

Emanuel Gilvan Souza Lima Júnior

Uma solução para fusão de dados em plataformas de resolução de conflitos

São Luís - MA, Brasil

2020

Emanuel Gilvan Souza Lima Júnior

Uma solução para fusão de dados em plataformas de resolução de conflitos

Dissertação apresentada ao programa de Mestrado Profissional em Engenharia da Computação e Sistemas da Universidade Estadual do Maranhão como pré-requisito para obtenção do título de Mestre em Engenharia da Computação e Sistemas.

Universidade Estadual do Maranhão – UEMA

Centro de Ciências Tecnológicas

Programa de Pós-Graduação em Engenharia de Computação e Sistemas

Orientador: MSc. Antonio Fernando Lavareda Jacob Junior

Coorientador: Dr. Fábio Manoel França Lobato

São Luís - MA, Brasil

2020

Lima Júnior, Emanuel Gilvan Souza.

Uma solução para fusão de dados em plataformas de resolução de conflitos / Emanuel Gilvan Souza Lima Júnior. – São Luís, 2020.

66 f

Dissertação (Mestrado) – Curso de Engenharia de Computação e Sistemas, Universidade Estadual do Maranhão, 2020.

Orientador: Prof. Me. Antonio Fernando Lavareda Jacob Junior

Coorientador: Prof. Dr. Fábio Manoel França Lobato.

1. *Web Scraping*. 2. Mineração de texto. 3. Gerenciamento de reclamações. I. Título

CDU: 004.6

Emanuel Gilvan Souza Lima Júnior

Uma solução para fusão de dados em plataformas de resolução de conflitos

Dissertação apresentada ao programa de Mestrado Profissional em Engenharia da Computação e Sistemas da Universidade Estadual do Maranhão como pré-requisito para obtenção do título de Mestre em Engenharia da Computação e Sistemas.

Trabalho aprovado. São Luís - MA, Brasil, 25 de agosto de 2020:



Dr. Diego Lisboa Cardoso

UFPA

Examinador externo à instituição



Dr.ª. Eveline de Jesus Viana Sá

IFMA

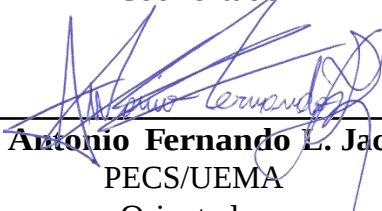
Examinadora interna



Dr. Fábio Manoel França Lobato

UFOPA

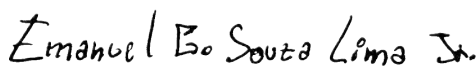
Coorientador



MSc. Antonio Fernando L. Jacob Jr.

PECS/UEMA

Orientador



Emanuel Gilvan Souza Lima Júnior

Mestrando

São Luís - MA, Brasil

2020

A HaShem, Sandra, Ali-Utta, Creusa e aos meus irmãos.

Agradecimentos

Ao Senhor, por me permitir, durante este mestrado, conhecer mais de Sua verdade, pois toda verdade é verdade de Deus.

Agradeço à toda minha família pelo apoio e cobrança necessários, e à minha amada Paulinha, o mais belo presente desta jornada, pelo encorajamento e motivação.

Aos meus amigos de infância, da academia e da Igreja pelo apoio e tolerância à ausência necessária para o desenvolvimento deste trabalho.

Ao professor Antonio F. L. Jacob Jr. pela orientação, direcionamento e ensinamentos, por me ajudar com recursos quando precisei, sem os quais eu não conseguiria efetuar matrículas nem desenvolver este trabalho. Dentre suas benesses, agradeço também pela paciência e compreensão, e por ser minha grande referência na área. Estendo aqui os agradecimentos a todo o corpo docente da UEMA e à FAPEAD por fomentar o desenvolvimento deste trabalho.

*“Porque Deus é o que opera em vós
tanto o querer
como o efetuar,
segundo a sua boa vontade.”
(Paulo de Tarso)*

Resumo

A crescente quantidade de usuários ativos na internet e em redes sociais acompanha um grande número de dados, sendo gerados a cada segundo. Esse fato pode ocasionar um problema para as empresas, uma vez que os consumidores estão buscando os recursos de mídias sociais para expor os seus problemas com fornecedores de produtos e/ou serviços. Tal contexto beneficiou o crescimento da utilização de plataformas de registro de reclamações e resolução de conflitos, tais como o Consumidor.gov.br e ReclameAQUI. Nota-se, porém, a ausência de uma ferramenta computacional que ofereça suporte às organizações no manejo e integração dos dados disponibilizados por diferentes plataformas de registro de reclamações e resolução de conflitos. Neste contexto, este trabalho propõe o desenvolvimento de um sistema computacional para extração, fusão e análise dos dados contidos nas plataformas supracitadas, a fim de obter conhecimento a ser utilizado no suporte à gerência destas organizações. Nos resultados obtidos deste trabalho encontra-se um *dashboard* para visualização gráfica dos dados obtidos nas análises de sentimentos, de distribuições quantitativas, temporais e geográficas, modelagem de tópicos e nuvem de palavras. O sistema computacional ainda permite a exportação de todos os dados coletados por um usuário.

Palavras-chave: *Web Scraping*. Mineração de texto. Gerenciamento de reclamações.

Abstract

The growing number of active users on the internet and on social networks is followed by a large number of data being generated every second. Also consumers are looking for social media resources to expose their problems with suppliers of products and/or services, which has increased the use of complaint resolution platforms, such as Consumidor.gov.br and ReclameAQUI. It is noted, however, the absence of a computational tools that supports organizations to handling and integrating the data obtained by different complaint registration and conflict resolution platforms. In this context, this work proposes the development of a computational system for extraction, fusion and analysis of the data contained in these platforms, in order to discovery knowledge, which could support the management of these organizations. In the results obtained from this work shows a dashboard for graphical visualization of the data obtained from sentiment analysis, quantitative, temporal and geographical distributions, topic modeling and word cloud. The computer system also allows the export of all data collected by a user.

Keywords: Web scraping. Text mining. Complaint management.

Lista de ilustrações

Figura 1 – O Modelo <i>JDL</i> (LYTRIVIS; THOMADIS; AMDITIS, 2009, tradução nossa)	28
Figura 2 – <i>Framework</i> para aplicação de fusão de informação a mineração de opinião (BALAZS; VELÁSQUEZ, 2016, tradução nossa).	30
Figura 3 – Arquitetura do sistema no trabalho de TEIXEIRA et al. (TEIXEIRA et al., 2018)	31
Figura 4 – Arquitetura do sistema no trabalho de NETTO et al. (NETTO et al., 2019)	32
Figura 5 – Relação entre tópicos das reclamações no trabalho de SOUSA et al. (SOUSA et al., 2020).	33
Figura 6 – Modelo de processo <i>CRISP-DM</i>	35
Figura 7 – Página inicial da plataforma Consumidor.gov.br	37
Figura 8 – Página inicial da plataforma ReclameAQUI	38
Figura 9 – Arquitetura da solução proposta.	43
Figura 10 – Opções de filtros para dados a serem exibidos	50
Figura 11 – Telas relacionadas à busca e seleção da empresa.	57
Figura 12 – Visualização gráfica de distribuição das reclamações.	58
Figura 13 – Análise de sentimentos e distribuição geográfica.	58
Figura 14 – Nuvens de palavras por tópicos.	59

Lista de tabelas

Tabela 1 – Avaliação das empresas na plataforma ReclameAQUI	38
Tabela 2 – Dados e filtros presentes nas plataformas Consumidor.gov.br e ReclameAQUI	39
Tabela 3 – Propriedades de relatos comuns às duas plataformas	39

Lista de abreviaturas e siglas

API	<i>Application Programming Interface</i>
CEJUSC	Centro Judiciário de Solução de Conflitos e Cidadania
CORBA	<i>Common Object Request Broker Architecture</i>
CSS	<i>Cascading Style Sheets</i>
CSV	<i>Comma-separated values</i>
DOM	<i>Document Object Model</i>
FDD	<i>Feature-Driven Development</i>
FLIR	<i>Forward Looking Infra-Red</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IANA	<i>Internet Assigned Numbers Authority</i>
JDL	<i>Joint Directors of Laboratories</i>
JSON	<i>JavaScript Object Notation</i>
LD	<i>Lean Development</i>
NoSQL	<i>Not Only SQL</i>
REST	<i>Representational State Transfer</i>
RUP	<i>Rational Unified Process</i>
SGBD	Sistema de Gerenciamento de Banco de Dados
SINDEC	Sistema Nacional de Informações de Defesa do Consumidor
SOAP	<i>Simple Object Access Protocol</i>
SPA	<i>Single Page Application</i>
WWW	<i>World Wide Web</i>
XML	<i>Extensible Markup Language</i>
XP	<i>Extreme Programming</i>

