2001, p. 349).

O entendimento das emoções básicas e suas combinações são de extrema importância para a construção da polaridade emotiva expressa em textos da internet. A partir de palavras que representam emoções secundárias e que são maioria nos textos de internet, será possível descobrir a emoção primária que as compõem. Além disso, esta monografia terá seu estudo restringido as duas famílias de emoções que mais contribuem para a formação de comportamentos agressivos e de desprezo na internet, que são as emoções básicas da raiva e da repugnância. Na próxima seção veremos como essas emoções de polaridade negativa podem ser expressadas nos meios informativos, especificamente através de textos na internet.

2.1.1 Agressividade

A emoção por si só, é um conceito muito abstrato. Torna-se realidade apenas no cérebro do indivíduo que a sente. O próprio sentimento é complexo, uma vez que, todos os nossos sentimentos são recriados a partir de estímulos elétricos e reconhecimento de padrões por nosso cérebro. Esses padrões acionam outras funções específicas que desencadeiam reações químicas, provocando alterações em nosso organismo e gerando, por fim, os nossos comportamentos.

Staats (1994) afirma que o comportamento é afetado diretamente pela emoção sentida. Sendo ela positiva ou negativa, a emoção gera estímulos para um grande número de respostas corporais. As emoções apenas podem ser transmitidas e compreendidas através das expressões desses comportamentos.

Quando falamos das emoções básicas que representam a raiva e a repugnância, logo nos vem à mente o comportamento agressivo (ver a “Roda de Plutchik” na sessão 2.1). De fato, essas emoções estão diretamente ligadas a esse comportamento. Pessoas enraivecidas, por exemplo, são mais suscetíveis a ofender outras pessoas, transportando os sentimentos internos para o exterior na forma de comportamentos agressivos. Esses comportamentos podem ser representados de forma verbal, musical, gestual e também na forma textual. Na verdade, as formas textual e verbal são formas comuns encontradas pelos humanos para transportar a carga emotiva negativa para fora de nossas mentes:

> No contexto terapêutico é comum observar pessoas expostas a eventos aversivos relatarem estados emocionais negativos como consequência da experiência: se a pessoa é reforçada por falar ou escrever sobre sentimentos, disso resultará melhora dos seus estados emocionais negativos. Verbalizar ou escrever o que se sente são duas maneiras que uma pessoa encontra para lidar com os efeitos dos eventos aversivos (BRITTO; ELIAS, 2009, p 3).

Segundo Strongman (2003), a raiva é quase sempre categorizada como negativa. A razão mais provável para isso é porque ela é parte integral do comportamento hostil e violento. Mas do ponto de vista evolutivo, a raiva foi essencial

para garantir a sobrevivência de nossa espécie nos momentos de autodefesa e proteção de nossos semelhantes.

Podemos aplicar a lógica do autor nos dias atuais. Não há problema algum em sentir raiva. Ela é essencial para garantir que não nos acomodemos perante as injustiças que cercam nossas vidas. A emoção em si não é um problema, mas sim o comportamento gerado a partir dessa emoção que pode desencadear atitudes hostis. Quando essas atitudes são transmitidas e compreendidas por outras pessoas, há a possibilidade de que esse comportamento agressivo seja multiplicado de forma danosa à sociedade.

Na próxima seção veremos como os comportamentos agressivos podem ser transmitidos através de textos e interpretados pelas pessoas.

2.1.2 Emoções em textos

Há inúmeras formas de transmissão de comportamentos agressivos que geram emoções negativas, mas para esta monografia, utilizaremos como fonte de dados, exclusivamente, os textos criados por usuários das maiores redes sociais do mundo. São dois os motivos principais para seguir nesta abordagem: o primeiro é pelo fato da disponibilidade de informações, uma vez que muitas dessas postagens nas redes sociais são feitas de forma pública; a segunda é porque, como veremos nas próximas seções, os textos são formados por regras léxicas e sintáticas que podem ser observadas e processadas computacionalmente.

Mas antes disso, precisamos entender um pouco sobre como as emoções negativas são expressadas na internet. Precisamos primeiro identificar as diferenças entre os textos que apresentam xingamentos (conhecidos como “palavrões”) e aqueles que mesmo sem apresentar xingamentos, podem ser ofensivos.

Segundo a professora de filosofia na Universidade de Londres, Rebecca Roache (2016), pensar que um xingamento sempre representa uma linguagem rude ou ofensiva, pode ser muito genérico. Uma linguagem rude não precisa conter palavrões, como por exemplo, quando alguém faz uma piada de mau gosto sobre algo de ruim que aconteceu a outra pessoa, ou quando alguém desqualifica outra pessoa sem motivos aparentes, ou quando ofendemos alguém utilizando palavras mais “rebuscadas”. Seguindo o raciocínio da autora, podemos dar alguns exemplos de frases ofensivas que não apresentam necessariamente algum palavrão, como as frases “você é totalmente desprovido de inteligência!” ou “aqui dentro, asiático não entra!”.

Roache (2016) afirma que para algo ser ofensivo, precisa ofender alguém. Ou seja, precisa ser direcionado para algum indivíduo ou um grupo de pessoas que possam compreender o texto comunicado como ofensa. Por exemplo, a palavra “viado” (com “i” mesmo, diferente do animal veado) pode representar uma ofensa a um indivíduo em específico ou a todo um grupo de pessoas (no caso os homossexuais). Já a sentença “vá se foder!”, geralmente é direcionada a algum indivíduo em específico.

Na internet é comum vermos frases contendo palavrões, porém sem ofender alguém diretamente. A expressão “puta que pariu, meu time perdeu de novo”, mesmo contendo um xingamento explícito, representa acima de tudo uma sensação de indignação e pode ser interpretada como não ofensiva para muitas pessoas. Muitas dessas expressões são incorporadas como ditados populares ou expressões idiomáticas:

> É preciso enfatizar que a questão das EIs [expressões idiomáticas] nos remete ao domínio da norma e não da língua. Assim sendo, [os ditos populares] são aprendidos de cor como se aprende o vocabulário do idioma e eles fazem parte do acervo da cultura e não do sistema linguístico. Por outro lado, sabemos que estas expressões vão sendo armazenadas na memória individual e na memória coletiva e passam a fazer parte do léxico da língua. (...) Cada termo deste sintagma não conserva sua identidade própria e se torna assim não-analisável. Por essa razão, os constituintes de uma EI como essa se tornam indissociáveis, não permitindo a supressão ou acréscimo de um elemento. As EIs são típicas de uma nação e enraizadas na sua cultura” (BIDERMAN, 2005, p. 56).

Considerando todo esse volume de variáveis que pode influenciar na interpretação dos textos da internet, uma vez que estão repletos de expressões

idiomáticas, xingamentos, palavrões e contextos, tem-se um grande desafio em compreender tais comportamentos agressivos de forma computacional. Nas próximas seções, buscaremos entender de que forma a computação pode ser direcionada para resolver esse tipo de problema.

2.2 Computação afetiva

O termo “Computação Afetiva” surgiu em 1997 com a publicação do livro “*Affective Computing*” de Rosalind Picard. Conforme a própria autora descreve em seu livro, na época em que o termo foi cunhado, a Computação Afetiva limitava-se ao reconhecimento das experiências emotivas que os usuários presenciavam ao interagir com sistemas (tanto sistemas em *software* quanto em *hardware*). Após o reconhecimento dessas emoções, os sistemas deveriam então ser remodelados em busca de melhoria nessas experiências. Até então, a computação não possuía avanços suficientes para gerar esse conhecimento de forma totalmente autônoma.

A partir dessa primeira tentativa de buscar compreender a relação de uso de sistemas com as experiências emotivas, outras ideias foram se fortalecendo, muitas delas provenientes das áreas de Inteligência Artificial. Segundo os autores Costa e Macedo (2012), a Computação Afetiva está tentando desenvolver computadores parecidos com humanos e com capacidades de observar, interpretar e gerar características afetivas. Essa “harmonia” entre humanos e computadores pode melhorar a qualidade da comunicação homem-máquina e também, melhorar a inteligência dos sistemas.

Considerando a opinião dos autores, podemos destacar o quanto é relevante para nós humanos, nos comunicarmos com algo semelhante a nós, algo que seja familiar. As características inteligentes dos sistemas computacionais não podem ser exageradamente eficientes, pois se fossem, gerariam um desconforto ou até medo nos usuários. Ou seja, a “inteligência” nesse caso, pode ser considerada como algo

“parecido com um humano”. Quando falamos com uma secretaria eletrônica, um assistente pessoal ou um *chatterbot*, queremos nos identificar humanamente com quem estamos conversando.

Nas próximas seções, serão abordados os mecanismos e tecnologias envolvidas no reconhecimento desses padrões humanos, especificamente nas demonstrações afetivas de polaridade negativa trafegadas nas redes sociais e consumidas por usuários da internet.

2.2.1 Inteligência artificial

A Inteligência Artificial (IA) é um campo da computação bastante recente, sendo que o termo foi cunhado em 1956 por John McCarthy (RUSSELL; NORVIG; 2013). Mas antes disso já haviam pensamentos elaborados e complexos a respeito da relação entre computação e inteligência. O maior nome da época, sem dúvida, é o tão conhecido Alan Turing e seu trabalho publicado em 1950 chamado *Computing Machiney and Intelligence*.

Turing (1950) não cunhou o termo referente a IA, mas desenvolveu o pensamento que chamou de “Teste da Imitação”, onde ele considerou que se uma pessoa normal não fosse capaz de identificar se está conversando com uma outra pessoa real ou uma máquina, essa máquina pode ser considerada como inteligente. Mais tarde a literatura atribuiu a esse teste o nome de Teste de Turing e passaram-se quase sessenta e cinco anos para que o primeiro computador a passar no teste fosse desenvolvido. Apenas recentemente e em meio a muitas controvérsias, o primeiro computador a passar no teste de Turing é conhecido como Eugene Goostman. Foi desenvolvido por uma equipe ucraniana e apresentado durante a *Turing Test 2014*, na University of Reading (UNIVERSITY OF READING, 2014).

Considerando as definições de inteligência computacional descritas por Turing, podemos afirmar que um computador não precisa ser necessariamente inteligente, basta apenas que ele apresente um comportamento que simule inteligência. Esse questionamento foi recentemente levado à mídia através da série de TV *Westword* (HBO, 2016), onde muitas máquinas foram criadas para simular pessoas, seus comportamentos e suas emoções. A série levanta o mesmo questionamento: existe inteligência nata ou a inteligência é apenas um grupo de comportamentos que parecem inteligentes?

Essa discussão envolve muitas áreas além da computação, como a biologia e a filosofia, porém, ela é de extrema importância para o desenvolvimento do tema desta monografia, afinal, será preciso identificar os comportamentos provenientes de emoções dos usuários da internet. Segundo Coppin (2004), "ao estudar Inteligência Artificial é útil ter um entendimento dos fundamentos de diversas outras áreas, principalmente filosofia, linguística, psicologia e biologia". E prossegue com a sua definição:

> Inteligência Artificial é o estudo dos sistemas que agem de um modo que a um observador qualquer pareceria inteligente. [...] envolve utilizar métodos baseados no comportamento inteligente de humanos e outros animas para solucionar problemas complexos (COPPIN, 2010, p. 4).

O campo da IA que abrange o estudo de elementos que podem ser analisados e utilizados para "ensinar" um computador a ter comportamentos inteligentes, é conhecido como Aprendizado de Máquina[1]. Mecanismos de aprendizado de máquina estão intensivamente presentes no dia-a-dia de quem navega na internet:

> Talvez você não saiba, mas o machine learning está em todos os locais ao seu redor. Quando digitamos uma consulta em um mecanismo de busca, é dessa forma que o mecanismo define os resultados que deve exibir (e também os anúncios). Quando lemos e-mails, não vemos grande parte do spam, porque o machine learning desconsidera essas mensagens. Quando acessamos a Amazon.com para comprar um livro ou a Netflix para assistir a um vídeo, um sistema de machine learning recomenda outros que possam nos interessar. (DOMINGOS, 2015, p. 30).

[1] Do inglês não literal *Machine Learning*.

Para que um computador realmente consiga “aprender” sobre algo, precisamos fornecer a ele um grande volume de conhecimento. Ou seja, é necessário efetuar treinamentos repetitivos, com informações úteis que poderão ser utilizadas na tomada de decisão de forma automática. Considerando os objetivos desta monografia, precisaremos treinar os algoritmos de IA fornecendo a ele informações e variáveis relevantes que o auxiliem a decidir se um determinado texto contém ou não, um comportamento emotivo de polaridade negativa.

Há vários modelos de aprendizado de máquina recomendados de acordo com o tipo de problema a ser resolvido. Dentro do escopo desta monografia, focaremos na descrição de duas formas de aprendizagem específicas: a Aprendizagem Supervisionada e a Aprendizagem Semissupervisionada.

Segundo Russell e Norvig (2013), na aprendizagem supervisionada, são fornecidos aos algoritmos de aprendizagem, vários grupos de informações contendo as entradas e as saídas resultantes dessas entradas. Com base no exemplo e na detecção de características salientes, o computador começa a obter um determinado conhecimento, fazendo com que aos poucos, ele seja capaz de prever as possíveis saídas de acordo com as entradas. Ou seja, se por exemplo informarmos ao algoritmo uma série de frases como entrada e para cada uma delas informarmos se trata ou não de uma emoção agressiva, ele terá condições de avaliar frases futuras sem a necessidade de fornecer as saídas.

Mas as entradas de informações nem sempre são consistentes. Por isso, para abranger o tema proposto, precisaremos considerar juntamente, o método de aprendizagem semissupervisionado. Russell e Norvig (2013) descrevem que o método semissupervisionado deve ser utilizado quando não se tem certeza dos valores de saída para cada entrada fornecida como conjunto de dados de aprendizado. Como temos por objetivo a detecção de emoções em textos extraídos da internet, a aprendizagem estará sujeita a vários ruídos nos dados de treino, uma vez que a linguagem é repleta de termos ambíguos, palavras podem possuir interpretações diferentes (veja a seção 2.1.2), expressões sofrem impactos culturais, entre outros

problemas de imprecisão na definição das saídas. Há uma série de tratamentos e pré-processamentos que deverão ser efetuados para amenizar a incidência de ruídos nos dados textuais. As próximas seções abordarão alguns desses mecanismos.

2.2.2 Processamento de Linguagem Natural

Desde os primeiros computadores digitais introduzidos em nossa cultura a partir dos anos 40, os computadores vêm trazendo avanços e facilidades em praticamente todos os setores socioeconômicos. Ao contrário do que muitos pensam, a computação não fica restrita somente à área de informática, mas sim, serve como uma ferramenta que potencializa a execução de tarefas em qualquer outro setor.

A Fundação Getulio Vargas (FGV, 2017) realizou uma pesquisa de escopo nacional e concluiu que “para cada 1% a mais de gasto e investimento em TI, depois de 2 anos, o lucro aumentou 7%”. Ou seja, mais do que uma questão tecnológica, a utilização dos computadores é uma questão estratégica.

Entendendo que o computador faz parte das nossas vidas, o homem vem tentando aprimorar o modo como se comunica com as máquinas para instruí-las adequadamente a respeito das tarefas que devem ser executadas. Com o surgimento das primeiras linguagens de computador, foi possível instruir os computadores (através dos programas) a desenvolverem novas formas de comunicação com elas mesmas, com linguagens de alto nível, cujos programas resultam em linguagens intermediárias, que por fim, resultam em linguagens de baixo nível que realmente executam as instruções a nível de máquina.

Todas essas “camadas” de linguagens têm por objetivo “naturalizar” o processo de conversa homem-máquina e, consequentemente, facilitá-lo. O Processamento de Linguagem Natural (PLN) está no topo dessas camadas de linguagens, pois tem como

objetivo, permitir a comunicação entre homem e máquina da forma mais natural possível.

> Com efeito, essa preocupação com a comunicação “mais natural” entre o homem e a máquina já se instalava, desde o momento da própria criação dos primeiros computadores. As preocupações, porém, foram muito mais além. Por que não ousar? Por que não criar meios que instruam o computador a “traduzir” frases e textos de uma língua para a outra? Questões como essas motivaram os pesquisadores a investigar o processamento automático das línguas naturais (PLN) (NUNES *et al*., 1999, p. 6).

Porém, há uma enorme diferença entre as linguagens de programação utilizadas para instruir máquinas a realizar tarefas e as linguagens naturais que costumamos utilizar para nos comunicar com seres da nossa espécie. Segundo Coppin (2010), enquanto que as linguagens formais (Java, C, R, Python, etc.) são altamente descritivas, seguem à risca suas regras sintáticas e léxicas e não permitem interpretações diferentes, as linguagens naturais podem ser altamente ambíguas, ou seja, uma mesma sentença pode ter vários significados e esses significados podem ser bastante difíceis de identificar.

Até para nós humanos pode ser complicado identificar um sentido de ironia em uma frase qualquer. Imagine uma senhora conversando com seu filho que acabou de quebrar o pote de biscoitos, quando ela fala “que bonito, hein?”. Afinal, o que essa frase significa do ponto de vista computacional? Será possível que um computador consiga processar adequadamente sentenças ambíguas como esta? Se esta mesma sentença fosse representada com uma linguagem formal e altamente descritiva, ela poderia ser reescrita como “eu não gostei que o meu filho tenha quebrado o pote de biscoitos”.

Coppin (2010) finaliza sua ideia afirmando que “linguagens formais são quase sempre projetadas para garantir que não ocorra ambiguidade”. Ou seja, se você escrever um programa na linguagem C#, todos os computadores capazes de reproduzir o seu programa, deverão apresentar o mesmo comportamento. Isso é altamente desejável, pois do contrário, o computador poderia tomar uma decisão errada e interpretar o seu programa de forma indesejada. Mas seria possível instruir um computador a realizar alguma tarefa, utilizando uma linguagem informal? Qual o estado da arte atual para o tratamento de textos e inclusive, o entendimento de emoções?

Para o alcance dos objetivos desta monografia e uma boa exploração do tema proposto, será necessário aplicar mecanismos de pré-processamento nos textos informais obtidos através da internet. O objetivo desses procedimentos é tentar "formalizar" os trechos de texto de modo a torná-los menos ambíguos. Nas próximas seções, abordaremos de forma sintética, os principais conceitos para a decomposição de textos informais em textos mais informativos e de processamento mais eficiente.

2.2.3 Análise Morfológica

O ponto de partida para a análise de textos de forma computacional se dá pelo desmembramento de sentenças em objetos menores que possam ser analisados individualmente. Grande parte das linguagens utiliza um símbolo comum para efetuar a separação dos diferentes *tokens*[2] que compõem uma sentença textual. É o caso do português e da maioria dos idiomas contemporâneos, que possuem como símbolo separador o "espaço" e cada *token* isolado é conhecido como "palavra". Petter (2003, p. 59), descreve, do ponto de vista linguístico, que "a palavra é identificada como uma unidade formal da linguagem, que, sozinha ou associada a outras, pode constituir um enunciado".

O estudo da formação dessas palavras assim como as suas variações, é conhecido como morfologia.

> "Morfologia. Este é o primeiro estágio da análise que é aplicado a palavras, uma vez que tenham sido identificadas a partir da linguagem ou inseridas no sistema. Morfologia examina os modos pelos quais palavras se desmembram em componentes e como isto afeta o *status* gramatical delas" (COPPIN, 2010, p. 497).

Tomemos por exemplo a palavra "forma" e algumas de suas variações morfológicas:

[2] O termo *token* é frequentemente empregado na computação, para representar um fragmento de uma estrutura maior, como as palavras de um texto (IBM, 2013).

Quadro 1 – Morfologia da palavra primitiva “forma”

Forma
Re**forma**
Dis**forma**
Trans**forma**
Con**forma**
In**forma**

Fonte: Do Autor.

Repare que o sentido da palavra é alterado de acordo com a utilização de alguns prefixos ou sufixos, gerenciados por regras gramaticais. Podemos facilmente decompor essas palavras em mais de um *token* para facilitar nossa compreensão, como por exemplo, a palavra “disforma”, onde o prefixo “dis” representa um sentido de negação, aliado ao substantivo “forma”. A resultante, é algo como “não forma”.

A morfologia das palavras pode ir além da formação de palavras derivadas dos mesmos substantivos primitivos. As variações podem também indicar tempos verbais, modos, gerúndios, gêneros, etc. (MARGOTTY *et al*., 2011). Podemos utilizar como exemplo a palavra “transformar” e algumas de suas variações morfológicas:

Quadro 2 – Morfologia da palavra “transformar”

Transformar
Transformado
Transformei
Transformação
Transforma
Transformando

Fonte: Do Autor.

Todas essas permutações são compreendidas sistematicamente em nossos cérebros, porém requerem um grande processamento computacional para serem entendidos e representados digitalmente. Ao mesmo tempo, as variações morfológicas das palavras são essenciais para que um computador possa compreender o sentido de

um texto e mais ainda, no caso do processamento de emoções. Como descrito na seção introdutória desta monografia, um computador tem extrema dificuldade em avaliar palavras dentro de um contexto, pois o próprio contexto é algo além da realidade compreendida por uma máquina.

Para operacionalizar esse processo, as palavras devem ser pré-processadas individualmente, a fim de detectar suas origens morfológicas independentemente de quaisquer contextos que a frase como um todo possa apresentar. O objetivo desse processamento é definir a classe gramatical de cada uma das palavras presentes no texto alvo de análise, antes de avaliar a relação com as demais. Esta monografia utilizará como base, o sistema de classificação PoS (*Part of Speech*) que adequa as palavras de acordo com a tabela a seguir (NYU, 2017):

Tabela 1 – Classes de palavras no padrão PoS

Tag	Class Name
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol

TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Fonte: Adaptado pelo autor com base em NYU (2017).

Após esse processo de classificação em cada uma das palavras dos textos, partiremos para a análise sintática, que conforme Coppin (2010, p. 499), “envolve mapear um fragmento linear de texto em uma hierarquia que represente o modo como as várias palavras interagem entre si sintaticamente”.

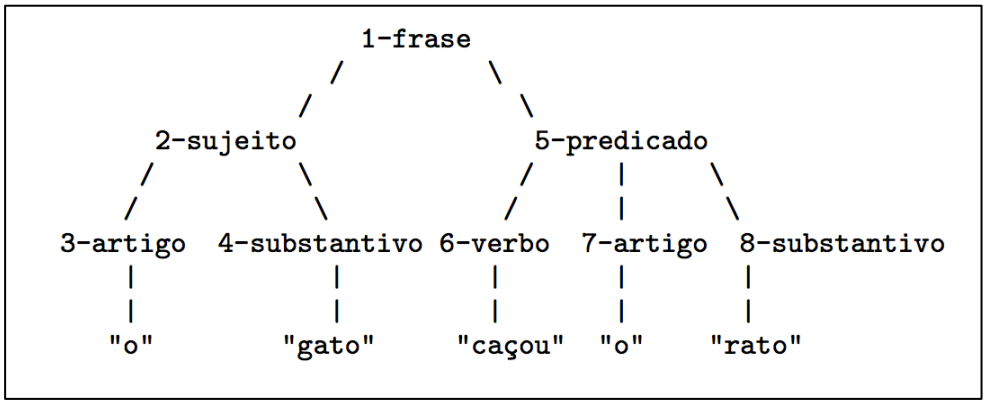
2.2.4 Análise Sintática

Conforme Russel e Norvig (2013, p. 777), “a Análise Sintática é o processo de analisar uma cadeia de palavras para descobrir a sua estrutura frasal, de acordo com as regras de uma gramática”. Após um processo de classificação individual das palavras de uma sentença de acordo com o padrão PoS (conforme descrito na seção 2.2.3), é possível construir uma árvore sintática que permite a análise da relação entre cada uma dessas palavras.

Tomemos por exemplo a frase: “o gato caçou o rato”, composta pelo sujeito “o gato” e pelo predicado “caçou o rato”. Podemos ainda classificar cada uma das

palavras conforme o padrão PoS[3], resultando em uma estrutura do tipo *top-down*, conforme a figura 2.

Figura 2 – Exemplo de Árvore Sintática



Fonte: Pereira (2013).

Segundo Coppin (2004), a análise sintática pode ser realizada computacionalmente seguindo uma série de regras e autômatos de transição baseados na própria gramática utilizada. Não é objetivo desta monografia, abordar detalhadamente cada um desses autômatos, mas é possível resumir as regras mais importantes da seguinte forma:

- Frase = Locução Nominal (LN) + Locução Verbal (LV)
- LN = Artigo + Substantivo + Adjetivo
- LV = Verbo + LN

A análise sintática combinada com a análise morfológica será de extrema importância para a abordagem segura do tema proposto nesta monografia. Isolar as frases de um determinado texto para então aplicar o algoritmo de descoberta das classes morfológicas é essencial para que haja uma limitação no processamento e treinamento, considerando apenas palavras classificadas como relevantes à descoberta do sentido emotivo da sentença.

[3] Apesar do padrão PoS apresentado na seção 2.2.3 seguir a gramática em inglês, é possível realizar a mesma classificação morfológica para o português, pois ambos os idiomas possuem características sintáticas e classificações similares.

É importante ressaltar que a estrutura da frase auxilia a tornar mais eficiente o processamento computacional de textos, porém, ainda é necessário gerar um conhecimento sobre a informação processada e entender realmente o significado passado. Essa interpretação é realizada através das análises semântica e pragmática que serão os assuntos da próxima seção.

2.2.5 Análise Semântica e Pragmática

O Processamento de Linguagem Natural em um sistema computacional é muito eficiente quando seu escopo se limita às análises fonológicas, morfológicas e sintáticas. Isso ocorre pois todas essas análises podem ser descritas na forma de um algoritmo executável por qualquer tipo de computador. Trata-se de uma “receita de bolo” que recebe entradas esperadas, manipula dos dados segundo um conjunto de regras e gramáticas, gerando uma saída processada. O principal desafio atual do PLN está nas análises Semântica e Pragmática.

Conforme Coppin (2004), a análise semântica envolve o processamento do significado das palavras e do sentido de uma frase. Frases como “eu como comida com faca e panela”, podem ser analisadas morfologicamente e sintaticamente, mas mesmo assim, não transmitir nenhum sentido. Do mesmo modo, é objetivo do PLN que um computador consiga entender sentenças com sintaxe incorreta, mas que tenham algum sentido ou informação útil.

O autor descreve a análise pragmática como a capacidade de um computador entender frases que podem oferecer mais de um significado ou que não são diretamente claras. Frases que envolvem ironia, metáfora, perguntas retóricas ou que envolvem respostas diferentes das solicitadas. O autor utiliza como exemplo a pergunta: “pode me dizer a hora?”, onde um computador deveria compreender que a resposta não deve ser “sim”, mas que se espera uma resposta com o horário atual.

Para resolver essas questões, é preciso que o computador realize um processo conhecido como Desambiguação, que “é o processo de recuperar o significado mais provável de um enunciado” (RUSSEL; NORVIG, 2013, p. 790). Infelizmente não há “receita” para esse processo. Um sistema computacional atualmente só é capaz de interpretar os sentidos semânticos e pragmáticos com base no seu conhecimento adquirido a partir do aprendizado de máquina. Tendo esse conhecimento em mãos, é possível aplicar algoritmos probabilísticos para tentar “adivinhar” qual o provável sentido de uma sentença (vide seção 2.2.1).

As próximas seções apresentam um resumo dos principais mecanismos para a classificação dos textos com base em seu provável significado, utilizando de algoritmos probabilísticos para gerar aprendizado de máquina.

2.3 Classificação de textos

A classificação de textos, quando realizada por um computador e utilizando os processos de PLN, consiste em detectar a categoria a qual pertence determinado texto com base nas características observadas e na probabilidade dessas características pertencerem a determinadas classes pré-definidas (RUSSEL; NORVIG, 2013).

A classificação de textos envolve principalmente os elementos estatísticos das palavras que os compõem. Um exemplo desse mecanismo pode ser encontrado nos serviços *antispam* de e-mail, onde a presença de termos como “preço baixo” ou “promoção incrível” indica uma probabilidade maior de que o e-mail possa ser um *spam* (HASTIE *et al*., 2008).

Há uma série de algoritmos que podem ser utilizados para a categorização de textos com base na probabilidade de suas características. Cada um deles pode apresentar diferentes taxas de eficácia, que variam de acordo com o tipo de dado processado e com a qualidade das informações de treinamento. Nesta monografia

serão abordados dois algoritmos amplamente empregados dentro da área de PLN: o *Naive Bayes* e o Regressão Logística. As seções a seguir detalham as principais características de cada um.

2.3.1 Algoritmo Naive Bayes

Quando falamos que há uma probabilidade de que algum evento aconteça, podemos afirmar de que há uma incerteza de que esse mesmo evento ocorra. Thomas Bayes, um pastor nascido em Londres em 1971, foi quem deu os primeiros passos ao estudo probabilístico, descrevendo que a probabilidade de eventos não conectados ocorram é menor do que a probabilidade de eventos conectados ocorram (MLODINOW, 2009, p. 1967). O autor resume sua ideia descrevendo:

> "A Teoria de Bayes trata essencialmente do que ocorre com a probabilidade de ocorrência de um evento *se* outros eventos ocorrerem, ou *dado* que ocorram" (MLODINOW, 2009, p. 1967).

É, portanto relevante, considerarmos tal teoria nesta monografia, dado que a probabilidade de que um texto possua emoções negativas pode aumentar quando ele apresenta palavras que geralmente indicam esse tipo de emoção. Podemos citar um exemplo com a frase: "eu odeio este lugar de merda", que possui as características "odeio" e "merda" que aumentam substancialmente a probabilidade de que essa frase contenha emoções negativas.

Cada uma das palavras é avaliada de forma binária, sendo "Sim" quando estiver presente e "Não" quando não estiver presente no texto avaliado. Essa assinatura binária é atrelada a classe emotiva geral da sentença, gerando um critério específico para essa combinação de palavras. Quanto mais combinações binárias forem compostas, maior a diversidade dos critérios.

Podemos facilmente exemplificar o funcionamento do algoritmo de Naive Bayes, utilizando as frases do Quadro 3, comparando-as com o conjunto de treinamento

supervisionado da Tabela 2. Repare que para otimizar o processamento, algumas palavras que não possuem alta influência no sentido da frase, como as preposições, artigos e pronomes possessivos, são desconsiderados.

Quadro 3 – Frases a serem classificadas

Frase 1 – “Eu odeio jogar futebol”
Frase 2 – “Não se deve falar merda”
Frase 3 – “Eu odeio você seu merda”

Fonte: Do autor.

Tabela 2 – Critérios para treinamento supervisionado

Critério	**Possui a palavra “odeio”?**	**Possui a palavra “merda”?**	**Possui a palavra “você”?**	**Classe Emotiva**
1	Sim	Não	Não	Não Negativa
2	Não	Sim	Não	Não Negativa
3	Sim	Sim	Não	Negativa
4	Sim	Não	Sim	Não Negativa
5	Não	Sim	Sim	Negativa
6	Sim	Sim	Sim	Negativa

Fonte: Do Autor.

Esses critérios deverão ser utilizados para efetuar o treinamento do algoritmo de classificação, que considerará a probabilidade de existência, ou não, de determinada palavra e a possível classificação emotiva. A Tabela 3 detalha a probabilidade de que uma determinada classe emotiva contenha, ou não, as palavras-chave escolhidas como critério de treinamento.

Tabela 3 – Distribuição de probabilidades

Classe Emotiva	**Possui a palavra "odeio"?**		**Possui a palavra "merda"?**		**Possui a palavra "você"?**	
	Sim 4/6	Não 2/6	Sim 4/6	Não 2/6	Sim 3/6	Não 3/6
Não Negativa (NN) 3/6	2/4	1/2	1/4	2/2	1/3	2/3
Negativa (N) 3/6	2/4	1/2	3/4	-	2/3	1/3

Fonte: Do Autor.

Uma vez descobertas, as probabilidades individuais do campo amostral, é possível aplicar o Teorema de Bayes, que conforme Coppin (2010, p. 497), pode ser declarado do seguinte modo:

$$P(c|X) = \frac{P(\mathrm{X}|c) \times P(c)}{P(X)}.$$

Onde P(c | x) representa a probabilidade de que uma classe *c* ocorra devido a ocorrência de uma palavra *x*.

Segundo Sayad (2017A), o Teorema de Bayes pode ser estendido para expressar uma probabilidade condicionada envolvendo mais de duas variáveis, utilizando o formato normalizado declarado como:

$$P(c|X) = P(x_1|c) \times P(x_2\,|c) \times \ldots \times P(x_n|c) \times P(c)$$

A partir da tabela de distribuição das probabilidades e do Teorema de Bayes, é possível descobrir a probabilidade P de que determinada frase pertença a uma classe emotiva negativa ou não, multiplicando as probabilidades individuais das palavras-chave selecionadas para cada classe emotiva. Vejamos os resultados probabilísticos, utilizando como exemplo as frases do Quadro 3, e aplicando a lógica *bayesiana* sobre cada uma das assinaturas binárias, avaliando a probabilidade individual de cada um dos critérios.

Tabela 4 – Calculando a probabilidade de cada classe

Frase	**P(Não Negativa)**	**P(Negativa)**
1 – “Eu **odeio** jogar futebol”	P = 2/4 * 2/2 * 2/3 * 3/6 = 0,166 (66,6%)	P = 3/6 * 2/4 * 1/3 = 0,083 (33,3%)
2 – “**Você** não deve falar **merda**”	P = 1/2 * 1/4 * 1/3 * 3/6 = 0,02 (14,2%)	P = 3/6 * 1/2 * 3/4 * 2/3 = 0,125 (85,8%)
3 – “Eu **odeio você** seu **merda**”	P = 2/4 * 1/4 * 1/3 * 3/6 = 0,02 (14,2%)	P = 3/6 * 2/4 * 3/4 * 2/3 = 0,125 (85,8%)

Fonte: Do Autor.

No exemplo supracitado podemos perceber, principalmente na frase 2, que o algoritmo de classificação Bayesiana pode apresentar resultados questionáveis, principalmente porque há uma dificuldade computacional em compreender o contexto das palavras e não apenas o seu significado. A quantidade e a qualidade do grupo de treinamento também influenciam diretamente na eficiência de classificação obtida.

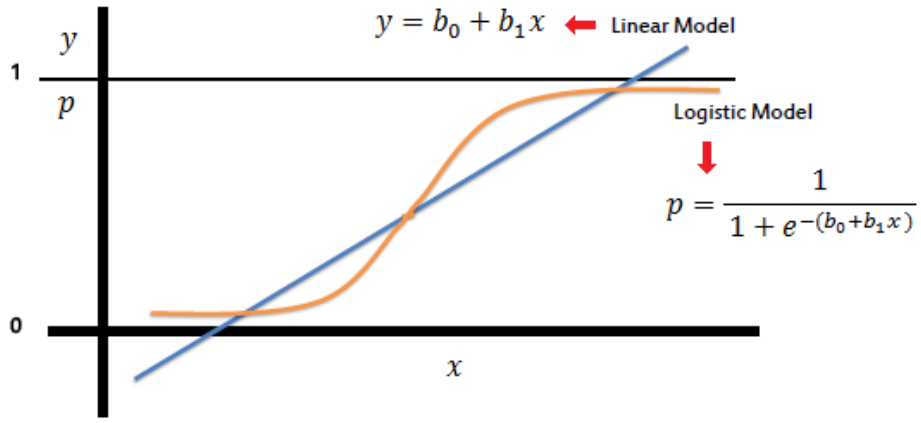
2.3.2 Algoritmo Regressão Logística

A Regressão Logística permite a identificação probabilística de que uma série de variáveis é responsável por um resultado ou outro (SAYAD, 2017B). Aplicando ao tema desta monografia, ela permite identificar se determinado texto contém emoções negativas (com resultado tendendo a 1) ou se não contém emoções negativas (com resultado tendendo a 0).

O autor prossegue sua definição da Regressão Logística, comparando-a com o formato linear, afirmando que a regressão linear não é recomendada para a análise de

soluções binárias, pois o cálculo pode gerar resultados fora da zona aceitável (que deve permanecer entre 0 e 1). A Figura 3 demonstra graficamente a relação entre o modelo linear e o modelo logístico.

Figura 3 – Comparação entre regressão linear e logística



Fonte: SAYAD (2017B).