Sumário

1	INTRODUÇÃO	23
1.1	Justificativa	25
1.2	Objetivos	25
1.2.1	Objetivo geral	25
1.2.2	Objetivos específicos	26
1.3	Estrutura do documento	26
2	REFERENCIAL TEÓRICO	27
2.1	Fusão de informações	27
2.2	Trabalhos correlatos	31
3	MATERIAIS E MÉTODOS	35
3.1	A metodologia <i>CRISP-DM</i>	35
3.2	Plataformas de registro de reclamações e resolução de conflitos	37
3.2.1	Consumidor.gov.br	37
3.2.2	ReclameAQUI.com.br	38
3.2.3	Dados disponíveis nas plataformas	39
3.3	Arquitetura cliente-servidor	40
3.3.1	O Protocolo <i>HTTP</i>	40
3.3.2	O servidor	41
3.3.3	O cliente	42
4	ARQUITETURA	43
4.1	Modelagem de dados	44
4.2	Componentes de servidor	46
4.2.1	Servidor <i>HTTP</i>	46
4.2.2	Aquisição de dados	47
4.2.2.1	<i>Web Crawler</i> Consumidor.gov.br	49
4.2.2.2	<i>Web Crawler</i> ReclameAQUI	51
4.2.3	Fusão e análise de dados	51
4.2.4	Exportação de dados	53
4.3	Componente de cliente	53
5	RESULTADOS	55
6	CONCLUSÕES	61

REFERÊNCIAS	63
-------------------	----

1 Introdução

A *Web 2.0* proporcionou uma grande mudança na forma de utilização da internet: o conteúdo passou a ser gerado pelo usuário (DONELAN; KEAR; RAMAGE, 2012). Como exemplos disso, temos a popularização de projetos colaborativos, blogs, comunidades, mundos virtuais (jogos e redes sociais) *etc* (KAPLAN; HAENLEIN, 2010), que ocorre concomitantemente ao crescente interesse neste conteúdo por parte do mercado (HU; LIU, 2004; ZHANG; SKIENA, 2010), administração pública (TUMASJAN *et al.*, 2010; VELASQUEZ; GONZALEZ, 2010) e até análise de comportamento social (JOIA; SOARES, 2018; RODRIGUES; JÚNIOR; LOBATO, 2019; LOBATO *et al.*, 2018).

Essa permeabilidade das mídias digitais na sociedade pode ser verificada nos relatórios publicados anualmente pela parceria entre *Hootsuite* e *We are social*. Em janeiro de 2020, foram identificados mais de 4,5 bilhões de usuários da internet, um crescimento de 7% em relação ao ano anterior. Destes, aproximadamente 3,8 bilhões são usuários ativos de redes sociais. Este número reflete a grande popularidade de mídias digitais em países como Emirados Árabes Unidos, onde 99% da população é de usuários ativos de redes sociais, e mesmo na Malásia, Coreia do Sul e Taiwan, esta porcentagem fica acima dos 80%. Alguns países colaboraram fortemente para este crescimento. No Irã, o número cresceu 39%, sendo 9,4 milhões de novos usuários ativos de redes sociais. Na Índia, um acréscimo de 48% significou 130.000.000 novos usuários ativos em redes sociais. No Brasil, o número subiu 8,2%, indicando 11 milhões a mais de usuários ativos em redes sociais (WE ARE SOCIAL, 2020).

Essa crescente quantidade de usuários ativos na internet e em redes sociais ocasiona um grande volume de dados sendo gerados a cada segundo. O relatório anual *Data Never Sleeps*, da *DOMO*, referente ao ano de 2019, aponta, dentre outros dados relevantes, o número de publicações criadas por usuários a cada minuto em redes sociais: no *Twitter* são 511.200, no *Instagram*, 55.140 fotos e 277.777 *stories*. Em plataformas destinadas à transmissão de mídia, a cada minuto são mais de 694.444 horas de *streaming* de vídeo (*Netflix*), mais de 4,5 milhões de vídeos assistidos (*YouTube*) e mais 1 milhão de vídeos assistidos na *Twitch*. Segundo este mesmo levantamento, em 2020 haverá 40 vezes mais *bytes* de dados publicados na internet do que estrelas no espaço observável. (DOMO, 2020).

Um estudo realizado na China levou em conta 70 milhões de publicações de 200 mil usuários na rede social *Weibo*, apontando que a raiva é mais influente que outras emoções (como a alegria), e que ela pode se expandir mais rápida e amplamente nas redes sociais (FAN *et al.*, 2013). GRÉGOIRE; SALLE; TRIPP e TRIPP; GRÉGOIRE apontam que isso pode ser um problema para empresas, uma vez que os consumidores estão buscando

os recursos de mídias sociais para expor os seus problemas com fornecedores de produtos e/ou serviços. Quando uma falha de serviço é seguida por uma falha de recuperação (o consumidor não teve o seu problema solucionado como esperado), o consumidor pode buscar auxílio com uma terceira parte, ou pode buscar apenas a publicação do fato e expor o seu rancor (fato mais danoso à imagem da empresa). O primeiro caso é menos grave, pois é notável que o consumidor ainda busca por uma solução. Assim, o fornecedor deve aproveitar a disposição do consumidor em busca de conciliação para reparar os danos causados e negociar em busca da melhor solução, b um atendimento personalizado ao consumidor. Os autores também afirmam que uma boa estratégia para a prevenção das reclamações é ter um sistema de triagem, para atender às demandas mais graves com maior prioridade e ser honesto com os processos ao lidar com o consumidor (GRÉGOIRE; SALLE; TRIPP, 2015; TRIPP; GRÉGOIRE, 2011).

No Brasil, os poderes públicos têm desenvolvido métodos alternativos de solução de conflitos, como as câmaras de conciliação, mediação e arbitragem, criadas pelo legislativo, de acordo com a Lei Número 9.307 de 23 de setembro de 1996 (BRASIL, 1996); e os Centros Judiciários de Solução de Conflitos e Cidadania (CEJUSCs), criados pelo judiciário, de acordo com a Resolução Número 125 de 29 de novembro de 2010 (CONSELHO NACIONAL DE JUSTIÇA, 2010). Além disto, em junho de 2014, a Secretaria Nacional do Consumidor (órgão do Ministério da Justiça) disponibilizou uma plataforma online para a resolução de conflitos de maneira direta entre consumidor e fornecedor, trata-se do Consumidor.gov.br¹.

Com parcerias firmadas com PROCONs, defensorias públicas, tribunais de justiça e ministérios públicos, e com o reconhecimento do Conselho Nacional de Justiça como ferramenta eficiente de solução extrajudicial, a plataforma Consumidor.gov.br, segundo o Ministério da Justiça e Segurança Pública, “possibilita a resolução de conflitos de consumo de forma rápida e desburocratizada, sem a necessidade de recorrer ao Judiciário” (MINISTÉRIO DA JUSTIÇA E SEGURANÇA PÚBLICA, 2019a). Segundo boletim divulgado pela Secretaria Nacional do Consumidor, dados referentes ao Consumidor.gov.br e ao SINDEC (Sistema Nacional de Informações de Defesa do Consumidor, utilizado por Procons nos estados), coletados no ano de 2018, apontam que o índice de solução médio na plataforma Consumidor.gov.br chega a ser maior que o dos PROCONs integrados ao SINDEC: 81% das 609.644 reclamações cadastradas e 76,5% dos 2.274.191 atendimentos realizados, respectivamente (MINISTÉRIO DA JUSTIÇA E SEGURANÇA PÚBLICA, 2019b).

Além do Consumidor.gov.br, os consumidores brasileiros também podem contar com outras plataformas de registro de reclamações e resolução de conflitos, dentre as quais está a ReclameAQUI². Estas plataformas oferecem excelentes oportunidades para

¹ <<https://www.consumidor.gov.br/>>

² <<https://www.reclameaqui.com.br/>>

os fornecedores buscarem soluções para reclamações de consumidores. Além disso, essas plataformas contêm dados que podem ser relevantes para a descoberta de conhecimento, como saber os tipos de reclamação mais frequentes, o nível de satisfação do cliente com o atendimento recebido, a distribuição geográfica dos problemas relatados *etc.*

1.1 Justificativa

Atualmente, é possível o desenvolvimento de soluções próprias de *web crawling* com a utilização de ferramentas automatizadas como *jsdom*; *cheerio* e *Scrapy*; ou de navegadores *headless*, como *PhantomJS*; *Nightmare* e *Puppeteer*. Os navegadores *headless* são navegadores mais leves e possibilitam, por suas *APIs* (*Application Programming Interface* – Interface de Programação de Aplicação), o acesso a elementos das páginas, possibilitando inspeção do *DOM* (*Document Object Model* – Modelo de Objeto de Documento) interação com o código fonte das páginas *web* por meio de funcionalidades como injeção de código. Sua principal vantagem é a interpretação de códigos *JavaScript*, o que permite o carregamento dinâmico de conteúdo indisponível em aplicações como *SPAs* (*Single-Page Applications*).