Um dos componentes da Regressão Logística é conhecido como *odds,* que é a chance c de que algum evento ocorra de acordo com a probabilidade p desse mesmo evento ocorrer quando há a ocorrência de determinada variável. Ele pode ser obtido a partir da relação (FULTON *et al*., 2012):

$$c = \mathrm{b}_0 + \mathrm{b}_1 x = \frac{\mathrm{p}}{1 - \mathrm{p}}$$

Considerando ainda os dados da Tabela 3, podemos concluir, por exemplo, que há 2 vezes mais chance de um texto não conter emoções negativas quando a palavra “você” está presente (c = 2/3 / (1 – 2/3) = 2). Quando há mais de uma variável envolvida, é necessário efetuar uma combinação de cálculos, característicos da regressão logística (SAYAD 2017B):

$$p = \frac{1}{1 + e^{-(\mathrm{b}_0 + \mathrm{b}_1 \mathrm{x}_1 + \mathrm{b}_\mathrm{n} \mathrm{x}_\mathrm{n})}}$$

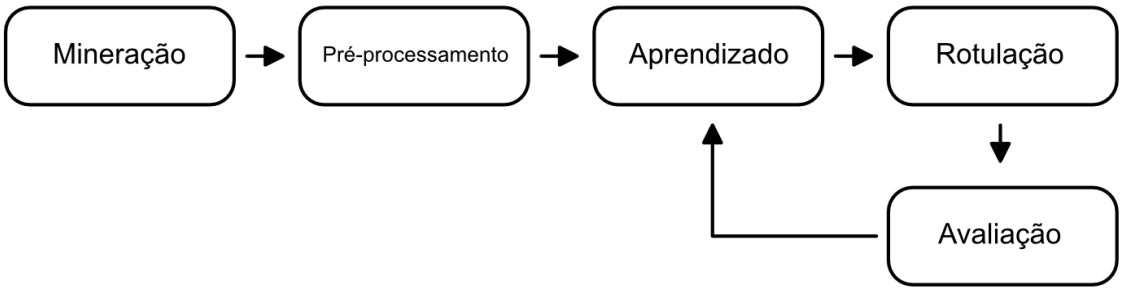
Agora conhecidos os algoritmos e dados envolvidos, é possível efetuar um estudo prático para a abordagem do tema proposto. O próximo capítulo contempla as etapas metodológicas envolvidas para a concepção de um experimento prático capaz de identificar o sentido emotivo dos textos informados pelos usuários.

3 METODOLOGIA

Atuar de forma computacional sobre informações textuais, inseridas por humanos e em um contexto social, para assim obter uma classificação do sentido emotivo desse conteúdo, requer uma série de passos intermediários. Cada um desses passos deve ser executado de forma linear, pois são totalmente interdependentes.

Na aplicação dos conceitos teóricos, para o atingimento dos objetivos propostos, serão necessárias 5 etapas: mineração, pré-processamento, aprendizado, rotulação e avaliação. A sequência e relação entre essas etapas pode ser visualizada na Figura 4.

Figura 4 – Etapas metodológicas



Fonte: Do Autor.

Há uma diferença entre dado, informação e conhecimento. O dado é um fato ou um valor documentado que precisa ser coletado de forma bruta através do processo de mineração. Quando o dado é pré-processado e a ele é atribuído um significado (através

das análises morfológica, sintática, semântica e pragmática) passamos a ter informação. Com a informação, é possível aplicar algoritmos específicos gerando o aprendizado de máquina necessário para efetuar a rotulação dos dados previamente coletados (DA SILVA *et al*., 2016).

Cada uma dessas etapas requer a utilização de técnicas específicas e um desenvolvimento separado. As próximas seções detalham os processos e recursos utilizados em cada uma delas.

3.1 Mineração

A Mineração de dados é essencial para gerar o conhecimento necessário para a utilização do PLN nesta monografia:

> A partir do estudo e da mineração dos dados, a descoberta acontece, e então novo conhecimento é produzido, contribuindo para a melhoria de produtos, sistemas, processos, negócios etc. Mineração de dados é, portanto, o meio pelo qual o fenômeno da descoberta se manifesta [...] (DA SILVA, 2016, p. 3).

O autor prossegue afirmando que a “mineração de dados é definida em termos de esforços para a descoberta de padrões em bases de dados“. Ou seja, dada uma fonte de dados muitas vezes de tamanho gigantesco, deve-se coletar aqueles que podem gerar uma informação qualidade.

Felizmente, as redes sociais dispõem de dados textuais suficientes para serem coletados e processados. Para a aplicação prática e validação dos objetivos propostos nesta monografia, a rede social escolhida foi o Twitter, por uma série de motivos. Primeiramente, devido ao volume de dados disponíveis, já que o Twitter registra mais de quinhentos milhões de *tweets* todos os dias (TWITTER, 2017A). Segundo, devido à limitação de 140 caracteres por *tweet,* que faz com que os textos se mantenham pequenos e diversificados. E por fim, devido ao fato de que o Twitter é uma plataforma amplamente utilizada para a expressão de atos de ódio e desprezo a outras pessoas ou entidades.

O processo de mineração se dará através da coleta de *tweets* a partir de uma API desenvolvida pelo próprio Twitter (TWITTER, 2017B). A partir de uma série de parâmetros especificados pela própria API, objetiva-se coletar um mínimo de trinta mil *tweets* contendo conteúdo agressivo. Para refinar a pesquisa, a mineração considerará apenas os *tweets* contendo ao menos uma das seguintes palavras: “filho da puta”, “caralho”, “merda” “bosta” e “porra”.

Será importante também coletar um segundo grupo de *tweets* que servirá como grupo de contraponto. Esse grupo será responsável por calibrar os algoritmos de treinamento com emoções neutras e positivas para que ele possa diferenciá-las das emoções negativas. A pesquisa desse grupo será focada em *tweets* que contenham uma ou mais palavras com sentido emotivo neutro ou positivo como “alegria”, “feliz”, “trabalho”, “amigo” e outras similares.

Os dados brutos minerados serão armazenados em uma base de dados relacional, para que assim, fiquem disponíveis para o pré-processamento, que será o assunto abordado na próxima seção.

3.2 Pré-processamento

O pré-processamento é a etapa onde os dados previamente minerados e armazenados, são preparados e otimizados para o aprendizado. Ou seja, esta etapa tentará transformar os dados em informações relevantes à geração de conhecimento. Todos os procedimentos descritos a seguir serão aplicados a todos os conjuntos de dados minerados no processo de mineração.

O estado da arte atual conta com algumas ferramentas de PLN que facilitam o pré-processamento, treinamento e classificação de textos. Uma das mais conhecidas ferramentas é o Natural Language Toolkit (NLTK, 2017) que contém uma ampla gama de bibliotecas para a manipulação de informações textuais, desenvolvidas na

linguagem Python. Apesar de possuir uma quantidade inferior de recursos, a ferramenta escolhida para o desenvolvimento da aplicação prática desta monografia foi a NaturalNode (NATURALNODE, 2017), que oferece algumas soluções similares desenvolvidas para a linguagem Javascript.

A primeira etapa a ser realizada é conhecida como *tokenização*, que consiste em fragmentar uma sentença em um conjunto de *tokens* (palavras) separados por um elemento comum (caractere que representa o espaço). O resultado consiste em um vetor de *tokens* que poderão ser processados individualmente.

Cada *token* agora poderá ser submetido a um algoritmo de classificação PoS (vide seção 2.2.3). Com isso, é possível eliminar aquelas palavras que não são relevantes para o contexto emotivo[4], como os artigos, preposições e palavras de ligação como “e” e “ou”. Eliminar palavras é um procedimento importante para reduzir o tamanho da base de dados e facilitar a sua manipulação.

A próxima etapa consiste em realizar a descoberta do radical das palavras, removendo todas as alterações morfológicas. Esse processo é conhecido como *steeming* ou “reduzir até a raiz” (SOUZA, 2005). Após esse procedimento, as palavras “idiota” e “idiotice”, por exemplo, poderão ser processadas de forma similar.

3.3 Aprendizado

A etapa de aprendizado consiste em submeter o conjunto de dados pré-processados aos dois algoritmos descritos na seção 2.3 desta monografia: O Naive Bayes e o Regressão Logística. O método de aprendizado escolhido é o semissupervisionado, onde assumiremos indiscriminadamente que o grupo de *tweets* obtidos a partir das palavras-chave “filho da puta”, “caralho”, “merda” “bosta” e “porra”,

[4] O termo técnico para esse tipo de palavra é conhecido como *Stopwords* (AGÊNCIA MESTRE, 2017).

representa frases com sentido emotivo negativo. Os demais *tweets* minerados serão considerados indiscriminadamente como “não negativos”.

A biblioteca NaturalNode dispõe de mecanismos de treinamento voltados para ambos os algoritmos escolhidos nesta monografia. O formato de inserção e extração de dados é bastante similar e compatível com a metodologia aqui descrita.

3.4 Rotulação

Após o treinamento dos algoritmos, baseado nas informações mineradas diretamente da internet e inseridas de forma semissupervisionada, o sistema de rotulação deverá estar apto a identificar se determinado texto informado é considerado negativamente emotivo, ou não.

3.5 Avaliação

O processo de avaliação é bastante simples: dado que o sistema está treinado e teoricamente apto a realizar a rotulação, basta introduzir um conjunto teste de textos, cujo o contexto emotivo é conhecido. O sistema deverá rotular cada um dos textos como “Agressivo” ou “Não Agressivo”.

O conjunto teste será composto por trezentos textos curtos, sendo um terço dele totalmente agressivo, um terço levemente agressivo e o outro terço não agressivo. Espera-se pelo menos 80% de acurácia na rotulação dos textos totalmente agressivos e não agressivos. Espera-se também uma acurácia de pelo menos 65% dos textos levemente agressivos. Com esse nível de eficácia, será possível desenvolver o trabalho proposto na seção 4 com um alto grau de confiabilidade.

O desenvolvimento prático da solução de rotulação incluirá uma área de *feedback*, onde o usuário poderá informar se concorda com a rotulação informada pelo sistema ou não. Esses dados serão reintroduzidos na base de treinamento, completando o ciclo.

4 TRABALHO PROPOSTO

Este trabalho propõe o desenvolvimento de um software capaz de coletar automaticamente o texto inserido pelo usuário em seu navegador de internet, efetuar uma análise seguindo as metodologias de PLN descritas nesta monografia e, por fim, classificar o texto informado de acordo com o seu grau de agressividade. Para viabilizar o software, será necessário o desenvolvimento de um conjunto de micro-serviços com responsabilidades diferentes. Alguns desses serviços funcionam de forma ativa, já outros são consultados sob-demanda, totalizando quatro serviços desenvolvidos: a Aplicação Cliente, o Webserver, o Rotulador e o Minerador.

A Aplicação Cliente contempla a interface de usuário. A proposta aqui descrita, resulta em uma extensão para o navegador de internet Google Chrome (GOOGLE, 2017B) compatível com a maioria dos computadores. A partir da inserção de textos por parte do usuário em qualquer página da internet, a aplicação deve encaminhar requisições de rotulação dos textos informados. De acordo com o retorno obtido, a extensão reagirá visualmente ao usuário, indicando se o texto inserido é considerado agressivo ou não. O usuário poderá ainda avaliar a acurácia do algoritmo e esse retorno (*feedback)* será considerado nas próximas avaliações.

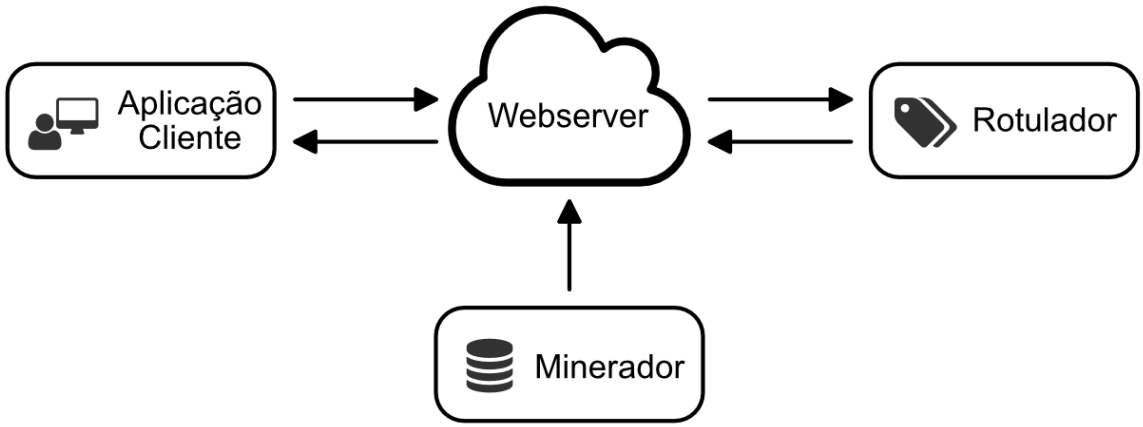
O Webserver é o serviço responsável por gerenciar as requisições provenientes da Aplicação Cliente, oferecendo o retorno adequado. Quando a requisição for referente a um pedido de rotulação de texto, ele deve reencaminhar à requisição ao

Rotulador. O Webserver também é responsável por encaminhar a informação de *feedback* do usuário para que possa ser novamente considerada nas próximas classificações de texto.

O Rotulador é responsável por atender as requisições provenientes do Webserver, classificando as informações passadas como “Agressivas” ou “Não Agressivas”, incluindo o percentual de confiabilidade na informação rotulada. É também no Rotulador que os algoritmos de classificação são treinados a partir da base de dados minerada e constantemente atualizada pelo serviço Minerador.

A Figura 5 demonstra a relação entre os serviços. As próximas seções detalharão o escopo do *software* como um todo, incluindo os requisitos a serem atendidos e os casos de uso atrelados às funcionalidades propostas.

Figura 5 – Arquitetura de serviços da Aplicação



Fonte: Do Autor.

4.1 Requisitos

De acordo com Silva e Videira (2001), requisitos de um software são um conjunto de especificações que relatam uma condição ou premissa que o mesmo deverá

atender. Ou seja, os requisitos servem para orientar a construção de um programa de computador baseado nas necessidades que ele deverá suprir. Há dois tipos de requisitos: os funcionais e os não funcionais.

As próximas seções detalharão os requisitos funcionais e não funcionais que, se atendidos plenamente, possibilitarão a validação prática dos conceitos já abordados nesta monografia.

4.1.1 Requisitos Funcionais

Os Requisitos Funcionais possuem relação direta com as funcionalidades e comportamentos esperados de um sistema. Ou seja, dada uma determinada ação de um usuário ou de outro ator, os requisitos funcionais descrevem o comportamento do *software* após esses eventos (SILVA; VIDEIRA, 2001).

A Tabela 5 contém a relação dos requisitos funcionais a serem desenvolvidos neste trabalho proposto:

Tabela 5 – Requisitos Funcionais

Requisitos Funcionais	
Código	**Descrição**
RF001	Permitir a captura automática de texto inserida pelo usuário no seu navegador de internet.
RF002	Permitir a classificação do texto informado pelo usuário de acordo com a sua classe emotiva.
RF003	Permitir o *feedback* do usuário quanto a eficácia do serviço de classificação emotiva.
RF004	Permitir a mineração sob demanda de novas informações textuais para treinamento dos algoritmos.
RF005	Registrar em banco de dados os dados minerados e as classificações efetuadas.
RF006	Permitir a avaliação do grau de certeza quanto às classificações efetuadas.
RF007	Permitir que o software mantenha constante aprendizado com base nos *feedbacks* dos usuários e textos das redes sociais.

Fonte: Do Autor.

4.1.2 Requisitos Não Funcionais

Os Requisitos Não Funcionais, permitem especificar necessidades não atreladas a funcionalidades ou comportamentos, mas sim, com aspectos gerais como: desempenho, tecnologia, segurança, etc. (SILVA; VIDEIRA, 2001).

A tabela 6 contém a relação dos requisitos não funcionais a serem desenvolvidos:

Tabela 6 – Requisitos Não Funcionais

Requisitos Não Funcionais	
Código	**Descrição**
RNF001	Permitir a instalação do software cliente por meio de uma extensão para o navegador Google Chrome.
RNF002	Permitir que vários usuários utilizem o serviço simultaneamente.
RNF003	Efetuar o desenvolvimento do software em arquitetura de serviços.

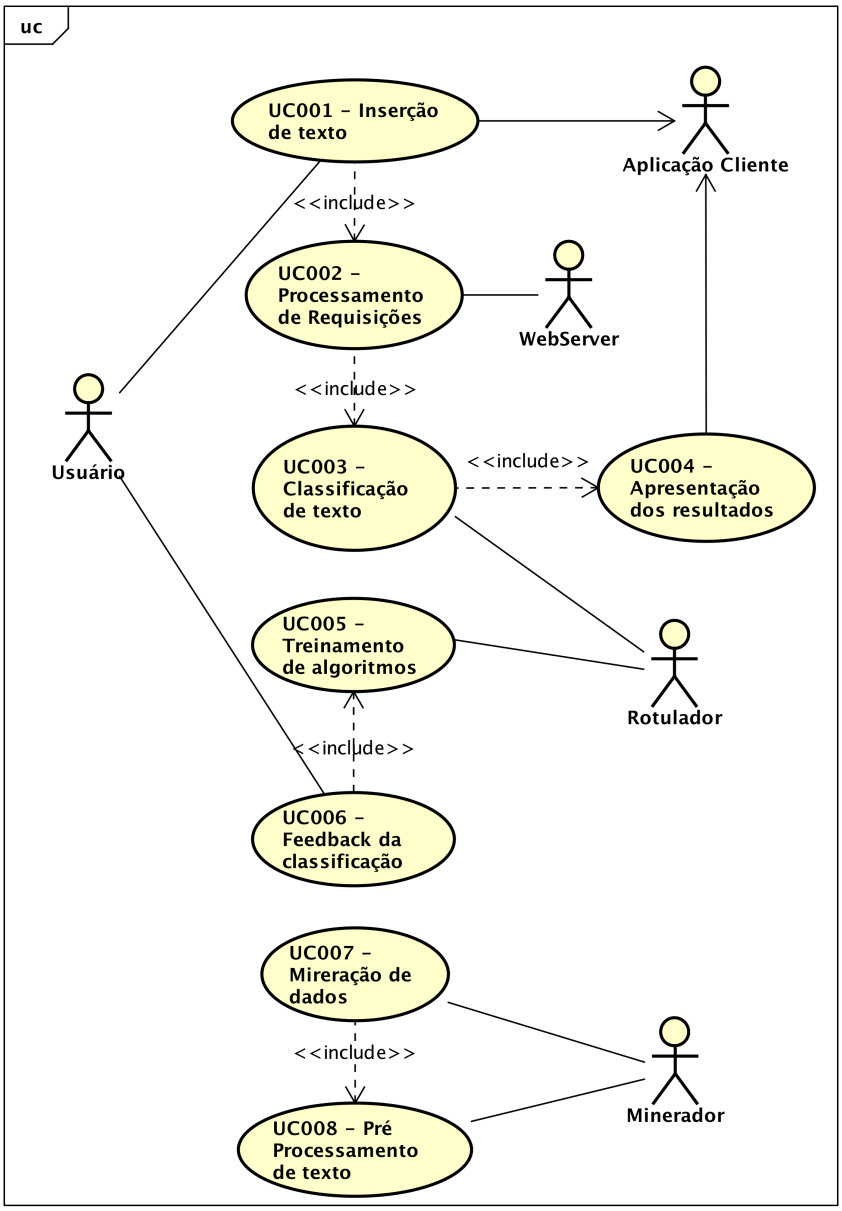
Fonte: Do Autor.

4.2 Casos de Uso

Os casos de uso especificam um conjunto de funções ou ações a serem efetuadas no sistema e a relação com os atores que as executam (OMG, 2017). Ou seja, dados os requisitos a serem desenvolvidos, é necessário descrever na forma de funções e atividades, como que cada ator do software atua para que tais requisitos sejam alcançados.

Para facilitar a visualização e a compreensão dos casos de uso, é possível demonstrar graficamente pelo diagrama de casos de uso A Figura 6 contém o Diagrama de Casos de uso que demonstra os casos de uso relacionados aos respectivos atores do software.

Figura 6 – Diagrama de Casos de Uso



Fonte: Do Autor.

Podemos reparar que há uma interdependência entre os diferentes casos de uso vinculados ao software. Cada micro-serviço pode ser considerado como um ator dentro do software, pois realizam funções desacopladas umas das outras. A tabela 7 descreve cada um dos Casos de Uso presentes no diagrama.

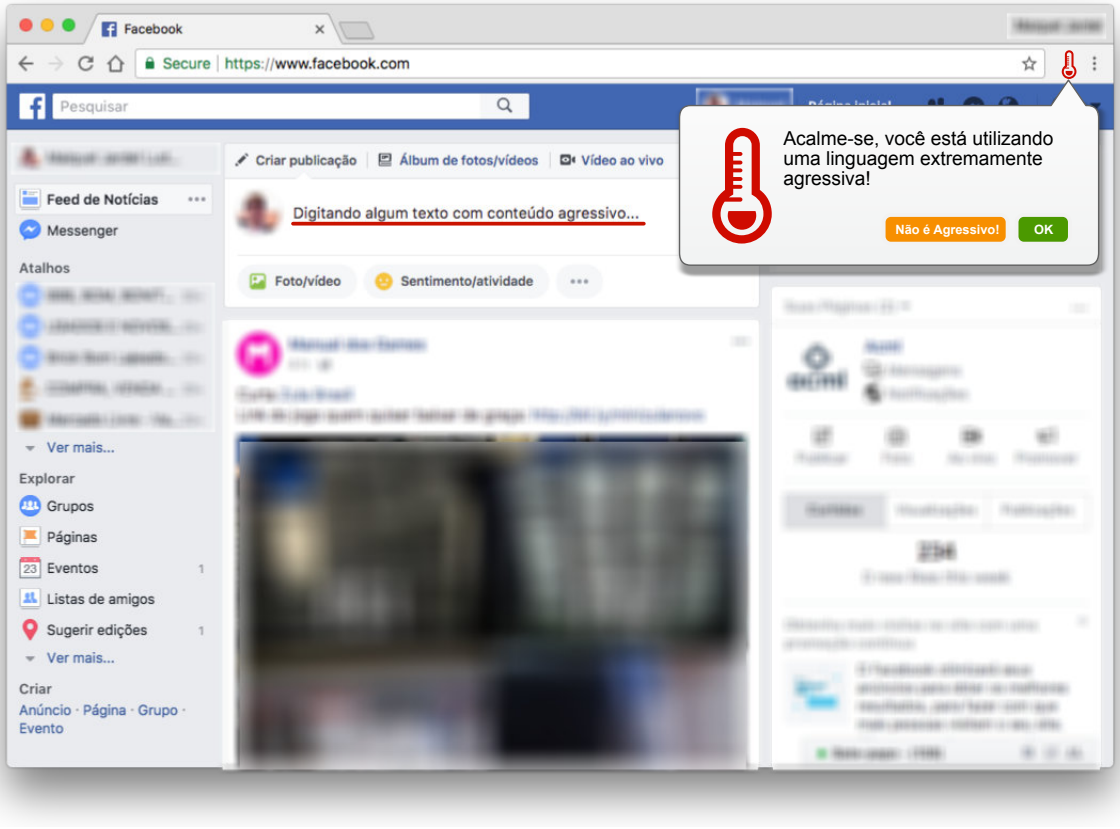
Tabela 7 – Descrição dos Casos de Uso

Casos de Uso	
Código	**Descrição**
UC001	O Usuário será capaz de inserir textos em seu navegador e a Aplicação Cliente deverá coletar esses textos e solicitar o seu processamento e apresentar os resultados processados.
UC002	O Webserver deverá processar as requisições provenientes da Aplicação Cliente e encaminhar o seu processamento ao Rotulador.
UC003	O Rotulador deverá ser capaz de classificar os textos de acordo com a classe emotiva a qual pertencem.
UC004	O texto já classificado deverá ser apresentado ao usuário pela Aplicação Cliente.
UC005	O Rotulador deverá ser capaz de adquirir aprendizado de acordo com o conjunto de treinamento fornecido pelo Minerador e com os *feedbacks* provenientes do Usuário.
UC006	O Minerador será capaz de coletar textos a partir das redes sociais.
UC007	O Minerador deverá efetuar o pré-processamento dos textos minerados e garantir o seu armazenamento em banco de dados.

Fonte: Do Autor.

A Interface de Usuário tem seu escopo restrito aos casos de uso de códigos UC001, UC004 e UC006. A Aplicação Cliente é responsável por coletar os textos digitados pelo usuário, por apresentar visualmente o índice de agressividade por meio da extensão para o navegador e por coletar o *feedback* do usuário quanto a efetividade. A expectativa visual para a aplicação destes casos de uso pode ser observada na Figura 7.

Figura 7 – Protótipo de Interface de Usuário do Software



Fonte: Do Autor.

4.3 Persistência de dados

O escopo desta monografia e os processos descritos neste trabalho proposto, são voltados fortemente ao processamento da linguagem e no aprendizado de máquina, portanto, o armazenamento de dados será bastante enxuto. Por questões práticas e de desempenho, optou-se por utilizar um banco de nados não relacional, que armazena estrutura de dados na forma de documentos e agrupa-os em Coleções.

As coleções de dados serão compartilhadas entre os serviços Minerador, Rotulador e Webserver, de forma que ambos poderão consultar, inserir ou atualizar informações. A Tabela 8 contém o detalhamento de cada uma das coleções envolvidas.

Tabela 8 – Descrição das Coleções do banco de dados

Detalhamento das Coleções	
Tabela	**Descrição**
raw_data	Coleção que armazena os dados minerados pelo serviço Minerador. Os dados são armazenados sem quaisquer tipo de processamento e são classificados de acordo com o tipo de busca efetuada na rede social.
pos_data	Coleção que armazena os dados pré-processados e prontos para serem utilizados como grupo de treinamento pelo serviço Rotulador.
log_activity	Coleção que armazena o registro de todas as requisições provenientes da Aplicação Cliente que necessitam processamento ou informam o *feedback* do usuário.

Fonte: Do Autor.

4.4 Prova de Conceito

A fim de viabilizar a implementação proposta nesta monografia, foi desenvolvido um protótipo inicial simplificado utilizando a biblioteca Javascript NaturalNode (NATURALNODE, 2017). O protótipo é capaz de simular em pequena escala, as etapas de pré-processamento, treinamento e classificação de textos. Como base de treinamento supervisionado, foram selecionados vinte textos simples, classificados como “Agressivos” ou “Não Agressivos”. O conjunto teste utilizado pode ser visualizado na Figura 8.

Figura 8 – Base de treinamento supervisionado

```
['eu sou admirada por muitos', 'Não Agressivo'],
['me sinto completamente sozinho nesta casa', 'Não Agressivo'],
['mas você é burro mesmo', 'Agressivo'],
['estou me sentindo um lixo ultimamente', 'Não Agressivo'],
['pare de ser tão idiota seu lixo', 'Agressivo'],
['vai tomar no seu cu seu desgraçado', 'Agressivo'],
['o dia está muito bonito', 'Não Agressivo'],
['estou de saco cheio com o comportamento desse menino', 'Não Agressivo'],
['pare de encher o saco seu idiota', 'Agressivo'],
['essa porra ai não vai dar em nada', 'Agressivo'],
['merda de empresa que só fode com os consumidores', 'Agressivo'],
['vai se foder seu filho da puta', 'Agressivo'],
['não sou nenhum idiota, você deveria ter vergonha na cara', 'Não Agressivo'],
['este lugar e apavorante', 'Não Agressivo'],
['se perdermos outro jogo seremos eliminados', 'Não Agressivo'],
['seu desgraçado de merda', 'Agressivo'],
['se eles descobrirem estamos encrencados', 'Não Agressivo'],
['nem fodendo que volto para aquela casa', 'Agressivo'],
['é foda quando isso acontece com um de nós', 'Agressivo'],
['o filho da puta vem todo dia encher o saco', 'Agressivo']];
```

Fonte: Do Autor.

Um conjunto de treinamento reduzido pode apresentar limitações de classificação quando houverem palavras desconhecidas para o classificador. Mesmo assim, foi possível obter bons resultados de classificação utilizando o algoritmo Naive Bayes. A figura 9 demonstra a saída visual do protótipo, quando o texto “nem vou escutar aquele idiota desgraçado” foi classificado como “Agressivo” com probabilidade de 88%.

Figura 9 – Saída do software de prova de conceito

```
Classificando o texto: "nem vou escutar aquele idiota desgraçado"

Resultado = "Agressivo"
Agressivo :  88.24 %
Não Agressivo :  11.76 %
```

Fonte: Do Autor.

Com estes resultados preliminares foi possível concluir que há viabilidade técnica e conceitual para o desenvolvimento do software em pequena escada, já que em sua implementação definitiva, foi necessário considerar um volume de dados muito maior e proporcionar uma boa experiência aos usuários finais. Todo esse volume de dados, aliado a expectativa de um resultado efetivo útil, introduz uma grande quantidade de desafios interessantes a serem superados.

5 DESENVOLVIMENTO

O desenvolvimento de um software capaz de colocar em prática as metodologias propostas e aplicar os conceitos bibliográficos referenciados requer uma série de escolhas: a plataforma destino do software, a linguagem de programação, as ferramentas, a arquitetura, etc. Neste contexto, a escolha mais simples está no codinome do projeto, denominado de KeepCalm, fazendo alusão a necessidade de manter-se dentro dos padrões sociais aceitáveis enquanto se está utilizando a internet.

Este capítulo servirá de referência técnica ao leitor e descreverá os principais pontos e escolhas no desenvolvimento do KeepCalm.

5.1 Tecnologias

Conforme proposto no capítulo 4, o KeepCalm é composto por 4 quatro serviços principais com responsabilidades desacopladas: Minerador, Rotulador, Webserver e Aplicação Cliente. Ou seja, não é necessário que todos sejam executados em um mesmo ambiente e nem que sejam desenvolvidos dentro dos mesmos padrões de linguagem e arquitetura. É preciso apenas garantir que consigam trocar informações entre si de forma eficiente.

Mesmo assim, um padrão de projeto foi seguido, pelo menos no que se referente à linguagem de programação. Todos os serviços foram desenvolvidos utilizando a linguagem de programação Javascript (ECMAScript 6) na camada *back-end*, enquanto que o *front-end* (camada visual) foi desenvolvido com a linguagem de marcação HTML5 em sua forma pura. Optou-se pela utilização do conjunto Javascript/HTML pelo fato do KeepCalm ser uma aplicação *web* que apresenta suas informações em um navegador de internet e tem seus serviços interconectados através de rotas. Por isso, justifica-se a utilização de tecnologias projetadas para o ambiente de internet.

Os serviços Minerador, Rotulador e Webserver podem ser executados em ambiente *desktop* sem a necessidade de um navegador de internet. Optou-se portanto, pela utilização do ecossistema Node.js, que permite a utilização do motor Google V8 para a interpretação de Javascript diretamente em ambiente Desktop (NODE.JS, 2018).

Além disso, o Javascript é uma linguagem extremamente popular, como mostra o relatório Octoverse, disponibilizado anualmente pelo Github (2018). Segundo o relatório, o Javascript foi a linguagem com o maior número de *pull requests* em 2017 dentre as mais de 300 linguagens da plataforma. Toda essa popularidade faz com que haja uma grande quantidade de bibliotecas e ferramentas de apoio desenvolvidas pela comunidade. Uma dessas bibliotecas é a NaturalNode, que fornece ferramentas Javascript para a manipulação e classificação de textos para o processamento de linguagem natural (NATURALNODE, 2017).

Para o desenvolvimento do KeepCalm, optou-se por um paradigma não relacional de banco de dados, conhecido como NoSQL, orientado a documentos.

> A definição geral apresentada é que os Bancos de Dados orientados a Documentos utilizam o conceito de dados e documentos autocontidos e auto descritivos, e isso implica que o documento em si já define como ele deve ser apresentado e qual é o significado dos dados armazenados na sua estrutura (DEVMEDIA, 2018).

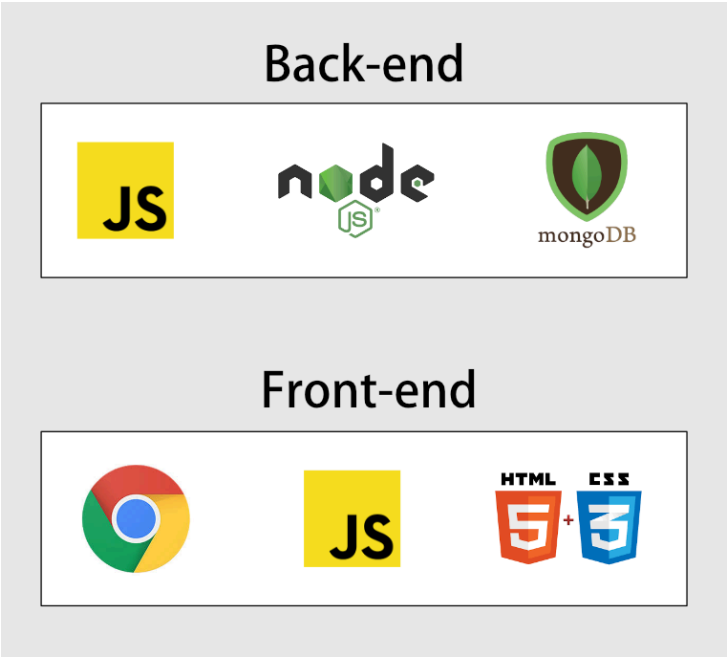
Optou-se pela utilização do banco de dados não relacional MongoDB por uma série de razões. O MongoDB permite a instalação e configuração do servidor de forma muito simplificada. Há bibliotecas para Javascript como o Mongoose (MONGOOSE,

2018) que permitem um fácil gerenciamento das conexões entre o aplicativo e o banco de dados. Ele é muito eficiente no armazenamento e na consulta de um grande volume de dados, já que não necessita realizar o produto entre diversas tabelas relacionadas para compor suas entidades.

Por fim, a aplicação cliente, que é responsável por captar os dados de uso do usuário e apresentar os resultados, foi construída dentro de uma extensão para o navegador Google Chrome. Segundo a documentação de desenvolvedor do Google, “Extensões são pequenos programas que customizam a experiência de navegação” (CHROME, 2018). Optou-se por esse mecanismo, pois é preciso que o KeepCalm atue de forma transparente em todas as páginas de internet que o usuário utiliza, captando os dados inseridos pelos usuários e monitorando a agressividade dos textos.

Mais detalhes referentes a arquitetura de desenvolvimento e escolhas técnicas serão abordados na próxima seção. A figura 10 resume a utilização das principais tecnologias na camada de negócio e persistência (*back-end*) e na camada de apresentação (*front-end*).

Figura 10 – Resumo das tecnologias utilizadas no KeepCalm



Fonte: Do Autor.

5.2 Arquitetura

O modelo de arquitetura *Model View Controller* (MVC), é um padrão de arquitetura amplamente popular que separa a lógica das aplicações desenvolvidas nesse modelo, de modo a segmentar suas responsabilidades, permitir a modularidade, desenvolvimento colaborativo e principalmente, o reuso de lógicas desenvolvidas (MOZILLA, 2018). Como o KeepCalm tem seu desenvolvimento segmentado em serviços de responsabilidade bem específica, acabou-se por utilizar um modelo híbrido do MVC.

Os serviços Minerador e Classificador possuem aspectos de *Controller* e *Model*, uma vez que lidam com o processamento dos dados e sua representação como entidades. Os modelos são estruturas de dados que podem ser armazenados e manipulados computacionalmente e, no KeepCalm, optou-se pela utilização do Mongoose (2018) que permite uma maior abstração do código e agilidade de desenvolvimento.

Já o serviço Webserver, abrange o conceito de *Controller* somente, pois atende as requisições provenientes da camada *View* e direciona os pedidos de processamento aos demais serviços.

Por fim, a camada *View* é representada pelo que o usuário enxerga e interage com, ou seja, o plugin do KeepCalm. Todas as ações que o usuário realiza na camada de apresentação, resulta em um processamento de dados (*Controller*), assim como a sua consulta e armazenamento em banco de dados (*Model*).

As próximas seções descreverão mais detalhadamente cada um dos serviços que compõem o KeepCalm.

5.2.1 Minerador

O serviço Minerador é responsável por duas tarefas principais que, uma vez realizadas, permitem que o serviço possa ser temporariamente desativado ou com funcionamento passivo.

A primeira responsabilidade, é realizar a coleta de textos agressivos e não agressivos na internet. Como fonte de dados, optou-se pela utilização da rede social Twitter que, por ser amplamente utilizada, contém informações em grande volume e de fácil acesso. Além disso, a plataforma possui limites no tamanho dos textos, obrigando seus usuários a serem mais concisos e diretos em suas declarações. O Twitter é amplamente utilizado para a disseminação de conteúdos agressivos e constantemente abordado como forma oficial para xingamentos a personalidades públicas. A famosa frase “vou xingar muito no Twitter” resume bem esse conceito.

Outro fator relevante para a escolha da fonte de dados, foi a disponibilidade de recursos para o desenvolvedor, como APIs de consulta à base de dados e filtros de conteúdos. O Twitter disponibiliza a biblioteca *Search Tweets* (TWITTER, 2017B), que permite a localização de *tweets* públicos com base em uma série de filtros. Utilizando esta API, foi possível obter vinte mil *tweets* públicos com base nas palavras-chave: “fdp”, “caralho”, “merda”, “bosta”, “porra”, “puta” e “foder”. Outros vinte mil *tweets* também foram coletados de forma aleatória (sem o uso das palavras-chave), para contrabalancear a base de treinamento com conteúdo não agressivo e gerar diversidade de dados.