Nota-se, porém, a ausência de uma ferramenta computacional que ofereça suporte às organizações no manejo e integração dos dados disponibilizados por diferentes plataformas de registro de reclamações e resolução de conflitos.

Não é possível, ainda, analisar esses dados de forma automatizada, a fim de aplicar modelos estatísticos no processo de descoberta de conhecimento que agrega valor às organizações cadastradas nestas plataformas. Desta forma, este trabalho propõe o desenvolvimento de um sistema computacional para análise dos dados contidos nas duas plataformas supracitadas, a fim de obter conhecimento a ser utilizado no suporte à gerência destas organizações.

1.2 Objetivos

Nesta seção serão apresentados os objetivos gerais e específicos deste trabalho.

1.2.1 Objetivo geral

Este trabalho tem como objetivo geral o desenvolvimento de sistemas computacionais para extração e fusão de dados das plataformas Consumidor.gov.br e ReclameAQUI, a fim de oferecer suporte à tomada de decisão em organizações fornecedoras de serviços e produtos, cadastradas nestas plataformas.

1.2.2 Objetivos específicos

Com a finalidade de alcançar o objetivo pretendido, buscar-se-á os seguintes objetivos específicos:

1. Desenvolver uma ferramenta para captura e fusão de dados;
2. Desenvolver uma ferramenta viabilize a análise e visualização de dados.

1.3 Estrutura do documento

No Capítulo 2 é apresentada uma revisão bibliográfica de fusão de informações em mídias digitais. No Capítulo 3 são descritas as plataformas, a metodologia aplicada e as ferramentas utilizadas no desenvolvimento deste trabalho. O Capítulo 4 descreve a arquitetura do sistema, bem como os seus componentes, e o Capítulo 5 apresenta os resultados obtidos. Por fim, no Capítulo 6 são apresentadas as conclusões obtidas.

2 Referencial teórico

Neste capítulo, serão apresentadas as bases teóricas sobre as quais o desenvolvimento deste trabalho foi construído. Serão apresentados o processo de fusão de dados utilizado e alguns conceitos fundamentais de análise de mídias digitais.

2.1 Fusão de informações

Trabalhar com múltiplas fontes de dados pode ser uma tarefa complexa, mas de grande utilidade, pois permite maior confiança, redução de ambiguidade, maior detecção, maior robustez, melhoria de cobertura espacial e temporal e diminuição de custos ([Sarma; Raju, 1991](#)). Situações como detecção de dados duplicados, corrompidos, ou mesmo incompletos, são exemplos de problemas que podem ser solucionados através de diferentes leituras ou percepções dos atributos de um objeto.

Aplicando técnicas de fusão de dados, [ALI; JOORABCHI; MESBAH](#) realizam um estudo comparativo entre aplicativos presentes em lojas de aplicativos de plataformas distintas (*Play Store App Store*), identificando mais de 80 mil pares de aplicativos com presença nas duas plataformas em um universo de 2,4 milhões de aplicativos coletados a fim de conhecer as diferenças de percepção dos aplicativos entre as plataformas. Para tanto, os aplicativos foram combinados através da identificação do seu nome e do nome de seu desenvolvedor. Critérios diferentes foram utilizados na identificação dos aplicativos correspondentes nas plataformas: busca por nome de aplicativo e de desenvolvedor exatamente iguais; e busca pelos nomes aproximadamente iguais. Para realizar as análises, foram utilizados somente os pares identificados pelo primeiro critério.

Os autores realizaram análises nas revisões de usuários com a utilização de classificadores. O primeiro classificador classifica o conteúdo da reclamação de acordo com as seguintes classes: (i) *descoberta de problema*; (ii) *solicitação de funcionalidade*; e (iii) *não informativo*. O segundo classificador realiza uma análise de sentimentos para classificar o conteúdo das reclamações como: (i) *positivo*; (ii) *negativo*; ou (iii) *neutro*. Com uma amostra de 2.100 revisões anotadas (1.050 em cada plataforma), os classificadores foram construídos com de duas maneiras: (i) com o algoritmo *Support Vector Machine* e (ii) com o algoritmo *Naive Bayes*. Com o auxílio da ferramenta *Scikit Learn Tool*, foram utilizados 1.575 aplicativos para treinamento e 525 para teste, e foram classificadas cerca de 1,7 milhões de revisões.

Os autores ainda realizam uma análise para identificar porque há discrepância na nota dos mesmos aplicativos nas plataformas diferentes. Para isto, são analisadas as

avaliações de usuários em aplicativos com notas mais discrepantes. Para tanto, foram consideradas reclamações do tipo *descoberta de problemas* para classificá-las em subcategorias: (i) *crítico*; (ii) *pós-atualização*; (iii) *reclamação de preço*; (iv) *funcionalidades de aplicativo*; e (v) *outros*.

CAMPBELL et al. realizam fusão de dados para combinação de entidades entre plataformas de mídias sociais. Os autores identificam esse problema como *cross-domain entity resolution*, pois destacam características peculiares do cenário de mídias sociais, apontando o fato de que a informação sobre as entidades nestas plataformas estão atribuídas a perfis (nome de usuário, nome completo, imagem de perfil, links), conteúdo e estrutura de grafos. Logo, os autores baseiam as suas análises em (i) informações de perfis (nome de usuário e nome completo); (ii) conteúdo de publicações (quantidade de palavras e *hashtags*); e (iii) grafos construídos com menções a usuários e *hashtags*. Foram utilizados também alguns algoritmos para comparação de *strings* a fim de identificar diferentes ordenações de caracteres de um mesmo nome próprio, método comum em registro de nomes de usuários indisponíveis (CAMPBELL et al., 2016).

Em 1986, o JDL (*Joint Directors of Laboratories*) criou o *Data Fusion Working Group* para empregar esforços na modelagem de processos de fusão de dados de múltiplas fontes. Como resultado, nasce o modelo JDL, o mais popular e difundido modelo de fusão de dados. Este modelo divide os subprocessos da fusão de dados em cinco níveis de fusão de dados, localizados entre as extremidades do processo: as múltiplas fontes de dados e os operadores do sistema.

O modelo JDL está ilustrado na Figura 1 e pode ser descrito da seguinte maneira:

- *Fontes* — São sensores e/ou outros dados de múltiplas fontes utilizados na fusão de

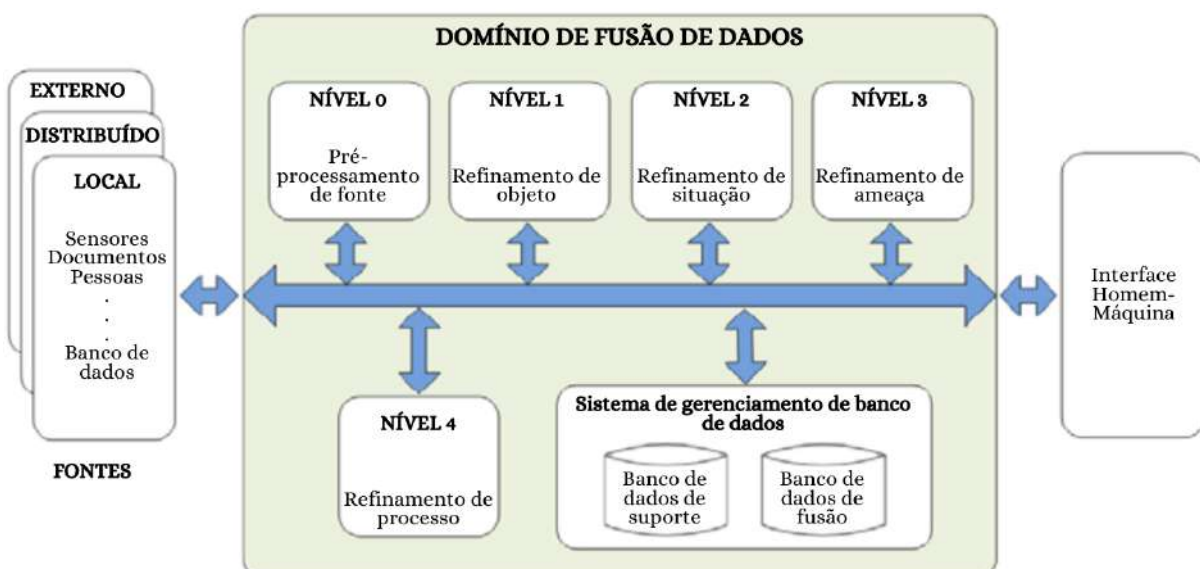


Figura 1 – O Modelo JDL (LYTRIVIS; THOMAIDIS; AMDITIS, 2009, tradução nossa)

dados;