Após a coleta, os textos são encaminhados ao pré-processador, onde passam por uma limpeza, removendo símbolos e palavras inúteis ao processamento. Esta etapa é importante para que haja um enxugamento das informações, facilitando a posterior classificação e economizando recursos de processamento.

Em seguida, os *tweets* são submetidos a análises sintáticas e morfológicas (ver seções 2.2.3 e 2.2.4). A análise sintática auxilia na descoberta da classe da palavra

segundo o padrão PoS, permitindo a eliminação de palavras pouco relevantes emocionalmente como artigos, preposições e conjunções. Para isto, optou-se pela utilização de um conjunto de regras léxicas, desenvolvido pelo usuário diegodorgam em sua página no Github (2018B), que permite a análise sintática com base na disposição dos *tokens* dentro do texto.

A última etapa do pré-processador, realiza uma análise morfológica das palavras, extraindo o seu radical (*steeming*) e reduzindo ainda mais o tamanho da base de dados. O mecanismo de *steeming* foi desenvolvido utilizando uma das funcionalidades da biblioteca NaturalNode e implementando o algoritmo de Porter, que analisa o contexto morfológico, detecta os morfemas das palavras e elimina seus sufixos flexionais (TARTARUS, 2018). As palavras “foder” e “fodido”, por exemplo, são consideradas como sendo o mesmo *token* para o KeepCalm.

A Figura 11 exemplifica as chamadas para os métodos de pré-processamento dos textos minerados, incluindo a eliminação de conteúdos irrelevantes e obtenção dos radicais.

Figura 11 – Funções do pré-processador

```
let preProcessText = (text) => {

    //Remove resíduos textuais como "ahuahua", "kkkk", "http", etc
    processedText = removeTrashWords(text);

    //Remove simbolos e números
    processedText = removeSymbols(processedText);

    // Efetua Análise Sintática e remove palavras sintaticamente
    // irrelevantes para a detecção de emoções
    processedText = removeIrrelevantWorlds(processedText);

    //Análise Léxica, extração dos radicais
    processedText = steemer(processedText);

    return processedText;
};
```

Fonte: Do autor.

Após o pré-processamento, tem-se como resultado um conjunto de treino que será utilizado pelo Rotulador para a classificação de textos. Esse conjunto é armazenado na base de dados para que possa ser consultado posteriormente pelos demais serviços do KeepCalm.

5.2.2 Rotulador

O Rotulador é o serviço responsável por treinar os algoritmos de classificação de textos, utilizando como base de treinamento, dois grupos de informações: o primeiro grupo é composto pelos *tweets* pré-processados e armazenados pelo serviço minerador, o segundo grupo é proveniente dos históricos de classificações já efetuadas e já avaliadas pelo usuário do KeepCalm.

O treinamento efetuado com base nos milhares de *tweets* minerados, é considerada semissupervisionada, pois mesmo que mineração tenha ocorrida com base em palavras-chave claramente agressivas, não há como ter certeza se um determinado *tweet* é realmente agressivo ou não. Como já detalhado na seção 2.2.1, o aprendizado semissupervisionado deve ser utilizado quando não se há certeza das saídas para cada grupo de dados utilizados como entrada (RUSSELL; NORVIG, 2013).

Já o segundo grupo de treinamento formado pelo registro de classificações efetuadas pelo KeepCalm, permite que haja uma aprendizagem supervisionada, já que as classificações que obtiveram o *feedback* do usuário, possuem as entradas e saídas individualmente supervisionadas. É esperado que este segundo grupo de treino tenha a capacidade de refinar as capacidades de classificação do algoritmo do KeepCalm de acordo com os perfis dos usuários, fazendo com que a ferramenta se atualize constantemente e aumente sua eficiência com o passar do tempo.

A biblioteca NaturalNode, disponibiliza uma interface simplificada para a seleção do algoritmo desejado, criação do classificador, adição de elementos ao grupo de treino

e posterior classificação dos elementos. Os algoritmos desenvolvidos e já presentes dentro da API do NaturalNode incluem os dois classificadores escolhidos para a comparação desta monografia: o algoritmo Naive Bayes e o Regressão Logística.

A utilização é bastante enxuta: basta instanciar o classificador desejado, adicionar documentos de treino com entradas e saídas supervisionadas ou semissupervisionadas e solicitar a classificação. É importante neste momento, que seja informado ao classificador, o método de *steeming* escolhido e também o idioma, para que o classificador consiga interpretar as palavras de acordo com as regras sintáticas corretas.

Figura 12 – Lógica básica do Rotulador usando o método Naive Bayes

```
// Inicializa o classificador com o algoritmo desejado
let bayesClassifier = new natural.BayesClassifier(natural.PorterStemmerPt);

// Adiciona elementos na base de treino
bayesClassifier.addDocument("Deixe de ser babaca!", "Agressivo");
bayesClassifier.addDocument("Uma vez eu fui na casa dele.", "Não Agressivo");
bayesClassifier.addDocument("Sempre torci por ele, mas ele não.", "Não Agressivo");

// Obtêm a classificação do texto informado
let result = classifier.classify("Será que este texto é agressivo?");
```

Fonte: Do Autor.

O acesso ao Rotulador do KeepCalm foi encapsulado através de uma API de rotas, desenvolvida com o apoio do ExpressJS (2018). Com isso, é possível que qualquer serviço autorizado, possa solicitar ao Rotulador para que classifique um determinado texto enviado como parâmetro. A API então retorna os resultados através de um arquivo JSON, que é uma notação de objeto amplamente utilizada para a comunicação entre serviços (W3SCHOOLS, 2018). O JSON traz todos os atributos obtidos pelo classificador, incluindo os percentuais de probabilidade para cada resultado, a classificação PoS Tag para cada *token* e o resultado efetivo do Rotulador.

Figura 13 – JSON retornado pelo Rotulador


```
{
    "text": "Vá embora seu fedido!",
    "result": "Agressivo",
    "details": [
        {
            "label": "Agressivo",
            "value": 3.327634226682436e-8
        },
        {
            "label": "Não Agressivo",
            "value": 1.2466570807269804e-9
        }
    ],
    "percentuals": {
        "Agressivo": "96.39",
        "Não Agressivo": "3.61"
    },
    "effectiveResult": "Agressivo",
    "pos_tag": [["Vá","LDEN"],["embora","LDEN"],["seu","PPS"],["fedido!","NN"]
    ],
    "activityID": "5b072cdc1822960b0d77835d"
}
```


Fonte: Do autor.

Cada classificação efetuada pelo Rotulador, gera um registro de atividade que é armazenado na base de dados. Um código ID é atribuído a cada atividade e enviado juntamente com o JSON para que, no momento do *feedback* do usuário, seja possível retornar ao Rotulador a assertividade da classificação, gerando um registro de atividade passível de uso no grupo de treinamento supervisionado. Esse *feedback* deve ser encaminhado através de outra rota específica para o tratamento de retorno e registro da opinião dos usuários na respectiva atividade previamente salva na base de dados.

5.2.3 WebServer

O Rotulador é um serviço que demanda uma grande quantidade de recursos de memória RAM e de armazenamento em disco, por lidar com um grande volume de dados em seu campo amostral. Com isso, optou-se por elaborar um serviço dedicado somente ao tratamento de requisições provenientes dos usuários por meio do *plugin*

instalado em seus navegadores. Denominado de WebServer, esse serviço conta com recursos de hardware muito mais limitados, suficientes para lidar tranquilamente com centenas de requisições por segundo.

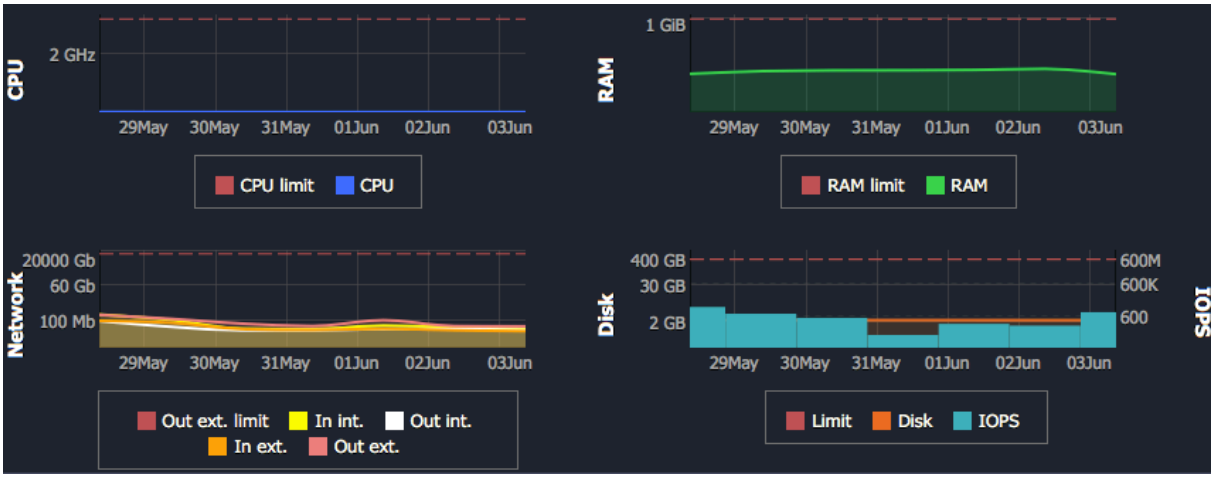
Além da economia de recursos, essa divisão permitiu atribuir um endereço IP fixo e público ao WebServer, sendo acessível a partir da internet em todo o mundo. O WebServer é totalmente independente e isolado do Rotulador, criando uma camada a mais de proteção contra possíveis ataques cibernéticos e encapsulando os dados do KeepCalm através da API padronizada já descrita na seção 5.2.2. A cada requisição proveniente da aplicação cliente, o WebServer valida os dados enviados pelo *plugin* instalado no computador do usuário e encaminha a solicitação de acordo com o seu tipo, podendo ser uma requisição para classificação de texto ou ainda uma solicitação para salvamento do *feedback* do usuário em uma classificação previamente efetuada.

Todos os dados trafegados entre o WebServer e o Rotulador são encaminhados obrigatoriamente através da API. Apesar de não ser o foco principal desta monografia, este modelo permite a elaboração de várias camadas de autenticação e de segurança para evitar ataques que visam a incapacitação do sistema por sobrecarga ou ainda por acesso não autorizado aos dados de outros usuários.

Seguindo o padrão já utilizado nos demais serviços do KeepCalm, o WebServer foi desenvolvido utilizando uma combinação bastante prática envolvendo o NodeJS e a biblioteca ExpressJS. Com um escopo bem definido e ferramentas bastante enxutas, o WebServer é todo contemplado em um único arquivo contendo 118 linhas de código. Dificilmente seria possível efetuar o mesmo desenvolvimento de forma tão resumida em outra linguagem de programação que não o Javascript.

Tanto o Rotulador quanto o WebServer foram hospedados no serviço Jelastic, que oferece um serviço PaaS (Plataforma como um serviço) de forma elástica, onde os recursos de hardware são disponibilizados conforme a demanda (JELASTIC, 2018). Além de oferecer os contêineres para a execução de cada um dos serviços, o Jelastic permite o monitoramento das atividades na forma de gráficos, conforme a Figura 14 a seguir.

Figura 14 – Monitoramento de recursos do Jelastic



Fonte: Do Autor com base em Jelastic (2018)

5.2.3 Aplicação Cliente

Todos os serviços previamente descritos neste capítulo, todos os algoritmos, linhas de código, base de dados, metodologias e tecnologias, são meios necessários para exibir algumas poucas informações ao usuário. A camada de apresentação do KeepCalm é o ponto do software onde todos os conceitos abordados são transformados em informações que visam o apoio ao usuário na detecção de possíveis comportamentos agressivos.

Os navegadores de internet são os softwares mais utilizados. 79% dos computadores pessoais possuem o navegador Google Chrome instalados e 44% deles possuem o Mozilla Firefox (AVAST, 2017). Unindo os dados desta pesquisa aos objetivos deste trabalho, optou-se pelo desenvolvimento de um software acoplável a um navegador de internet para que possa ser inserido em ambientes onde os usuários estão mais suscetíveis a visualizar e propagar conteúdos violentos. Esse tipo de módulo acoplável é chamado de *plugin* ou extensão.

O Google disponibiliza em seu site voltado a desenvolvedores, modelos e boas práticas para o correto desenvolvimento de Extensões para o seu navegador

(CHROME, 2018). Seguindo o estilo visual elaborado como caso de uso na seção 4.2, chegou-se a um resultado semelhante, desenvolvido com uma combinação de HTML, CSS e Javascript.

Em resumo, após instalada a extensão, o Google Chrome passa a monitorar os elementos da página que possuem propriedades para imputação de dados (desde que não sejam campos de dados sensíveis como senhas, e-mail, datas e endereços). Sempre que o usuário digitar alguma informação em algum formulário, e-mail ou rede social, o KeepCalm envia o conteúdo digitado na forma de requisição ao WebServer, que retorna os dados classificados pelo Rotulador. Em seguida a Extensão colore o seu ícone de acordo com o resultado obtido, sendo vermelho para textos considerados agressivos e verde para textos considerados não agressivos.

O usuário pode então clicar sobre o ícone colorido do *plugin* e visualizar seus detalhes. É neste momento que o KeepCalm pode coletar a opinião do usuário a respeito da classificação recém efetuada, que é então encaminhada ao WebServer para que seja incluída pelo Rotulador como fonte de treinamento supervisionado em classificações futuras.

Figura 15 – Tela de detalhamento e *feedback* do KeepCalm.

Fonte: Do Autor.

Para facilitar a sua distribuição e posterior atualização em todos os clientes, a Extensão foi publicada na loja oficial do Google Chrome de modo que possa ser

localizada. Como trata-se de uma versão Beta, ainda não é possível localizá-la pela busca por aplicativos, mas somente com o link correto. O Anexo A contém todas as instruções para a instalação do KeepCalm no navegador Google Chrome.

É importante destacar que a instalação da Extensão requer que o usuário dê as permissões solicitadas, que envolvem o acesso completo às páginas que o usuário acessa e de todos os dados textuais inseridos em campos do tipo “texto”. Mesmo efetuando controles para que o KeepCalm não colete informações pessoais, há um certo receio de que o aplicativo possa ser bloqueado pelo Google e banido da loja oficial. Por hora, o objetivo é apenas acadêmico, mas esse problema pode impedir futuras ampliações comerciais do KeepCalm.

6 TESTES E RESULTADOS

Após o desenvolvimento do software de codinome KeepCalm conforme descrito no capítulo anterior, é preciso avaliar a sua eficácia e aplicabilidade em cenário real, comparando os resultados com as informações pesquisadas e descritas no referencial teórico do presente trabalho. Os experimentos práticos são essenciais para a precisão da avaliação de sua eficácia e fazem parte do conceito de método científico.

> Quanto mais precisas as predições experimentais da teoria, maior a sua falseabilidade. Teorias vagas e imprecisas são imunes ao eventual veredito negativo dos testes a que seja submetida, e isso é séria desvantagem, pois desestimula a busca de teorias melhores (CHIBENI, 2006, p. 3).

As seções deste capítulo descrevem o método de testes aplicado a um grupo de usuários reais e os resultados obtidos após cada rodada.

6.1 Metodologia de Testes e Calibragem

Conforme já visto no referencial teórico, o Aprendizado de Máquina é um processo de educação computacional semelhante ao aprendizado humano, onde é preciso informar exemplos e regras consideradas verdadeiras, para que o computador

“aprenda” com elas e seja capaz de deduzir informações que sejam similares ou se encaixem nas regras aprendidas.

Trazendo para o cenário deste trabalho, onde se pretende descobrir se determinado texto é considerado “agressivo” ou “não agressivo”, é necessário que o serviço rotulador seja alimentado com o maior número possível de textos de ambas as categorias. Como visto no capítulo 5, a base de dados de aprendizado inicial foi obtida a partir da mineração de *tweets* cuja classificação de agressividade foi deduzida com base nas palavras-chave “filho da puta”, “caralho”, “merda” “bosta” e “porra”, gerando um aprendizado semissupervisionado. Desta forma, não é possível ter certeza se determinado *tweet* é realmente agressivo ou se ele apenas contém alguma palavra geralmente classificada como agressiva. Podemos considerar então que há uma alta probabilidade de que hajam *tweets* pré-classificados de forma errada.

Por exemplo, a frase “ela é uma criança de crescimento retardado” pode ser considerada como agressiva devido ao fato de que a palavra “retardado” é frequentemente utilizada para fins agressivos. A probabilidade de que determinada palavra represente ou não um contexto agressivo no texto, é o ponto chave para a precisão nos resultados.

Uma forma de resolver esse problema é criando ciclos de calibragem com treinamento mais próximo do supervisionado. Um exemplo comercial é o ReCaptcha (GOOGLE, 2018C), serviço online que evita abuso de uso e *spam* e, além disso, utiliza do conhecimento humano em reconhecer letras distorcidas para treinar os algoritmos para digitalização de textos. Todas as vezes que um humano resolve um *captcha*, é gerada uma nova regra de treinamento supervisionado para a calibragem do algoritmo. No futuro, é possível que tal algoritmo seja capaz de digitalizar textos de maneira muito eficiente.

Esse processo de calibragem é comum nos serviços que utilizam o aprendizado de máquina e para o KeepCalm não será diferente. As seções a seguir descrevem o processo iterativo de calibragem e validação dos resultados em cada rodada com os usuários.

6.2 Aplicação de testes iterativos

Os resultados obtidos são fontes recursivas de aprendizado, gerando novas rodadas de testes enquanto os algoritmos são calibrados para obter a melhor relação de acertos. Foram realizadas três rodadas de teste, intercalando processos de análise e otimização. Para a primeira rodada, a fim de facilitar a instalação e configuração do KeepCalm pelos usuários de teste, foi elaborado um documento contendo um formulário de apoio e um passo-a-passo para a realização dos testes (ANEXO A).

O grupo de teste, foi constituído de 8 voluntários adeptos a novas experiências. Atualmente são chamados de *early adopters*, os usuários com maior disposição para experimentar novas soluções (ROMANKAYO, 2018). Esses usuários, orientados pelo passo-a-passo fornecido conforme o Anexo A, elaboraram aproximadamente trinta frases. Um terço delas sendo claramente agressivas, podendo ou não conter palavrões e palavras obscenas. O segundo terço com frases claramente não agressivas, podendo inclusive conter palavrões, mas de forma contextualizada e, ainda, claramente sem agressões verbais. A porção final foi destinada a elaboração de frases agressivas em seu contexto, porém sem a utilização explícita de termos agressivos, podendo conter elementos de racismo, preconceito e sarcasmo.