- *Nível 0 (Pré-processamento)* — Neste nível, os dados podem ser analisados de forma a reduzir o custo de processamento, encaminhando cada dado ao seu respectivo nível de refinamento; ou mesmo descartando dados inutilizáveis no processo.
- *Nível 1 (Refinamento de objeto)* — Responsável pela estimativa e predição de estados de entidade e identificação de objetos. Por exemplo: predição de posição e velocidade; identificação de entidades (carros, construções, prédios *etc.*).
- *Nível 2 (Refinamento de situação)* — Objetiva construir uma visão mais ampla da situação da entidade. Ou seja: relacionar o objeto identificado (ou refinado) com um evento observado, ou mesmo com outros objetos identificados. Para Hall e Llinas, este nível busca interpretar os dados de maneira análoga a como um humano interpretaria o significado do dado de um sensor (HALL; LLINAS, 1997), em outras palavras: realiza uma análise contextual (WHITE, 1991). Por exemplo: processamento de informação; verificação de relacionamento entre entidades: agregação; atuação; intenção; sugestão *etc.* (LLINAS et al., 2004)
- *Nível 3 (Refinamento de ameaça)* — Projeta inferências acerca de situações futuras estimando e predizendo efeitos das situações (como o produto do nível 2) nas ações dos participantes envolvidos (LAMBERT, 2003). É responsável pela análise de vantagens e desvantagens na tomada de decisões. (ESTEBAN et al., 2005). Por exemplo: detectar ameaças inimigas; vulnerabilidades em terceiros; e oportunidades de operação (HALL; LLINAS, 1997).
- *Nível 4 (Refinamento de processo)* — Este nível é comumente considerado como parcialmente dentro e parcialmente fora do processo de fusão de dados (HALL; LLINAS, 1997), pois é responsável por atuar sobre todos os outros níveis. Assim, é um nível de gerenciamento de recursos (LAMBERT, 2003). Por exemplo: Monitorar performance e identificar potenciais fontes de melhoria de informação (ESTEBAN et al., 2005).
- *Interface Homem-Máquina* — Este nível é responsável pela interação em duas vias com o sistema automatizado com humanos. Permite, não somente o relato das informações obtidas através dos dados, alertas e *displays*; mas também a entrada de comandos e seleção de configurações por parte dos operadores do sistema.

O modelo de processo inclui ainda um Sistema de Gerenciamento de Banco de Dados (SGBD) para dar suporte ao processo e para armazenar os dados das fontes e resultantes de cada nível.

O modelo *JDL* situa-se nas primeiras definições de fusão de dados, que apontavam para o uso de sensores e aplicações militares:

“As técnicas de fusão de dados combinam dados de vários sensores e informações relacionadas de bancos de dados associados, para obter maior precisão e inferências mais específicas do que as obtidas com o uso de um único sensor” (HALL; LLINAS, 1997, p. 1, tradução nossa)

Posteriormente, houve um crescente interesse em fusão de dados gerados por seres humanos (BALAZS; VELÁSQUEZ, 2016) e definições mais modernas passaram a incluir fusão de informações:

“Fusão de informações é o estudo de métodos eficientes para transformação automática ou semi-automática de informações de diferentes fontes e diferentes momentos em uma representação que fornece suporte eficaz para tomada de decisão humana ou automatizada” (BOSTRÖM et al., 2007, p. 5, tradução nossa)

Finalmente, considerando as novas perspectivas de análise de dados, pode-se estabelecer uma correlação entre o modelo *JDL* e a fusão de informação em mineração de opiniões, apresentada na Figura 2.

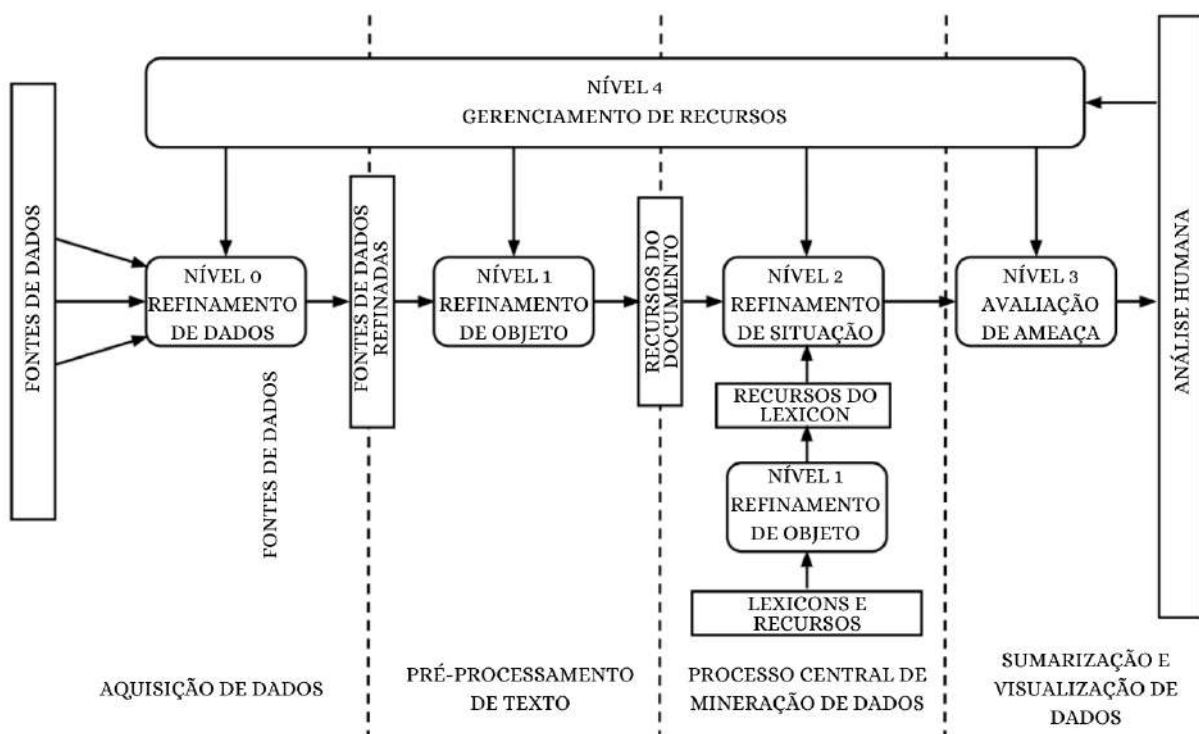


Figura 2 – *Framework* para aplicação de fusão de informação a mineração de opinião (BALAZS; VELÁSQUEZ, 2016, tradução nossa).

Assim, para fusão de informação em mineração de opinião, o nível 0 é o responsável pela filtragem e calibração de diferentes fontes de dados. O nível 1 é responsável pela correlação dos dados, colocando-os em uma única representação, como um vetor de dados. Este nível também é o responsável pelo pré-processamento dos dados. Uma vez que a coleção de documentos é pré-processada no nível 1, no nível 2 ela será analisada através

dos modelos e técnicas de mineração aplicados. No nível 3 há a entrega dos resultados para visualização e suporte à tomada de decisões. O nível 4 garante continuamente a qualidade dos níveis anteriores (BALAZS; VELÁSQUEZ, 2016).

2.2 Trabalhos correlatos

Existem trabalhos publicados na área de análise de mídias digitais que realizam extração e análise de dados de plataformas de reclamação. Alguns deles estão destacados a seguir.

TEIXEIRA et al. realizam um trabalho de análise de dados na plataforma de avaliação de reputação eBit, objetivando identificar os tópicos mais relevantes dentre as avaliações na plataforma. O trabalho baseou-se em três pilares, a saber o desenvolvimento de um *web crawler* para a aquisição dos dados, o desenvolvimento de um sistema *web* para requisição e recebimento da aquisição e análise dos dados, e a modelagem de tópicos que, para os autores, “consiste em explorar uma coleção de dados discretos de modo a encontrar uma descrição dos seus elementos”. Para o desenvolvimento do *web crawler*, foi utilizada a ferramenta *Scrapy*, na qual foram configurados os componentes *Spiders* para a leitura das páginas da plataforma e extração dos dados delas. Para o desenvolvimento do sistema *web*, foi utilizado o *framework Django*, com arquitetura *model-template-view* (MTC). Finalmente, para a modelagem de tópicos foi utilizado o algoritmo *Latent Dirichlet Allocation* (LDA), considerado pelo autores como o mais utilizado para desenvolver esta tarefa. O diagrama de funcionamento do sistema pode ser observado na Figura 3

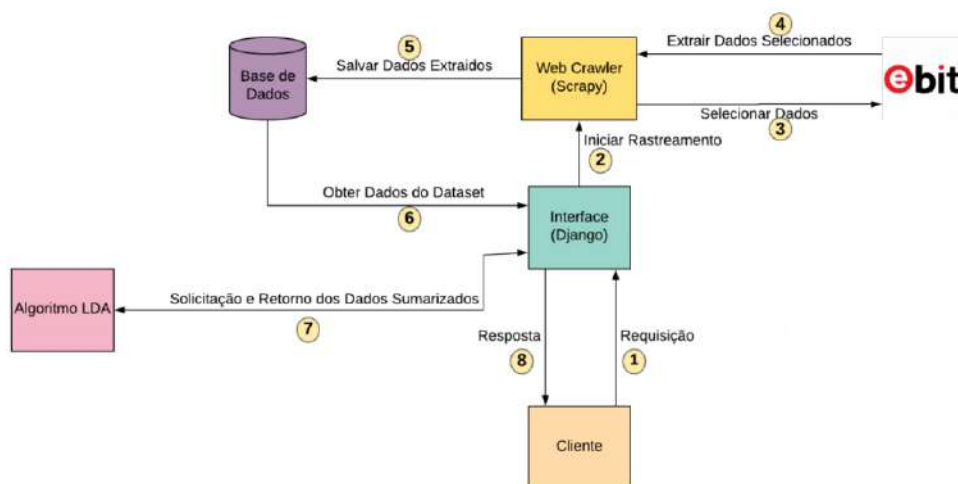


Figura 3 – Arquitetura do sistema no trabalho de TEIXEIRA et al. (TEIXEIRA et al., 2018)

O funcionamento do sistema é iniciado com uma requisição do cliente do sistema *web*. A partir da requisição, o sistema desenvolvido em *Django* inicia o rastreamento e o módulo de *web crawler* seleciona e extrai os dados úteis presentes nas páginas. Os

dados são então armazenados em um banco de dados, e são recuperados na página de exibição, para onde o cliente é redirecionado. É nesta página que o cliente pode visualizar a modelagem de tópicos, clicando em um botão para solicitar a análise. (TEIXEIRA et al., 2018).

NETTO et al. analisam dados da plataforma ReclameAQUI para suporte à tomada de decisões de empresas através de análises e visualização de dados baseadas em mapa de calor, nuvem de palavras e modelagem de tópicos. Foi utilizada a metodologia *Design Science Research*. Para alcançar a estratégia proposta, foram desenvolvidos os módulos de extração e análise de dados. Para o módulo de extração de dados, foi utilizada a ferramenta *PhantomJS*, uma vez que os autores consideram a forma como os dados são carregados na plataforma (dinamicamente), impossibilitando assim utilizar ferramentas como o *Scrapy*. Os dados coletados neste módulo são armazenados em um arquivo *comma-separated values (CSV)*, onde os dados são pré-processados e o resultado é disponibilizado para ser utilizado no módulo de análise de dados. Os resultados das análises foram disponibilizados em um *dashboard* com a visualização gráfica dos dados resultantes das análises implementado em um servidor desenvolvido sobre o *framework Flask*, da linguagem *Python*. A Figura 4 ilustra o fluxo de dados do sistema.

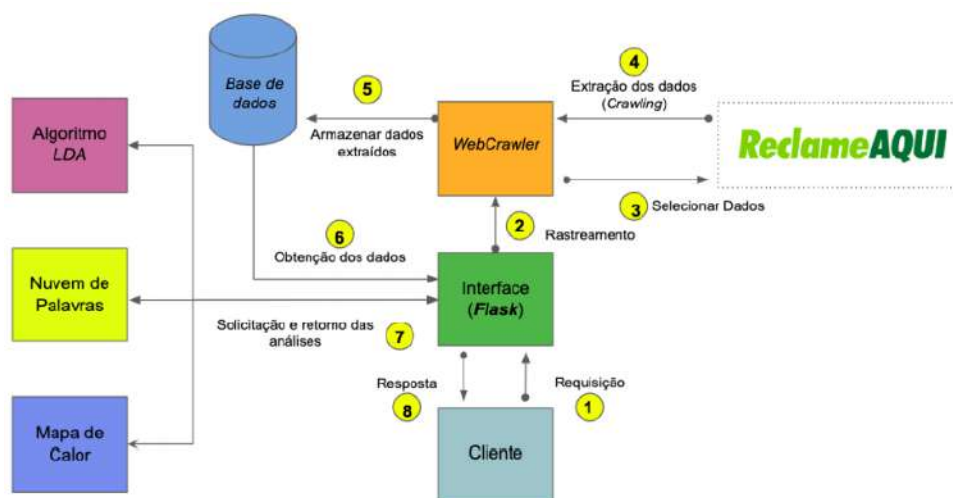


Figura 4 – Arquitetura do sistema no trabalho de NETTO et al. (NETTO et al., 2019)

Como pode ser verificado na Figura 4, o cliente da aplicação realiza uma requisição ao servidor, que inicia o processo de aquisição de dados, em seguida os dados são armazenados em um banco de dados. O usuário é então redirecionado a uma página onde pode solicitar as análises de modelagem de tópicos, nuvem de palavras ou mapa de calor, na qual o mapa do Brasil é segmentado em regiões para exibição da distribuição das reclamações.

SOUSA et al. analisam dados de reclamações da plataforma ReclameAQUI para identificar os principais tópicos presentes nas reclamações e suas relações entre si (conforme ilustra a Figura 5, com resultados obtidos para as empresas Tim, Vivo, Oi e Claro), além

de apresentar as distribuições geográficas e temporais das reclamações. Com o objetivo de identificar os principais assuntos/problemas reportados nas reclamações e os aspectos específicos destas, analisar a distribuição das reclamações considerando dimensões geo-temporais e identificar as implicações práticas delas nas empresas, os autores optaram pela utilização da metodologia *Cross-industry standard process for data mining (CRISP-DM)*, ou Processo Padrão Inter-Indústrias para Mineração de Dados. Para a aquisição dos dados foi utilizado um *web crawler* desenvolvido com o apoio da biblioteca *requests*, escrita em *Python*.



Figura 5 – Relação entre tópicos das reclamações no trabalho de SOUSA et al. (SOUSA et al., 2020).

Para pré-processar os dados, foram removidos caracteres identificados como pertencentes a saudações, *URLs*, *stopwords*, números, acentuação e caracteres especiais. Foi identificado que o algoritmo *Non-Negative Matrix Factorization* teve resultados mais coerentes e diversos, se comparado ao *LDA* e ao *Latent Semantic Analysis*, por isto este foi utilizado para a modelagem de tópicos realizada. Foi utilizada uma combinação simples para que os tópicos de cada documento fossem associados aos pares. Assim, cada par resultante foi armazenado em uma linha de um arquivo *CSV*, o qual foi utilizado como lista

de adjacências no software *Gephi* para gerar uma visualização de uma rede de tópicos, a fim de reconhecer a rede de tópicos presente nos documentos. Os autores ainda realizaram análises de distribuição regional e geo-temporal dos tópicos identificados. (SOUSA et al., 2020).

Todos estes trabalhos realizam extração e análise de dados em plataformas de gerenciamento de conflitos ou reputação de empresas. Enquanto os três implementam soluções de análise de tópicos, TEIXEIRA et al. e NETTO et al. utilizam o algoritmo *LDA* para esta finalidade, enquanto SOUSA et al. opta pelo algoritmo *Non-Negative Matrix Factorization*. Nesta solução proposta, também é implementada uma modelagem de tópicos com *LDA*, o algoritmo mais utilizado nos trabalhos publicados nos anais das conferências mais relevantes da área de análise de mídias digitais (JÚNIOR et al., 2020).

Os três trabalhos utilizam estratégias diferentes para a aquisição dos dados, a saber as ferramentas *Scrapy*, *PhantomJS* e a biblioteca *request*. Nesta solução proposta, é utilizada a ferramenta *Selenium* para lidar com a natureza dinâmica das plataformas, além da ferramenta *Beautiful Soup*, que auxilia na extração de dados da plataforma Consumidor.gov.br, fornecendo um interpretador de segmentos incompletos de documentos *HTML*.

Os trabalhos de NETTO et al. e SOUSA et al. utilizam um *dashboard* construído em uma plataforma *web* para visualização dos dados. O primeiro faz uso do *Flask*, um *microframework* para construção de rotas também utilizado nesta proposta de solução. Estes dois trabalhos também realizam análise de distribuição geográfica. O primeiro ainda disponibiliza uma nuvem de palavras. O segundo, distribuição temporal e análise de rede dos tópicos identificados. Nesta solução proposta, serão também exibidos gráficos para distribuição geográfica (através de mapa de calor), nuvem de palavras (mas por cada tópico identificado), e distribuição temporal das reclamações nas plataformas. A metodologia aqui utilizada é a *CRISP-DM*, a mesma utilizada por SOUSA et al..