Os testes não controlaram a elaboração obrigatória das frases e nem a classificação em categorias. Esse formato foi adotado apenas para gerar diversidade na base de dados de treinamento. É importante considerar também que a categorização das frases é um processo individual e pode divergir opiniões, já que, o que é agressivo para uma pessoa, pode não ser para outra. Portanto o processo de análise desconsidera a categorização das frases, uma vez que podem ser classificadas de forma diferente de acordo com cada usuário. Essa diversidade de opiniões, no grupo de teste, reflete a diversidade de opiniões e preceitos morais dos diferentes utilizadores de internet pelo mundo.

As rodadas 1 e 2 são destinadas basicamente para a calibragem da base de treinamento e não dos algoritmos. Uma vez que a efetividade depende diretamente da qualidade da base de treinamento, a comparação dos algoritmos foi deixada para a terceira rodada, onde as diferenças entre os algoritmos podem ser avaliadas com uma maior taxa de confiabilidade.

É importante considerar também, que os valores de efetividade do KeepCalm das rodadas 1, 2 e 3, são relativos ao *feedback* do usuário e considerando a sua opinião particular, que pode ser diferente de acordo com uma série de fatores culturais, políticos e educacionais. Como Goldberg (2004, p.38) descreve, a agressividade pode ser transmitida e absorvida de diversas formas e as pessoas são expostas diariamente a essas agressões, muitas vezes sem perceber:

> "Em certas ocasiões, a simples emissão de informações ameaçadoras é suficiente para submeter pessoas ou grupos ao domínio e ao pânico. (...) Palavras, gestos, programas e exibições agressivas substituem, às vezes com economia de esforços, o ataque propriamente dito.

Um dos aspectos que prejudicou consideravelmente a qualidade dos testes ao longo das rodadas, foi o engajamento dos usuários, que foi reduzindo ao longo do período. É perceptível a redução na quantidade e qualidade dos textos inseridos ao longo de cada rodada, discriminando a necessidade de melhoria nos aspectos lúdicos da ferramenta.

6.3 Rodada de teste 1

A primeira rodada de testes permitiu a avaliação da qualidade do grupo de treinamento semissupervisionado obtido a partir de 40 mil *tweets*, sendo aproximadamente 50% deles considerados "Agressivos" e a outra metade "Não agressivo". Como dito anteriormente, no treinamento semissupervisionado não é possível ter certeza se determinado *tweet* é agressivo ou não, mas é possível ter uma

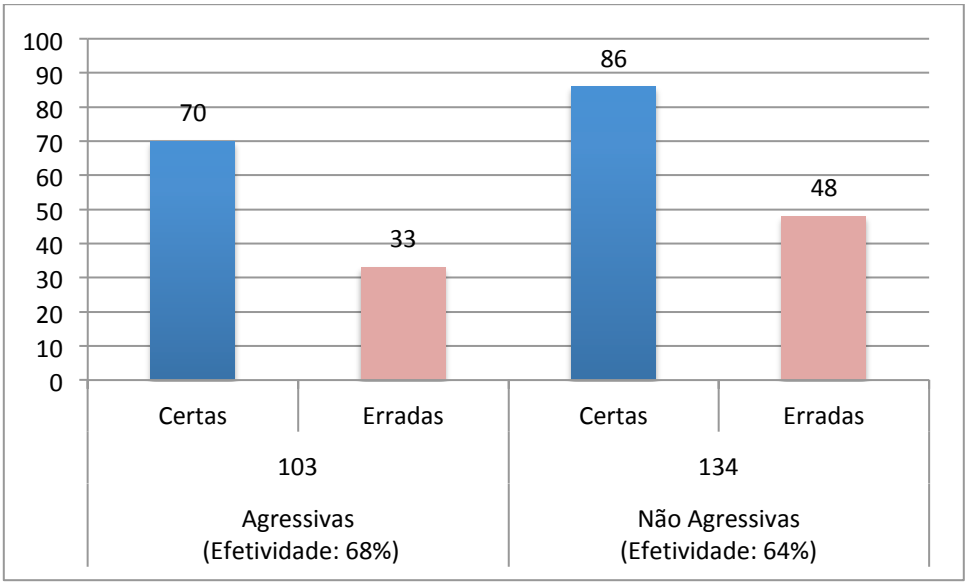
classificação estimada, se considerado o local de extração ou palavras-chave agressivas encontradas em seu escopo.

Para evitar o maior número possível de falsos positivos, estipulou-se que o KeepCalm deve ter 70% ou mais de certeza para classificar algo como agressivo. Por exemplo, se uma determinada frase for classificada como “Agressiva” com um percentual de 55%, o KeepCalm se resguardará dessa classificação e informará ao usuário que seu texto foi classificado como “Não Agressivo”, pois a taxa de certeza do algoritmo pode ser insuficiente para garantir que a classificação está correta.

Esse percentual de corte foi necessário, pois, nos testes iniciais, percebeu-se que frases não agressivas, mas que são complementares a outras frases agressivas, podem elevar o percentual de agressividade avaliado pelo KeepCalm. Um exemplo disso é o trecho “você é”, que é muitas vezes empregado para ofender alguém quando acompanhado de um palavrão, e que por isso, acaba aumentando a probabilidade de uma frase ser agressiva, mesmo não sendo. Esse ponto de corte pode ser reduzido ao longo dos testes, a medida que os algoritmos de classificação vão sendo treinados com grupos de treino mais precisos.

Durante o período de testes o KeepCalm registrou 764 classificações de textos, sendo que, 237 delas receberam *feedback* do usuário. A análise se concentra apenas sobre o histórico de avalições que possuem *feedback* do usuário, pois oferecem parâmetros para a avaliação da efetividade. O Gráfico 1 detalha quantidade de acertos para frases agressivas com efetividade de 68% e não agressivas com efetividade de 64%. Vale a pena ressaltar que essa é a efetividade considerando a opinião dos usuários de teste e não a opinião de experts no assunto. Esta monografia não questionará a habilidade dos usuários em discernir o que é agressivo ou não.

Gráfico 1 - Número de acertos, erros e efetividade por categoria de classificação obtidos com a rodada de teste 1.



Fonte: Do Autor.

Analisando algumas frases específicas, foi possível descobrir que o KeepCalm classificou corretamente como agressivas, frases com palavras diferentes das utilizadas como parâmetro de mineração de *tweets*, como “Idiota”, “bosta” e “burro”. Ele foi capaz de relacionar palavras agressivas, de forma indireta, já que, provavelmente palavras como essas foram encontradas em *tweets* considerados agressivos.

Outra constatação interessante é que a grande maioria das frases agressivas classificadas erroneamente pelo KeepCalm como “Não Agressivas”, são textos sem palavrões, mas que culturalmente podem ser interpretadas de forma agressiva e preconceituosa. Frases como “Nossa que decote tinha aquele vestido da garota”, “Não quero mais te ver na minha frente” e “O almoço estava um lixo” foram consideradas como “Não Agressivas” pelo KeepCalm, mas para os usuários testes, foram consideradas como “Agressivas”.

A efetividade geral do KeepCalm foi de 66% acertando 156 dos 237 textos processados. Muito desse resultado foi proveniente das classificações realizadas em textos claramente agressivos ou claramente não agressivos, com efetividade de

aproximadamente 90%. A maior taxa de erro se encontra nos textos em sentido dúbio ou com conteúdo racista e preconceituoso, obtendo resultados muito próximos dos padrões de aleatoriedade (próximo aos 50%). Isso ocorre principalmente pela dificuldade do entendimento do contexto das frases, que na mente dos usuários pode representar algo agressivo, mas para o KeepCalm, não há palavras relevantes o suficiente para gerar a classificação de acordo com que o usuário espera.

Algumas hipóteses foram elencadas para a busca de melhorias na eficiência para as próximas rodadas. A primeira, envolve a utilização do grupo de treino supervisionado gerado pelo *feedback* dos usuários nesta rodada, fazendo com que o classificador tenha acesso a uma maior variedade de dados. A segunda hipótese de melhoria está na orientação aos usuários de teste, para que reflitam se as frases elaboradas, são realmente agressivas e não apenas uma questão de opinião.

6.4 Rodada de teste 2

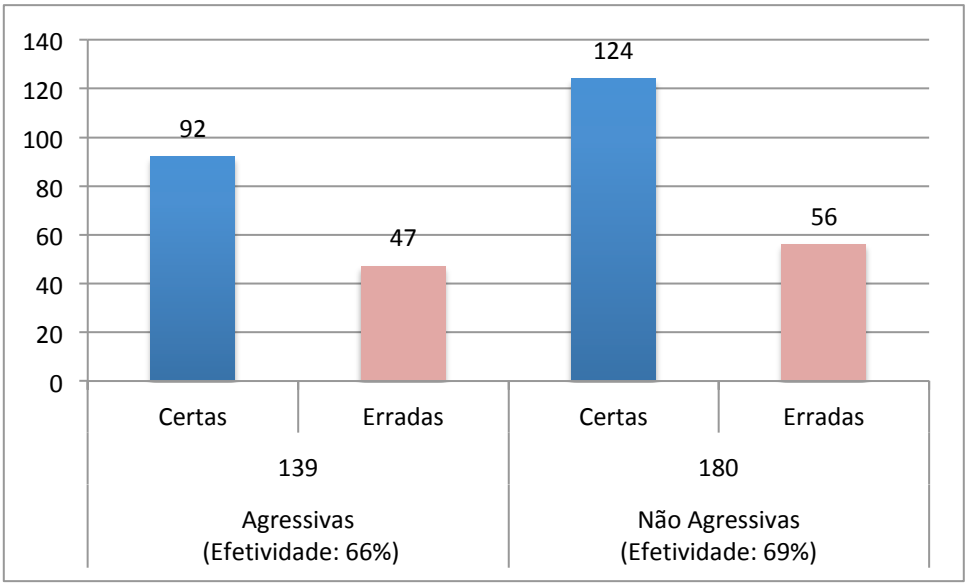
A segunda rodada de testes obteve um número de processamento de texto relativamente maior de textos, registrando um total de 1264 atividades, sendo que destas, 319 obtiveram o *feedback do usuário*. A versão disponibilizada na rodada dois trouxe algumas melhorias na extensão, passando a funcionar em praticamente qualquer site, o que aumentou as possibilidades de uso da ferramenta. O ponto de corte permaneceu em 70% para avaliar o grau de aprendizado da ferramenta, ou seja, para validar se um grupo de treino supervisionado realmente é capaz de aumentar a efetividade dos algoritmos de classificação.

Além do conjunto de treinamento original semissupervisionado (aproximadamente quarenta mil *tweets*), a segunda rodada teve cerca de 400 registros supervisionados, provenientes da rodada 1 de testes e de outras classificações realizadas durante o seu desenvolvimento. O principal objetivo da segunda rodada, foi

avaliar o grau de aprendizado da ferramenta e se com o feedback do usuário, o KeepCalm conseguiria ter maiores taxas de assertividade nas classificações em textos.

Infelizmente a efetividade dos algoritmos do KeepCalm teve um aumento de efetividade abaixo do esperado. Conforme é possível visualizar no Gráfico 2, a efetividade em relação a opinião do usuário foi de 66% para textos agressivos (queda de dois pontos percentuais em relação à rodada 1) e de 69% para textos não agressivos (um aumento de cinco pontos percentuais em relação à rodada 1. As 319 classificações dos usuários representam um percentual muito baixo em relação aos 40 mil *tweets* já treinados pelo KeepCalm, o que resultou em uma baixa taxa de aprendizado e a calibragem aparentemente não teve efeito.

Gráfico 2 - Número de acertos, erros e efetividade por categoria de classificação obtidos com a rodada de teste 2 do KeepCalm.



Fonte: Do Autor.

Novamente houve um grande número de avaliações questionáveis e que ficam em um limiar dúbio entre agressividade e não agressividade. Frases como “seu talento não chega nem aos pés do Fulano”, “minha namorada bebe cerveja como agua”, “nossa que decote tinha aquele vestido da garota” e “este final de semana tive visitas

demais", são apenas exemplos de frases que o usuário julga agressivo, mas que o KeepCalm considerou como não agressivo.

Ainda na rodada 2, um dos usuários relatou um fato interessante: o KeepCalm considerou em seus testes o termo "Bolsonaro 2018" como agressivo, o que gerou uma certa desconfiança sobre o viés político do classificador do KeepCalm. É claro que não há nenhuma preferência política desenvolvida nos algoritmos do software, mas em uma rápida pesquisa no Twitter pelo termo "Bolsonaro", percebeu-se que um bom número dos *tweets* relacionados ao político, continham termos agressivos para reforçar sua mensagem.

Esse comportamento permitiu concluir que o KeepCalm aprende por associação. Se uma determinada palavra não agressiva for frequentemente encontrada junto de outras que são agressivas, o KeepCalm pode calcular que provavelmente essa palavra é um "gatilho" para comportamentos agressivos. O *feedback* do usuário é importante nesse momento para realizar a calibragem de contraponto, informando ao algoritmo do software que a palavra isoladamente não representa necessariamente algo agressivo.

Neste ponto dos testes, vieram a tona alguns questionamentos: seria o usuário capaz de discernir corretamente o que é agressivo e o que não é? Será que ele não é amplamente influenciado pelo ambiente onde vive, pelas questões sócio-políticas? Será que a opinião dos usuários é a melhor forma de treinar algoritmos de classificação do KeepCalm? Será que o KeepCalm não está sendo induzido a algum viés político, após receber inúmeros *feedbacks* envolvendo textos contendo "Bolsonaro"? Certamente são questionamentos válidos que podem embasar propostas futuras.

6.5 Rodada de teste 3

Na tentativa de melhorar a eficiência do KeepCalm na classificação de textos e aumentar a taxa de aprendizado dos algoritmos, o Rodada 3 de testes foi otimizada com alguns ajustes.

Inicialmente, a base de dados para treinamento semissupervisionado foi reduzida de 40 mil para 2 mil *tweets* (sendo mil deles provavelmente agressivos e outros mil não agressivos). Com isso, o *feedback* do usuário poderia ter um impacto muito maior nas classificações futuras, já que a base de dados mais reduzida é mais facilmente influenciável pelas opiniões dos usuários de teste. O efeito colateral dessa redução impacta na limitação do campo amostral do KeepCalm, desconhecendo algumas palavras informadas pelos usuários.

A segunda alteração se deu na redução do ponto de corte de classificação agressiva, de 70% para 60%. Esta alteração foi necessária pois nas rodadas 1 e 2 detectou-se que parte das frases que são agressivas mas não claramente agressivas, obtiveram uma classificação agressiva com probabilidade entre 50% e 70%, mas que devido ao ponto de corte, foram repassadas ao usuário como “Não Agressivas”. Esperou-se com isso, um aumento na taxa de efetividade das frases parcialmente agressivas, as custas do aumento de alguns casos de falsos positivos.

Como prometido, a terceira rodada de testes amplia o escopo de análise para incluir uma comparação do desempenho e efetividade entre os dois algoritmos escolhidos: o Naive Bayes e o Regressão Logística. Ambos lidam com o mesmo conjunto de treinamento e o mesmo formato de documentos. A alteração de um para outro é uma simples questão de parametrização, já que a biblioteca NaturalNode abstrai todo o restante.

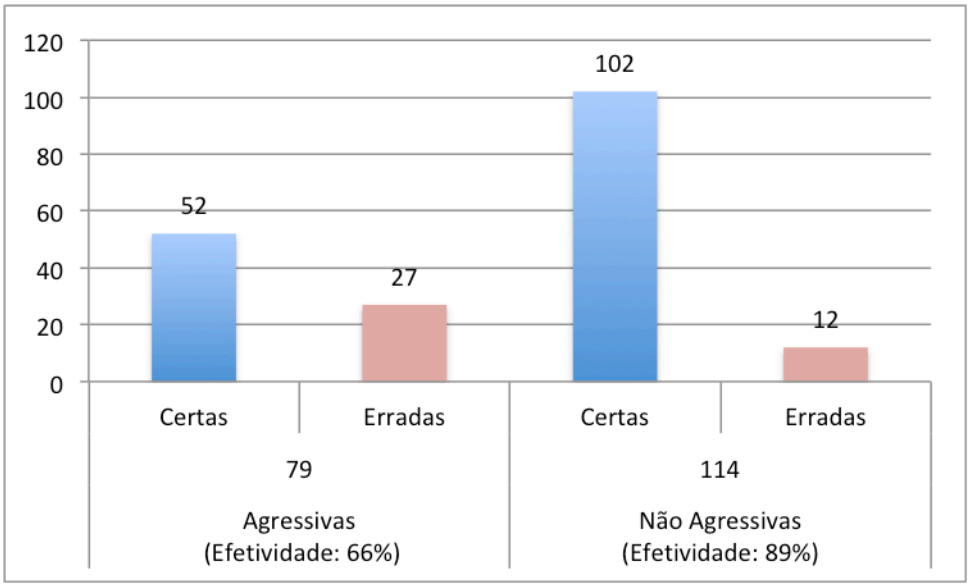
A primeira grande diferença entre os algoritmos foi a quantidade de recursos de hardware necessários para instanciar o serviço Rotulador utilizando o Regressão Logística. Nos testes 1 e 2, onde o conjunto de treinamento superou os 40 mil

documentos, não foi sequer possível iniciar o serviço devido ao alto consumo de memória RAM (superando os 8 gigabytes disponibilizados pelo container). Há a possibilidade de que o NaturalNode tenha problemas de otimização no algoritmo Regressão Logística, contudo, não é objetivo desta monografia avaliar tal implementação. Esse foi um dos fatores que levou a utilização do Naive Bayes nas duas primeiras rodadas dos testes.

Limitando o conjunto de treinamento em mil documentos para cada categoria de *Tweets*, ambos os algoritmos funcionam de maneira satisfatória, mas foi possível detectar a segunda grande diferença entre ambos os algoritmos de classificação: a velocidade de treinamento. Enquanto o Rotulador configurado para utilizar o modo Naive Bayes realiza o treinamento completo em cerca de 2 minutos, com o modo Regressão Logística esse tempo sobe para aproximadamente 15 minutos. Após esse período os recursos de hardware são liberados e o consumo de memória em ambos os modos oscila entre 450 e 500 megabytes.

Analisando os resultados, percebeu-se um grande aumento no percentual de eficiência no algoritmo Naive Bayes. O KeepCalm acertou 66% dos textos considerados “Agressivos” e 89% dos textos considerados “Não Agressivos” (do ponto de vista dos usuários). Em um total de 193 textos que tiveram um *feedback* por parte do usuário, o KeepCalm classificou-os corretamente em 154 vezes, gerando uma efetividade geral de 80%.

Gráfico 3 - Número de acertos, erros e efetividade por categoria de classificação utilizando o algoritmo Naive Bayes na rodada de teste 3 do KeepCalm.



Fonte: Do Autor

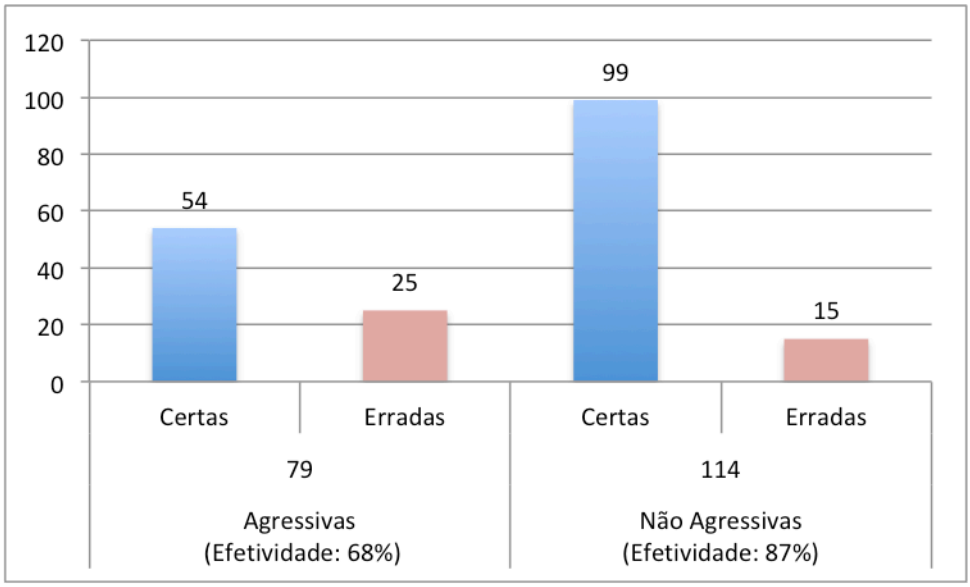
Por mais animadores que sejam esses resultados, é preciso considerar um fator que pode estar camuflando resultados e gerando falsos positivos. Muitas dessas 193 frases contêm expressões similares às utilizadas nas rodadas de teste anteriores. Após dezenas de frases em cada rodada de testes, o usuário foi perdendo a criatividade e o engajamento no teste, inserindo frases enviesadas ou semelhantes às que o KeepCalm havia avaliado incorretamente em etapas anteriores. É perceptível a necessidade de grupos de testes mais heterogêneos e diversificados para obter resultados mais confiáveis.

Mesmo assim, estes resultados comprovam a eficiência no aprendizado dos algoritmos, destacando uma evolução rodada a rodada. Acredita-se que com a liberação dos acessos a uma base de testadores maior e mais diversificada, seja possível melhorar ainda mais o campo amostral de expressões conhecidas pelo KeepCalm e aumentar a efetividade gradualmente.

Com o algoritmo de classificação por Regressão Logística e utilizando as mesmas 193 frases inseridas e avaliadas pelos usuários no teste 3, obteve-se um

resultado bastante similar. O KeepCalm acertou 68% dos textos considerados “Agressivos” e 87% dos textos considerados “Não Agressivos” (do ponto de vista dos usuários). Em um total de 193 textos, o KeepCalm classificou-os corretamente 154 deles, gerando uma efetividade geral de 79%.

Gráfico 4 - Número de acertos, erros e efetividade por categoria de classificação utilizando o algoritmo Regressão Logística na rodada de teste 3 do KeepCalm.



Fonte: Do autor.

Comparando a efetividade geral de cada um dos métodos de classificação, houve uma diferença percentual de apenas 2 pontos. Percebe-se neste caso que o algoritmo de classificação pouco influenciou nos resultados efetivos do KeepCalm, indicando que a qualidade do grupo de treino supervisionado é mais relevante que o método de classificação si. As maiores diferenças se mostraram no consumo de recursos computacionais, onde o algoritmo Naive Bayes se mostrou muito mais eficiente.

O NaturalNode é um projeto de software livre que permite que a comunidade auxilie no seu desenvolvimento, portanto, há a possibilidade de que o método de Regressão Logística ainda não tenha passado pelas devidas otimizações e, por isso, talvez esteja consumindo uma quantidade maior de recursos do que realmente poderia. Mesmo assim, a biblioteca mostrou-se muito útil e de fácil utilização, permitindo que o

desenvolvedor foque na resolução do seu problema específico sem a necessidade de desenvolver todas as ferramentas.

7 CONCLUSÃO

O Aprendizado de Máquina, como um todo e em específico o Processamento de Linguagem Natural, revelam uma série de informações fascinantes a respeito da computação e de como ela se relaciona com os humanos. Pode-se dizer, com algumas ressalvas, que atualmente já é possível interagir de forma incrivelmente natural com as máquinas. Estamos acostumados a simplesmente perguntar aos nossos dispositivos móveis, podemos interagir com interfaces inteligentes ou conversar com robôs vendedores em um site de compras e nem percebemos que por trás disso tudo, há um grande legado de estudo e tecnologia computacional.

Quando consideramos aspectos psicológicos do ser humano, como os sentimentos e emoções, nos deparamos com um desafio maior: Se nem nós humanos conseguimos diferenciar muitas vezes algo triste de algo alegre, como podemos esperar que um computador consiga? Por sorte, como foi descrito nesta monografia, há uma série de “gatilhos” que podem ser utilizados, padrões a serem mapeados e processamentos a serem efetuados. E nisso, um computador pode ser muito eficiente.

Há grandes empresas como Google, Microsoft e IBM efetuando altos investimentos em pesquisa todos os anos, na busca de soluções cada vez mais conectadas com os usuários. Percebeu-se que o aprendizado de máquina é chave para a geração de vantagens competitivas em empresas que precisam construir o perfil de seus clientes e desvendar seus comportamentos. Segundo José Papo, uns dos

evangelistas do Google Cloud no Brasil (GOOGLE, 2018D) é preciso três elementos chave que, unidos, garantem o sucesso de aplicações com o aprendizado de máquina: grandes conjuntos de dados, bons modelos e muito poder computacional (PAPO, 2017).

Um dos projetos mais relevantes da área atualmente é o IBM Watson, que permite a união da grande quantidade de informações com a capacidade computacional da IBM, gerando soluções para as áreas de pesquisa, detecção de riscos e falhas, comportamentos e decisões (IBM, 2017).

Outro ponto interessante que foi possível perceber durante as pesquisas é que a computação está cada vez mais presente em nossas vidas, ao mesmo tempo que há esforços gigantescos para que essa presença não seja notada, que computadores sejam inseridos em contextos humanos da forma mais despercebida possível. É necessário acima de tudo, que as funcionalidades tecnológicas não sejam um empecilho às vidas dos usuários. Elas devem ser incorporadas aos processos cotidianos de forma natural e transparente, como se estivéssemos lidando com outras pessoas.

Em relação ao tema proposto, quando falamos em “auxiliar no controle emotivo dos usuários”, entramos em um ponto muito pessoal. Será interessante poder contar com o *feedback* dos usuários e avaliar se a ferramenta proposta terá boa aceitação ou não. O KeepCalm não deve ser intrusivo, pelo contrário, ele deverá funcionar como suporte. O próprio usuário precisará compreender que uma relação respeitosa com outros usuários da internet é essencial para que todos possam se expressar.

Um dos objetivos pessoais envolvidos na realização deste trabalho, se refere ao aprendizado de novas ferramentas e a utilização de uma linguagem nova. Apesar de muitas implementações e ferramentas na área de PLN serem desenvolvidas em Python, a escolha pelo Javascript se deu devido ao fato de muitas das funcionalidades desenvolvidas para o KeepCalm, poderem ser utilizadas em outras ferramentas que envolvem a comunicação de serviços, servidores *web*, APIs e páginas de internet. Foi de fato, um grande aprendizado (a partir de muitos erros), afinal, não me considero um

bom desenvolvedor de software, mas com certeza, um desenvolvedor melhor do que era antes de iniciar o trabalho.

Durante esse processo de aprendizado, um aspecto no desenvolvimento da Extensão do KeepCalm para o Google Chrome me chamou a atenção: ela possui a capacidade de analisar todo o conteúdo digitado pelo usuário em qualquer página da internet que ele esteja navegando. Para isso, basta uma solicitação de permissão que pode muito bem passar desapercebida por um usuário um pouco desatento. O KeepCalm conta com mecanismos de proteção que evitam o acesso a campos destinados a informações sensíveis, como senhas e dados pessoais. Mesmo assim, permanece em nossas mentes um questionamento um tanto perturbador: será que outras extensões não poderiam se aproveitar dessa característica do Google Chrome, abusando das permissões e acessando dados confidenciais como a senha do seu banco?

Esse mesmo problema ocorre em outros dispositivos e software, uma vez que o acesso a informações importantes dos usuários é ativado no momento em que o usuário acessa determinado recurso do aplicativo ou site e precisa dar algo em troca, ou ainda quando aceita termos de uso muito extensos e difíceis de ler. Durante a escrita do presente trabalho veio à tona a notícia de venda de dados pessoais coletados dentro do Facebook por uma empresa com supostos fins acadêmicos, a Cambridge Analytica, mas que no final se destinaram ao estudo e foco marqueteiro na campanha de Donald Trump, nas últimas eleições dos Estados Unidos (OFICINADANET, 2018). Caso o KeepCalm ou um serviço derivado dele venha a ser utilizado em larga escala ou de forma comercial, é preciso estar atento a essas questões, deixando claro aos usuários quais os dados que são coletados, como eles são utilizados e como serão mantidos.

Com relação aos testes, a primeira grande conclusão obtida está na falta de aspectos lúdicos do KeepCalm. Durante a execução das rodadas de testes, houve uma queda gradual no engajamento dos usuários, que reduziram a quantidade e qualidade dos textos classificados. Acredita-se que um dos principais motivos seja a falta de

recompensas que a ferramenta proporciona a cada utilização, já que o usuário não fica realmente contente ao ser repreendido enquanto navega na internet. Mas o KeepCalm, mesmo com essa característica punitiva, pode ser útil na educação, por exemplo, de usuários que precisam de limites em seus comportamentos sociais. Como proposta de desenvolvimento futuro, deve-se elaborar mecanismos que adicionem mecanismos de jogos ao aplicativo, de modo a tornar sua utilização mais agradável e recompensadora.

A segunda conclusão com relação aos testes é referente a dificuldade, tanto computacional como também humana, em decernir textos ou frases que não são claramente agressivos e nem claramente não agressivos. Frases com ironia, preconceito ou com sentidos agressivos do ponto de vista cultural, tiveram valores de efetividade muito próximos dos 50%, beirando a aleatoriedade. Por exemplo as frases “este trabalho é lixo” e “este papel é lixo” são opostas no sentido de agressividade e, a menos que o KeepCalm já tenha efetuado uma classificação com frases similares e tenha recebido *feedback* pelo usuário, poderão gerar classificações incorretas pela ferramenta. Esse aspecto acaba levando a um dilema técnico: é preciso aumentar a quantidade de dados de treinamento para aumentar a diversidade de conhecimentos do classificador, mas ao fazer isso, essa quantidade de dados diminui a relevância dos *feedbacks* dos usuários, dificultando o aprendizado de novos textos.

Esse problema é agravado ainda mais quando adicionados elementos de opinião aos textos. Após uma análise no conjunto de treinamento gerado com as classificações do KeepCalm, foi possível perceber que frases com estruturas bastante similares, foram classificadas de forma diferente pelos usuários, isso porque agressividade pode ser uma questão de opinião. O ambiente em que vivemos, os conteúdos que consumimos e, principalmente, a cultura em que estamos inseridos, afetam significativamente a nossa capacidade de identificar o que é agressivo ou não. Alguns palavrões foram adicionados ao vocabulário diário de muitas pessoas, como “porra”, “merda” e “bosta” e são utilizadas para intensificar um determinado sentimento na oração. Um exemplo encontrado na base de dados foi “Merda. Estou atrasado!”, onde o usuário a classificou como “Não Agressiva”. Tudo isso acaba gerando regras contraditórias dentro do

conjunto de treinamento do KeepCalm e pode impactar diretamente nos resultados futuros.

Isso nos leva a terceira conclusão: talvez o usuário final não seja o elemento ideal para efetuar o treinamento supervisionado da ferramenta. Será que nós, indivíduos brasileiros médios, com a nossa formação política e educacional, temos reais condições de destacar o que é agressivo daquilo que não é? Será que o nosso discernimento não pode ser influenciado pelos aspectos sociais e políticos do ambiente em que vivemos? Esses questionamentos trazem consigo os receios de que o KeepCalm possa ser enviesado de acordo com o que os usuários acham correto, o que pode não ser o cenário ideal. Talvez sejam necessários estudos mais aprofundados nas áreas sociais e psicológicas de modo a criar regras mais claras daquilo que é agressivo ou não, mas novos termos violentos serão criados reiniciando a discussão. No futuro, o KeepCalm precisará se moldar de acordo com esses cenários e receber otimizações constantes de modo a considerar o maior número possível de nuances da nossa comunicação textual.

Além do problema da opinião, os usuários tentem a se repetir na estrutura de frases. Isso fez com que o KeepCalm se tornasse muito eficaz nas frases onde o testador apenas trocou uma palavra de lugar ou apenas alterou de um palavrão para outro. Em um cenário de aplicação em massa da Extensão, é bem provável que o KeepCalm se depare com classificações muito mais diversificadas cujo o conteúdo ainda não tenha sido treinado, reduzindo a efetividade geral da classificação em um primeiro momento, mas aumentando posteriormente com o feedback desses usuários.

Mesmo com todos esses problemas descritos, o KeepCalm foi o produto resultante deste trabalho visando o alcance dos objetivos propostos. A jornada de construção desta monografia passou naturalmente pela pesquisa literária na busca do entendimento das emoções, e de como o processamento de linguagem natural auxilia na sua detecção computacional. Esse aprendizado permitiu focar no desenvolvimento de uma solução capaz de minerar informações agressivas da internet e aprender com

isso, usando esse conhecimento para auxiliar o usuário de internet a controlar e entender suas emoções antes que sejam compartilhadas a outras pessoas.

Conclui-se, então, que o KeepCalm abrange o tema e alcança os objetivos propostos no presente trabalho. Mesmo assim, isso não isenta a necessidade de melhoria da ferramenta, principalmente nos aspectos lúdicos. Os usuários precisam sentir-se confortáveis ao utilizar ferramentas desse tipo, sendo recompensados pelo bom comportamento gerado e desafiados a melhorar. Somente assim, a ferramenta poderá realmente conscientizar massivamente os usuários da internet sem que se sintam constantemente repreendidos e deixem de usá-la.

É possível elencar algumas possibilidades para trabalhos e desenvolvimentos futuros envolvendo o próprio KeepCalm ou outras soluções da área de PLN.

- Efetuar trabalhos conjuntos, envolvendo as áreas da linguística, psicologia e computação, para o desenvolvimento de soluções que considerem um maior número de características emotivas e textuais, aumentando a efetividade das soluções;
- Soluções como o KeepCalm, podem ser implantadas de forma comercial em instituições privadas de modo a auxiliar no controle emotivo dos processos de comunicação. É possível, por exemplo, garantir que e-mails e contatos via *chat* sejam minimamente adequados e evitem conflitos com colaboradores ou clientes.
- Abranger o desenvolvimento do KeepCalm de modo a considerar outros algoritmos de classificação, melhorar as metodologias de testes levando como aprendizado os erros efetuados neste trabalho, migrar a extensão para outros navegadores ou plataformas e melhorar a experiência geral dos usuários.
- Além do processamento de emoções, é possível utilizar o PLN para o desenvolvimento de outras soluções envolvendo a detecção do humor de usuários ou clientes, perfis psicológicos, da ética e do caráter, da intenção de compra, etc.

Por fim, é possível considerar que ferramentas são apenas ferramentas. Ou seja, elas podem ser utilizadas tanto para a elaboração de ótimas soluções como para péssimas iniciativas. O pensamento geral dos utilizadores da internet é que precisa ser revisto e direcionado a um modo mais harmonioso. O KeepCalm foi desenvolvido de modo a comprovar que soluções desse tipo podem ser desenvolvidas sem muita dificuldade, aproveitando-se de outras ferramentas e do avanço na computação.

REFERÊNCIAS

AGÊNCIA MESTRE. **Stop Words**: Como Funcionam Palavras de Parada? Disponível em: <http://bit.ly/2yoiYJK>. Acesso em: 16 Out. 2017.

ALPAYDIN, Ethem. **Introduction to Machine Learning**. The MIT Press. 2ª Edição. Pg 31-32. 2010.

APPLE, Developer. **Build more intelligent apps with machine learning**. Disponível em: <https://developer.apple.com/machine-learning/> 2017. Acesso em: 21 Ago. 2017.

AURÉLIO, Dicionário. **Semântica**. Disponível em: <https://dicionariodoaurelio.com/semantica> 2016. Acesso em: 21 Ago. 2017.

AURÉLIO, Dicionário. **Sintaxe**. Disponível em: <https://dicionariodoaurelio.com/sintaxe> 2017. Acesso em: 21 Ago. 2017.

AVAST. **PC Trends Report:**Top 7 facts about PCs in 2017. Disponível em <https://blog.avast.com/pc-trends-report-top-7-facts-about-pcs-in-2017>. 2017. Acesso em: 01 Jun. 2018.

BIDERMAN, M. T. C. **Unidades complexas do léxico**. In: Rio-Torto, G.; Figueiredo, O. M; Silva, F. (Org.). Estudos em Homenagem ao Professor Doutor Mário Vilela. 1ª ed. Porto, Portugal: Faculdade de Letras da Universidade do Porto, 2005, v. II, p.747-757.

BRADLEY, M. M.; LANG, P. J. **Handbook of Psychophysiology**. Cambridge University Press. 2ª Edição. Pg 581. 2006.

BRAND24, **Social Media Monitoring Tool**. Disponível em: <https://brand24.com>. Acesso em: 21 Ago. 2017.

BRITTO, Ilma A. Goulart de Souza; ELIAS, Paula Virgínia Oliveira. **Análise comportamental das emoções**. Psicol. Am. Lat. n.16 México jun. 2009

CGI.BR, TIC Kids Online Brasil 2015: **Pesquisa sobre o Uso da Internet por Crianças e Adolescentes no Brasil**: Núcleo de Informação e Coordenação do Ponto BR. Pg 170. 2015.