Nenhum dos trabalho, porém, realiza a fusão de dados de diferentes plataformas, mas limitam-se a apenas uma plataforma (de reclamação ou reputação). Nesta solução proposta, as análises a serem realizadas utilizam uma base de dados coletada de duas plataformas de registro de reclamação e gerenciamento de conflitos: a Consumidor.gov.br e a ReclameAQUI.

3 Materiais e Métodos

Neste capítulo, serão apresentados o método de pesquisa utilizado, as fontes de dados (plataformas de registro de reclamações e resolução de conflitos) e ferramentas e tecnologias utilizadas para o desenvolvimento deste trabalho. É também proposta a arquitetura de aplicação cliente-servidor em decorrência da natureza da aplicação (ambiente *web*).

3.1 A metodologia *CRISP-DM*

Para desenvolver a solução de análise proposta, foi utilizada a metodologia *CRISP-DM*. Este modelo de processo está dividido em 6 fases genéricas (como definiram seus autores), conforme ilustrado na Figura 6. As fases deste modelo estão descritas a seguir.

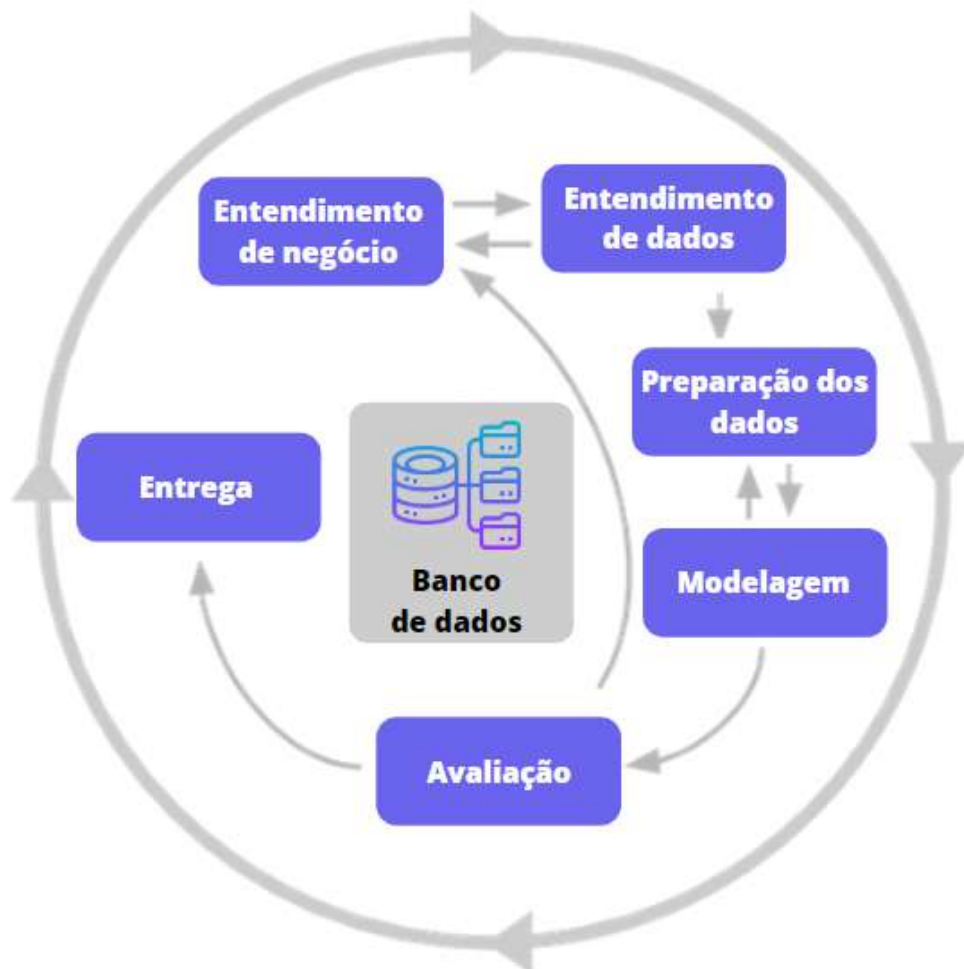


Figura 6 – Modelo de processo *CRISP-DM*.

1. *Business Understanding* — A primeira fase da metodologia *CRISP-DM* é conhecida como *Business Understanding* (Entendimento de negócio), e consiste em conhecer o cenário do problema para propor soluções que se adequem às necessidades da organização. Nesta fase, é importante compreender os objetivos e requerimentos do projeto, para traçar um plano preliminar de atuação.
2. *Data Understanding* — Entendimento de dados é a fase destinada para o conhecimento dos dados do domínio do problema. Nesta etapa, é importante identificar problemas de qualidade nos dados, para formular hipóteses para atuação.
3. *Data Preparation* — A preparação de dados é a fase na qual haverá trabalho sobre os dados brutos para gerar um *dataset* válido para trabalhar, e aplicar um modelos de aprendizagem de máquinas para descoberta de conhecimento. Algumas tarefas desta fase podem ser: limpeza, pré-processamento, transformação, mapeamento, redução de dimensão, seleção de atributos, e até criação de novos atributos, fusão/integração de dados *etc.*
4. *Modeling* — Modelagem é a fase responsável pela escolha de modelos computacionais e calibragem de seus parâmetros para atuação nos dados do problema.
5. *Evaluation* — Esta fase é a responsável por identificar a qualidade dos modelos aplicados aos dados e verificar se os resultados obtidos satisfazem as métricas e objetivos traçados desde a fase de *Business Understanding*. Dependendo dos resultados desta fase de Avaliação, o processo pode voltar à fase inicial para revisão dos parâmetros e planejamento estratégico.
6. *Deployment* — A entrega é a última fase do modelo *CRIPS-DM*. Após a escolha e configuração dos parâmetros dos modelos, os resultados das análises precisam ser compartilhados com os *stakeholders*. Nesta etapa, são construídos relatórios e visualizações gráficas dos resultados obtidos das análises.

Apesar de apresentar uma sequência natural de fases, este modelo de processo permite a retroalimentação de fases. Desta forma, não é um modelo de processo unidirecional, o que favorece à qualidade do desenvolvimento da ferramenta de *Data Mining*. Tem-se, assim, descrito um modelo de processo independente tanto da área de aplicação quanto das tecnologias utilizadas no processo ([WIRTH; HIPPE, 2000](#)).

A fase 3 (*Data Preparation*) da metodologia *CRISP-DM* pode ser associada aos níveis 0 e 1 do modelo de fusão de informação, a fase 4 (*Modeling*) ao nível 3, a fase 5 (*Evaluation*) ao controle de qualidade do nível 4. Finalmente, o nível 6 (*Deployment*) é correlato ao nível 3.

3.2 Plataformas de registro de reclamações e resolução de conflitos

Para este trabalho, foram selecionadas duas plataformas de registro de reclamações e resolução de conflitos, a saber Consumidor.gov.br e ReclameAQUI. A seguir, estão descritas cada uma destas plataformas.

3.2.1 Consumidor.gov.br

Disponibilizado em junho de 2014, o Consumidor.gov.br é uma plataforma de conciliação criada pelo poder público com grande participação popular: são mais de 1,2 milhões de consumidores e 494 empresas cadastradas. Das 1,7 milhões de reclamações registradas, 609.644 foram somente no ano de 2018, um crescimento de aproximadamente 30% em relação ao ano anterior (enquanto os Procons integrados ao SINDEC têm registrado uma diminuição contínua nos atendimentos desde 2015) (MINISTÉRIO DA JUSTIÇA E SEGURANÇA PÚBLICA, 2019a). O prazo máximo para uma empresa responder a uma reclamação nesta plataforma é de 20 dias. No entanto, o tempo médio de resposta está abaixo de 7 dias. Após a resposta da empresa, o consumidor pode avaliar o atendimento em até 10 dias. O consumidor não é prejudicado se o problema não for solucionado, pois ele ainda tem à sua disposição os órgãos tradicionais de defesa do consumidor. A Figura 7 ilustra a tela inicial do Consumidor.gov.br.



Figura 7 – Página inicial da plataforma Consumidor.gov.br

Nesta plataforma, é possível ver os relatos dos consumidores acerca dos atendimentos às suas reclamações. Estes relatos podem ser filtrados por várias opções disponíveis em um formulário. Além disto, é possível visualizar também dados referentes a cada empresa cadastrada.