CHIBENI, Silvio Seno. Algumas observações sobre o método científico. Departamento de Filosofia, IFCH, Unicamp. Pg 3. 2006.

CHROME. **Extensions:** maximize the Chrome browsing experience. Disponível em: <https://developer.chrome.com/home>. Acesso em 05 Mai. 2018

COPPIN, Ben. Inteligência Artificial. LTC. Pg 03-10. 2010.

CORNELIUS, R. R. **Theoretical approaches to emotion**. ISCA Workshop on Speech and Emotion. Pg 3–10. 2000.

COSTA, Hernani; MACEDO, Luis. **Affective Computing**. ATCM State of the Art, Theoretical Report. 2012.

DA SILVA, Leandro A; PERES, Sarajane M; BOSCARIOLI, Clodis. **Introdução à Mineração de Dados com aplicações em R**. 1ª Ed. Elsevier. 2016.

DARWIN C. R. **The expression of the emotions in man and animals**. 1ª Edição. Pg 60-61. Disponível em: <http://bit.ly/1c9Y4YJ>. 1872. Acesso em: 22 Ago. 2017.

DEVMEDIA. Introdução ao MongoDB. Disponível em: <https://www.devmedia.com.br/introducao-ao-mongodb/30792>. Acesso em 30 Mar. 2018.

DOMINGOS, Pedro. **O Algoritmo Mestre**: Como a busca pelo algoritmo de Machine Learning definitivo recriará nosso mundo. Novatec. Pg 30. 2016.

EXPRESSJS. Fast, unopinionated, minimalist web framework for Node.js. Disponível em <http://expressjs.com>. Acesso em: 23 Mai. 2018.

FACEBOOK, **Newsroom**. Disponível em: <https://newsroom.fb.com/company-info/>. 2017. Acesso em: 17 Ago. 2017.

FGV, Fundação Getulho Vargas. **28ª Pesquisa Anual do Uso de TI**, 2017. Disponível em: <http://eaesp.fgvsp.br/ensinoeconhecimento/centros/cia/pesquisa>. Acesso em: 28 Set. 2017.

FULTON, Lawrence V.; MENDEZ, Francis A.; BASTIAN, Nathanael D.; MUSAL, R. **Confusion Between Odds and Probability, a Pandemic?** Journal of Statistics Education. Volume 20. 2012.

GITHUB. PT_BR Postagger. Disponível em: <https://github.com/diegodorgam/postagger>. Acesso em: 10 Mai. 2018B.

GITHUB. The state of the Octoverse 2017. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 27 Mar. 2018.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. The MIT Press. Pg 463. 2016.

GOOGLE, Chrome. **Use um navegador da Web gratuito e mais rápido**. Disponível em: <https://www.google.com/chrome/browser/desktop/index.html> 2017. Acesso em: 18 Out. 2017B.

GOOGLE, Cloud. What is machine learning? Dispoível em: <https://cloud.google.com/what-is-machine-learning/>. Acesso em: 08 Jun. 2018D.

GOOGLE, ReCaptcha. Tough on bots Easy on humans. Disponível em: <https://www.google.com/recaptcha/intro/android.html>. Acesso em: 27 Mar. 2018C.

GOOGLE, Research. **Natural Language Processing**. Disponível em: <https://research.google.com/pubs/pub45791.html> 2017. Acesso em: 21 Ago. 2017A.

HASTIE, Trevor; TIBSHIRANI, Robert; FRIEDMAN. Jerome. **The Elements of Statistical Learning**: Data Mining, Inference, and Prediction. 2º Ed. Springer. 2008.

HBO. **Westworld**. Disponível em: <http://www.hbo.com/westworld/about/index.html>. Acesso em: 05 Set. 2017.

IBM, **DeveloperWorks**; The Art of Tokenization. Disponível em: <https://ibm.co/1f0Zz0G> 2013. Acesso em: 18 Set. 2017.

IBM. **The Chatbot Never Sleeps**: How We Created a Chatbot Integration with Mendix That Enables 24/7 Customer Service. Disponível em: <https://ibm.co/2v9npU6> 2017. Acesso em: 21 Ago. 2017.

J. J. A. Van Berkum, D. van den Brink, C. M. J. Y. Tesink, M. Kos and P. Hagoort, **The Neural Integration of Speaker and Message**. Journal of Cognitive Neuroscience, vol. 20, no. 4, pp. 580-591, 2008.

JELASTIC. Platform-as-a-Service. Disponível em: <https://jelastic.com>. Acesso em: 31 Mai. 2018.

KAYO, Roman. O que é early adopter e earlyevangelist? Disponível em: <http://ramonkayo.com/conceitos-e-metodos/o-que-e-early-adopter-e-earlyvangelist>. Acesso em: 18 Abr. 2018.

KOVACICH, Gerald L.; HALIBOZEK, Edward P. **The Manager's Handbook for Corporate Security**: Establishing and Managing a Successful Assets Protection Program. 1ª ed. Elsevier Science. 2003b.

MARGOTTY, Felício Wessling; MARGOTTY, Rita C. M. Ferreira. **Morfologia do Português**. 2º Período. UFSC. 2011.

MCCALLUM, Andrew. **Introduction to Natural Language Processing**. CICS, 2007.

MICROSOFT, Technet. **Cloud-Scale Text Classification with Convolutional Neural Networks on Microsoft Azure**. Disponível em: <http://bit.ly/2kqRahL/> 2017. Acesso em: 21 Ago. 2017.

MLODINOW, Leonard. **O Andar do Bêbado**: Como o acaso determina nossas vidas. Zahar. Versão Kindle. 2009.

MONGOOSE. Elegant mongodb object modeling for node.js, Disponível em: <http://mongoosejs.com>. Acesso em: 08 Mai 2018.

MOZILLA, Developer. MVC Architeture. Disponível em: <https://mzl.la/2KSMOtE>. Acesso em: 09 Mai. 2018.

NATURALNODE. **Natural Node**. Disponívem em: <https://github.com/NaturalNode/natural>. Acesso em: 16 Out. 2017.

NLTK. **Natural Language Toolkit**. Disponível em: <http://www.nltk.org>. Acesso em: 16 Out. 2017.

NODE.JS. About Node.js. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 27 Mar. 2018

NUNES, M.G.V.; DIAS SA SILVA, B.C.; RINO, L.H.M.; OLIVEIRA, Jr., O.N.; MARTINS, R.T.; MONTILHA, G. **Introdução ao Processamento das Línguas Naturais**. Notas Didáticas do ICMC, N. 38. p 6. 1999.

NYU, **Penn Part of Speech Tags**. Disponível em: <https://cs.nyu.edu/grishman/jet/guide/PennPOS.html>. Acesso em: 18 Set. 2017.

OFICINADANET. Endendendo o escândalo Facebook e Cambridge Analytica em 5 minutos. Disponível em: <https://bit.ly/2HsUPCu>. Acesso em: 02 Jun. 2018.

OMG, Object Management Group. **Unified Modeling Language 2.5**. Disponível em: <http://www.omg.org/spec/UML/2.5>. Acesso em: 23 Set. 2017.

PAPO. José. Por que o Google Cloud Platform é diferente. Disponível em: <https://pt.slideshare.net/jpapo/por-que-google-cloud-diferente> 2017. Acesso em: 08 Jun. 2018.

PEREIRA, Silvio do Lago; **Processamento de Linguagem Natural**. IME-USP, 2013.

PETTER, Margarida M. T. Morfologia. In: FIORIN, José Luiz (Org.). I**ntrodução à Linguística**: princípios de análise. São Paulo: Contexto, 2003.

PICARD, R. W. **Affective Computing**. Cambridge, EUA: The M.I.T. Press, 1997.

PLUTCHIK, P. **A Nature of Emotions**. American Scientis vol. 89. Pg 349. 2001.

PRESSMAN, Roger S. **Engenharia de Software**. São Paulo: Makron Books, 1995.

ROACHE, **Naughty words**: What makes swear words so offensive? It's not their meaning or even their sound. Is language itself a red herring here? Disponível em: <http://bit.ly/1RhRdna> 2016. Acesso em: 30 Ago. 2017.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência artificial**. Rio de Janeiro: Elsevier. 3ª Edição. 2013.

SAYAD, Saed. **An Introduction to Data Mining**. Disponível em: <http://chem-eng.utoronto.ca/~datamining/dmc/naive_bayesian.htm>. Acesso em: 05 Out. 2017A.

SAYAD, Saed. **An Introduction to Data Mining**. Disponível em: <http://chem-eng.utoronto.ca/~datamining/dmc/logistic_regression.htm>. Acesso em: 05 Out. 2017B.

SILVA, Alberto; VIDEIRA, Carlos. **UML**: Metodologias e Ferramentas Case. Centro Atlântico. 1ª Edição. 2001.

SOUZA, Renato Rocha. **Uma proposta de metodologia para escolha automática de descritores utilizando sintagmas nominais**. UFMG. 2005.

STAATS, A. W. **Psychological Behaviorism and Behaviorizing Psychology**. The Behavior Analyst. University of Hawaii. Pg 102-103. 1994.

TARTARUS. The Porter Stemming Algorithm. Disponível em: <https://tartarus.org/martin/PorterStemmer/>. Acesso em: 08 Mai. 2018.

TURING, A. M. **Computing machinery and intelligence**. Mind, 59, 433-460. 1950. Disponível em: <http://www.loebner.net/Prizef/TuringArticle.html>. Acesso em: 05 Set. 2017

TWITTER, **Business**: Twitter Basics. Disponível em: <https://business.twitter.com/en/basics.html>. Acesso em: 12 Out. 2017A.

TWITTER, **Developer**: Search Tweets. Disponível em: <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets>. Acesso em: 12 Out. 2017B.

UNIVERSITY OF READING, **Turing test success marks milestone on computing history**. Disponível em: <http://www.reading.ac.uk/news-and-events/releases/PR583836.aspx> 2014. Acesso em: 05 Set. 2017.

W3SCHOOLS. **JSON:** Introduction. Disponível em: <https://www.w3schools.com/js/js_json_intro.asp>. Acesso em: 24 Mai. 2018.

YOUTUBE. **Press Youtube**: YouTube by the numbers. Disponível em: <https://www.youtube.com/yt/about/press/> 2017. Acesso em: 21 Ago. 2017.

ANEXO A – Formulário de Testes Projeto KeepCalm

Trabalho de Conclusão de Curso - Maiquel Jardel Ludwig

Parabéns! Você foi escolhido para fazer parte deste incrível projeto. Eu sou o Maiquel e precisarei da sua ajuda para colocar em prática o meu projeto desenvolvido durante a execução do meu trabalho de conclusão de curso intitulado como: “Identificação computacional de emoções negativas em textos da internet: proposta de apoio ao usuário para controle emotivo”.

O KeepCalm é uma extensão para o navegador Google Chrome que “captura” os textos inseridos pelo usuário, envia as informações para um servidor remoto que, através de processamento de linguagem natural aliado a processos de inteligência artificial, retorna ao usuário a informação referente ao contexto emotivo do texto inserido, podendo ser classificado como “Agressivo” ou “Não Agressivo”. Inicialmente o algoritmo foi treinado com 40 mil *tweets* de forma semissupervisionada.

Importante:

1. O KeepCalm não coleta dados em campos do tipo e-mail, senha, telefone, endereço e valores numéricos.
2. Todos os dados são enviados de forma anônima para o processamento em nosso servidor, não armazenamos a sua sessão e nem o seu IP. A instalação não exige nenhuma identificação ou dado pessoal.
3. Participando deste teste você concorda em disponibilizar o seu *feedback* para fins acadêmicos. Ficarei muito contente em compartilhar os resultados após o período de testes

Mas, como você pode me ajudar? Simples, durante os próximos 30 dias, você me ajudará com a calibragem dos algoritmos de classificação, realizando 3 rodadas de testes muito simples, que deverão ocupar somente uns 10 minutos da sua vida. O cronograma das rodadas de teste será:

- 1ª Rodada – De 09/04/2018 a 15/04/2018
- 2ª Rodada – De 20/04/2018 a 26/04/2018
- 3ª Rodada – De 01/05/2018 a 08/05/2018

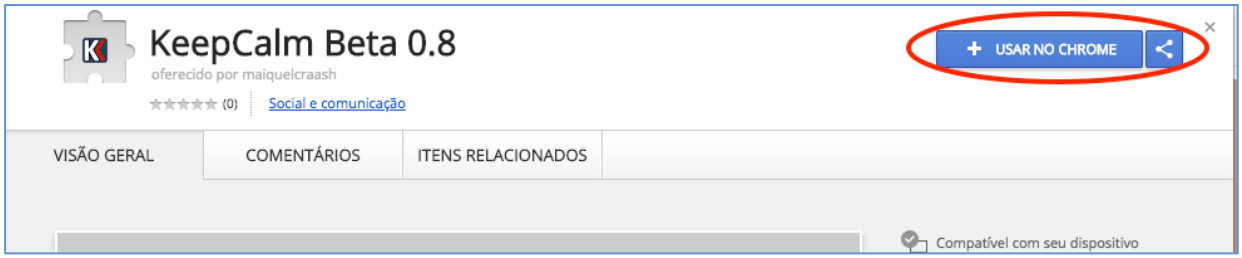
O período entre as rodadas será necessário para efetuar os processos de calibragem do algoritmo, coletar os dados coletados, analisá-los e documentar os resultados obtidos. Você será avisado ao início e ao fim de cada uma das rodadas.

Para iniciar os testes, siga os passos a seguir.

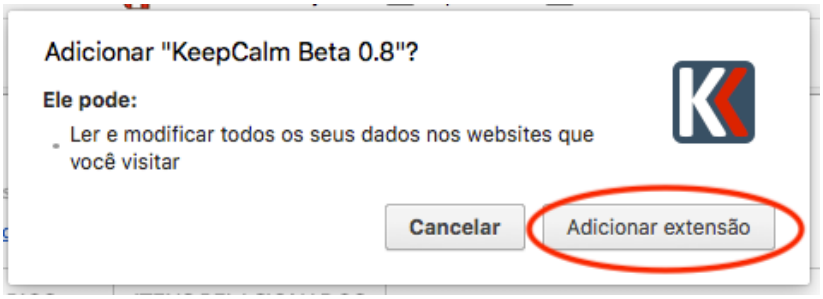
Passo 1 – Preencha a tabela abaixo. Esta é de longe a parte mais chata do teste, mas fique tranquilo que você só precisará fazer isso uma única vez, tenho certeza de que você consegue. Você terá que elaborar frases curtas ou longas de acordo com as 3 categorias descritas e exemplificadas abaixo.

#	**Categoria 1** **Claramente não agressivas** (Pode conter palavrões, porém em contexto não agressivo) Ex1: Gosto muito de morar nesta casa Ex2: Ela é uma criança de crescimento retardado.	**Categoria 2** **Agressivas** (Não deve conter palavrões explícitos) Ex1: Este lixo só pode ser criação sua! Ex2: Deixe de ser retardado Pedro.	**Categoria 3** **Claramente Agressivas** (Palavrões liberados) Ex1: Quem chamou esse idiota para participar? Ex2: Nunca vi pessoa mais burra, vai tomar no cu.
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

Passo 2 – Instale a extensão KeepCalm clicando **aqui** (abra o link utilizando o Google Chrome). Clique sobre o botão "**Usar no Chrome**".

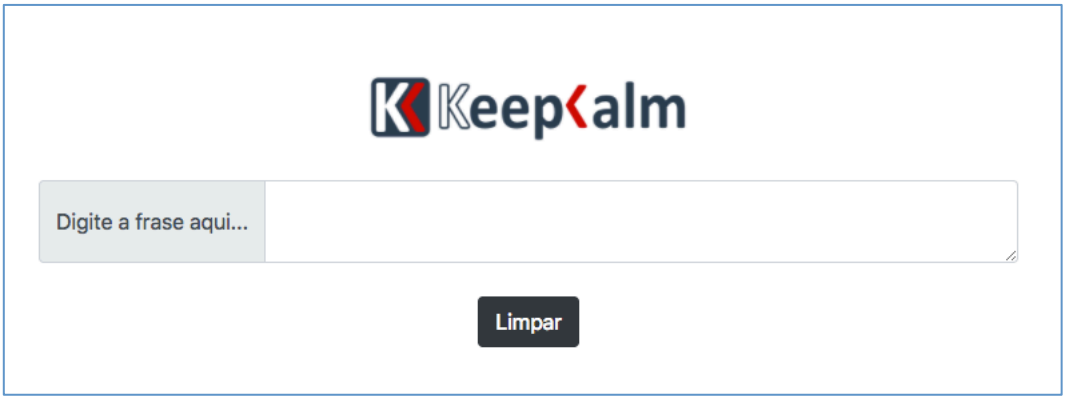


Passo 3 – Confirme a instalação da extensão clicando sobre o botão "**Adicionar Extensão**". Após a instalação, já será possível acessar o *popup* do KeepCalm, clicando sobre o botão localizado no canto superior direito do Chrome.



Passo 4 – Agora que a extensão foi instalada no seu navegador, é possível realizar o teste propriamente dito. Para isso, acesse a seguinte página de teste (utilizando ainda o Google Chrome): https://keepcalm.acml.com.br/

Passo 5 – No único campo presente na página, digite a primeira fase da categoria 1, presente na tabela preenchida no Passo 1. Você pode copiar e colar, desde que adicione um espaço ao final da frase, para que o evento de digitação seja acionado.



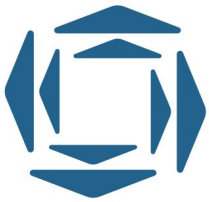
Passo 6 – Aguarde de 3 a 5 segundos para que o KeepCalm consiga processar o texto inserido e chegar a uma conclusão. Repare que o ícone da extensão no topo da tela é alterado para um circulo vermelho ou verde, de acordo com a classificação do texto inserido. Clique sobre o botão da extensão para abrir o *popup* conforme a imagem a seguir.

Passo 7 – Clique sobre o botão “Concordo” ou “Não concordo” de acordo com a sua própria opinião. Lembre-se, você está auxiliando a calibrar o algoritmo de classificação, ou seja, é bastante provável que a extensão erre em algumas classificações, principalmente na segunda categoria de frases.

PRONTO! Repita os passos 5, 6 e 7 para cada uma das frases do passo 1 e não se esqueça de dar seu feedback em cada uma delas.

Fique a vontade para utilizar a extensão o quanto quiser e dar seu *feedback* sempre que possível. Obrigado pela Ajuda!

Maiquel Jardel Ludwig
Graduando em Engenharia da Computação
Universidade do Vale do Taquari – UNIVATES



UNIVATES

R. Avelino Talini, 171 | Bairro Universitário | Lajeado | RS | Brasil