3.2.2 ReclameAQUI.com.br



Figura 8 – Página inicial da plataforma ReclameAQUI

Com 42 milhões de visualizações de páginas por mês, a plataforma ReclameAQUI (Figura 8) já conta com 120.000 empresas e 15.000.000 consumidores cadastrados (ReclameAQUI, 2019). Esta plataforma também possui um sistema de classificação por desempenho da empresa na plataforma, baseado no cálculo definido pela equação

$$AR = ((IR * 2) + (MA * 10 * 3) + (IS * 3) + (IN * 2)) / 100 \quad (3.1)$$

onde AR é a Avaliação do ReclameAQUI, que leva em consideração o Índice de Resposta (IR); Média das Avaliações (MA); Índice de Solução (IS); e Índice de Novos Negócios (Voltaria a fazer negócios?) (IN). As empresas são então avaliadas de acordo com o intervalo da nota obtida no cálculo da equação 3.1. A Tabela 1 identifica as avaliações de empresas e seus intervalos de avaliação.

Avaliação	AR
Não recomendada	$AR < 5.0$
Ruim	$5.0 \leq AR \leq 5.9$
Regular	$6.0 \leq AR \leq 6.9$
Bom	$7.0 \leq AR \leq 7.9$
Ótimo	$8.0 \leq AR \leq 10$

Tabela 1 – Avaliação das empresas na plataforma ReclameAQUI

3.2.3 Dados disponíveis nas plataformas

As plataformas Consumidor.gov.br e ReclameAQUI disponibilizam dados sobre empresas cadastradas e reclamações efetuadas, além de fornecer filtros para visualização destas. A Tabela 2 apresenta os dados e funcionalidades presentes nestas plataformas.

Dados e filtros	Consumidor.gov.br	ReclameAQUI
Dados de empresas	Índice de Solução (%), Satisfação com o Atendimento (%), Reclamações Respondidas (%), Prazo Médio de Respostas (dias) e Total de Reclamações Finalizadas.	Reclamações respondidas (%), Voltaria a fazer negócio (%), Índice de solução (%), Nota do consumidor (%), Número de Reclamações, Número de Reclamações Respondidas, Número de Reclamações Não Respondidas e Número de Reclamações Avaliadas.
Dados de reclamação	Data, Cidade, Estado, Relato, Resposta, Situação (Resolvida, Não resolvida, Não avaliada), Avaliação e Nota (de 1 a 5).	Status, Cidade - Estado, ID, Data, Hora, Título, Conteúdo, Respostas, Resposta final, O problema foi resolvido?, Voltaria a fazer negócio? e Nota do atendimento (de 0 a 10).
Filtros de reclamação	Palavras-Chave, Segmento de Mercado, Fornecedor, Área, Assunto, Problemas, Região, UF, Cidade, Data Inicial, Data Final, Avaliação e Nota do Consumidor.	Categorias, Produtos e serviços, Tipos de problemas e busca textual.

Tabela 2 – Dados e filtros presentes nas plataformas Consumidor.gov.br e ReclameAQUI

Algumas propriedades das reclamações são análogas nas duas plataformas. A Tabela 3 apresenta as propriedades em comum presentes nas reclamações efetuadas nas duas plataformas.

Plataforma	Propriedade
Consumidor.gov.br / ReclameAQUI	Relato
Consumidor.gov.br / ReclameAQUI	Cidade - Estado
Consumidor.gov.br / ReclameAQUI	Data
Consumidor.gov.br / ReclameAQUI	Resposta(s)
Consumidor.gov.br / ReclameAQUI	Avaliação
Consumidor.gov.br / ReclameAQUI	O problema foi resolvido?
Consumidor.gov.br / ReclameAQUI	Nota do atendimento

Tabela 3 – Propriedades de relatos comuns às duas plataformas

3.3 Arquitetura cliente-servidor

Nesta seção serão apresentadas as tecnologias utilizadas para o desenvolvimento do sistema *web*, a saber: O protocolo *HTTP*; o servidor; e o cliente da aplicação.

3.3.1 O Protocolo *HTTP*

O protocolo *HTTP* (*Hypertext Transfer Protocol* - Protocolo de Transferência de Hipertexto) atua na camada de apresentação e tem sido utilizado para a transmissão de dados pela rede mundial de computadores desde o início da década de 1990. Inicialmente, o protocolo apresentava apenas o método **GET**, que permite ao cliente solicitar um documento a um servidor. Posteriormente, métodos como **POST**, **HEAD**, **PUT** e **DELETE** (entre outros) foram adicionados em novas especificações do protocolo, hoje na versão **HTTP/2**. Apesar de ser baseado em texto simples (*plain text*), o protocolo permite a transmissão de dados de qualquer tipo de dados que o cliente e servidor entendam, indo muito além da transferência de hipertextos, permitindo, além disso, envio e recebimento de imagens, vídeos e demais tipos de arquivos ([Mozilla Developer Network, 2019b](#)). Esse protocolo não possui estado (não é possível compartilhar dados entre duas solicitações distintas, ainda que de um mesmo cliente), portanto, quaisquer dados referentes à interação cliente-servidor devem ser gerenciados por outras ferramentas, como um banco de dados e/ou uso de *cookies*.

O protocolo estabelece a comunicação por meio de trocas de mensagens, contendo um cabeçalho e um corpo, tanto para as solicitações quanto para as repostas. No cabeçalho devem ser encontradas algumas informações necessárias para o estabelecimento da comunicação, como o endereço de destino da transmissão, a versão do protocolo utilizada e linguagem e tipos de dados suportados. Essas informações estão dispostas no cabeçalho como pares de chave e valor. As informações de cabeçalho podem agrupar-se em cabeçalhos genéricos, cabeçalhos de solicitação e cabeçalhos de resposta. O código abaixo ilustra um exemplo simplificado de uma transmissão de dados através do protocolo *HTTP* para uma solicitação de um cliente a um servidor.

```
GET / HTTP/1.1
Host: www.uema.br
Accept: text/html,application/xhtml+xml
Accept-Encoding: gzip, deflate, br
Accept-Language: pt-BR,pt
```

No exemplo acima: na linha 1, são informados o método de requisição (**GET**), o caminho do destino (/) e a versão do protocolo (**HTTP 1.1**); na linha 2, é informado o **Host** de destino da transmissão dos dados. As demais linhas informam restrições para a

transmissão da resposta a ser enviada pelo servidor.

Outros cabeçalhos característicos do sistema podem ser utilizados para definir controle de cache, controle de versionamento, autenticação, sessões, controle de acesso de terceiros, controle de rastreamento, informações do corpo da mensagem, entre outros. Cabeçalhos do *HTTP/2* permitem, inclusive, sinalizar se conexões estabelecidas devem continuar existentes após o término da transação atual ou se devem ser finalizadas ([Mozilla Developer Network, 2019a](#)). Uma lista de todos os campos de cabeçalho *HTTP* já padronizados é constantemente atualizada pela *Internet Assigned Numbers Authority* (*IANA*) ([AUTHORITY, 2019](#)).

O código abaixo ilustra um exemplo simplificado de uma transmissão de dados através do protocolo *HTTP* para uma resposta a uma solicitação de um servidor a um cliente.

```
HTTP/1.1 200 OK
Date: Mon, 25 Aug 2020 10:00:00 GMT
Server: Apache
Content-Length: 12345
Content-Type: text/html

<!DOCTYPE html... (Aqui vêm os 12345 bytes da página web requisitada)
```

No exemplo acima: Na linha 1, é informado que o código *HTTP* da resposta à solicitação é **200 (OK)**. Isto significa que a requisição foi aceita, e não há problemas nem referentes ao cliente, nem com o servidor; e que a resposta é válida de acordo com o solicitado pelo cliente. Na linha 2, são informados horário e data do envio da resposta, e, a linha 3 informa o sistema do servidor. As demais linhas dão informações acerca do tamanho e tipo do dado contido na resposta. Demais cabeçalhos podem ser utilizados para outras informações.

O cabeçalho de uma mensagem *HTTP* (tanto requisição quanto resposta) que contém um corpo deve ter o cabeçalho separado do corpo da mensagem por uma linha em branco, conforme verificado na resposta acima.

3.3.2 O servidor

O conjunto de *hardware* e *software* dedicado a gerenciar requisições de acesso na *WWW* é conhecido por servidor *web*. Uma das linguagens de desenvolvimento de aplicações mais populares que favorecem a criação de um servidor web é o *Python*. A linguagem *Python* possui diferentes *frameworks* e bibliotecas para lidar com requisições na *WWW*,

como *Django*; *Bottle*; *web2py*; entre outros. Um deles é o *Flask*¹, um *microframework* para desenvolvimento *web* através da definição de rotas baseado em anotações. O *Flask* deverá lidar com as requisições *HTTP* para receber dados (em formato *JSON*) e processá-los, enviando requisições necessárias aos demais componentes do sistema, como SGBD; *Crawlers* e componente de análise. No código abaixo, encontra-se uma configuração simples de rota².

```
from flask import Flask, escape, request

app = Flask(__name__)

@app.route('/')
def hello():
    name = request.args.get("name", "World")
    return f'Hello, {escape(name)}!'
```

Faz-se necessária a utilização de um SGBD para o armazenamento dos dados que serão coletados e das análises que serão realizadas. Para isto, será utilizado o *MongoDB*³, um SGBD *NoSQL* (*Not Only SQL*) orientado a documentos, que possui *API* para troca de dados em formato *JSON* (*JavaScript Object Notation* – Notação de Objetos *JavaScript*). A ferramenta *PyMongo*⁴ permite o gerenciamento de documentos *JSON* no *MongoDB* de forma simples através dos métodos `.find()`; `.find_one()`; `insert_one()`; `delete_one()` entre outros.

3.3.3 O cliente

Este lado da aplicação é o responsável pela interação direta com o usuário do sistema, permitindo-o escolher os parâmetros necessários, por meio da seleção de itens e preenchimento de formulários. O cliente do sistema é uma aplicação desenvolvida com o *framework* *Vue.js* para a exibição dos resultados da análise dos dados. É no cliente da aplicação que serão disponibilizados os componentes para visualização gráfica dos dados resultantes da análise e fusão de dados.

¹ O *framework* *Flask* é mantido pela *The Pallets Projects*. Mais detalhes podem ser encontrados em <https://palletsprojects.com/p/flask/>

² Extraído do *site* oficial <https://palletsprojects.com/p/flask/>

³ O SGBD *MongoDB* é mantido pela *MongoDB, Inc.* Mais detalhes podem ser encontrados em <https://www.mongodb.com/>

⁴ A ferramenta *PyMongo* é mantido pela *MongoDB, Inc.* Mais detalhes podem ser encontrados em <https://api.mongodb.com/python/current/>

4 Arquitetura

Neste capítulo, serão descritos os componentes utilizados nesta proposta de *software*, bem como seus relacionamentos. A arquitetura proposta é baseada no modelo cliente-servidor. Dessa forma, é possível isolar as responsabilidades dos componentes do sistema, que podem ser classificados em componentes de servidor e componente de cliente. A Figura 9 ilustra como os componentes do *software* se relacionam. Os passos de *a* a *h* ilustram o fluxo de dados no sistema.

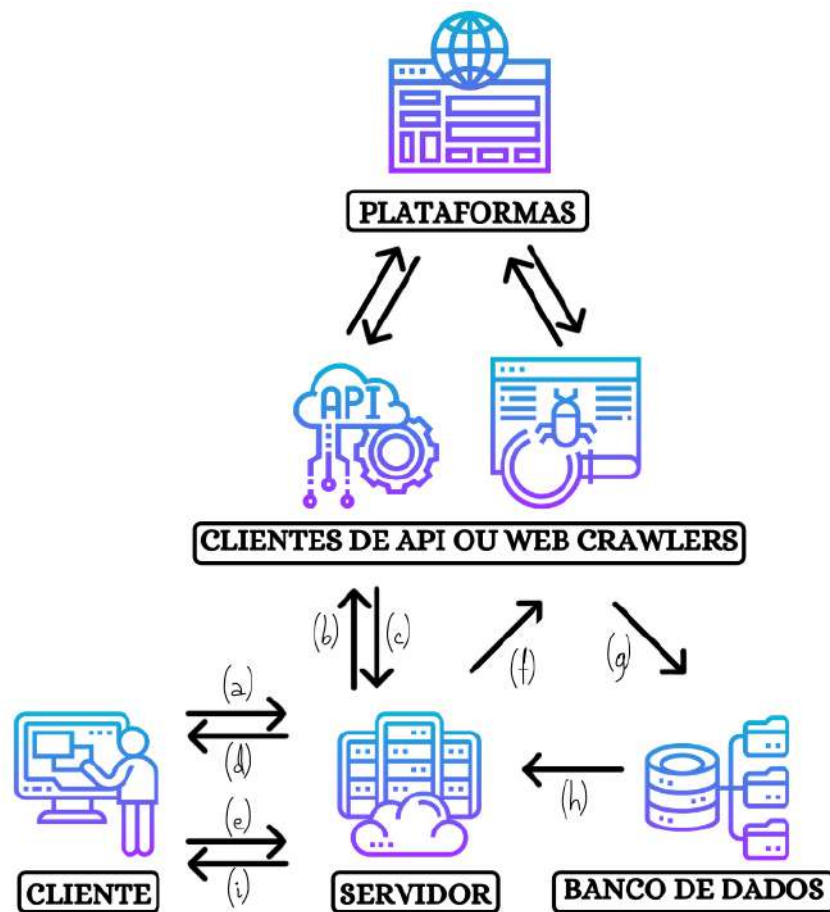


Figura 9 – Arquitetura da solução proposta.

A aplicação cliente foi desenvolvida em *JavaScript* com o *framework* *Vue.js*. Este *framework* é utilizado para a criação de uma *SPA*. Para o *dashboard*, foi usada a biblioteca *chart.js* para criação dos gráficos. O componente de servidor foi desenvolvido na linguagem *Python* com a utilização do *framework* *Flask*.

Os principais componentes da aplicação cliente são o de busca por empresas e o de visualização de análises. O primeiro componente (busca) está relacionado à seleção da empresa em cada plataforma, e sua importância está relacionada ao fato de que as empresas

podem conter nomes diferentes para cada plataforma (como “Americanas Marketplace” no ReclameAQUI e “Americanas.com” no Consumidor.gov.br). Assim, faz-se necessário conhecer o nome e os identificadores da empresa analisada em cada plataforma. Para isto, o usuário da aplicação insere o nome da empresa a buscar e a aplicação cliente envia os dados para a rota `/name` da aplicação em servidor (Fig. 9a). Nesta etapa, a ferramenta realiza requisições aos *web crawlers* específicos para cada plataforma (Fig. 9b) e recebe uma lista de empresas correspondentes ao nome inserido (Fig. 9c). Essa lista é tratada no servidor e enviada ao cliente da aplicação em formato *JSON*, que será carregado pelo componente de cliente, para utilizá-lo nos subcomponentes de visualização de dados (Fig. 9d). As chaves do objeto *JSON* enviado como resposta à requisição do cliente são os nomes de cada plataforma, e cada uma está associada a um vetor contendo objetos relacionados a cada item retornado pela respectiva plataforma. Estes objetos devem conter todos os dados necessários para que a empresa seja identificada pelos *web crawlers* e/ou clientes de *API* que irão realizar a aquisição dos dados.

Após realizar o carregamento do objeto *JSON* contendo os dados das empresas identificadas em cada plataforma, a aplicação cliente exibe cada resultado, agrupados por plataforma, para a seleção da representação da empresa em cada plataforma. Após a seleção realizada pelo usuário, a aplicação cliente submete os dados à rota `/analysis` da aplicação em servidor (Fig. 9e), que extrai os parâmetros da requisição *HTTP* para repassá-los aos componentes de aquisição de dados respectivos de cada plataforma (Fig. 9f). Os dados obtidos dos componentes de aquisição de dados são salvos em um banco de dados criado no SGBD *MongoDB* para armazenamento (Fig. 9g) e, por fim, os dados são repassados à aplicação em servidor (Fig. 9h) para realizar as análises propostas. Finalmente, os resultados das análises são enviados de volta à aplicação cliente (Fig. 9i) para que esta alimente os gráficos utilizados para visualização dos dados obtidos das análises.

A seguir estão descritos a modelagem de dados e cada um dos componentes do sistema, organizados em componentes de servidor e componente de cliente.

4.1 Modelagem de dados

Os dados do sistema são majoritariamente representados em formato *JSON*. A opção por um único modelo de representação de dados deu-se para que esta homogeneidade de estrutura elimine um problema muito comum em sistemas de informação que utilizam SGBDs relacionais, conhecido como diferença de impedância. Este problema está relacionado à diferença nas representações de dados do sistema e do SGBD. Por exemplo, um sistema pode ser escrito em linguagem *Java* e utilizar o paradigma orientado a objetos ao mesmo tempo em que armazena esses dados em registros de tabelas no modelo entidade-relacional. Alguns esforços para resolver este problema são conhecidos, como

a criação de *frameworks* de mapeamento de estruturas, conhecidos como *ORM* (sigla para *object-relational mapping* – mapeamento objeto-relacional), e até mesmo a criação de SGBDs orientados a objetos. Objetivando eliminar o processo de mapeamento, foram consideradas a natureza da aplicação proposta (*web*) e a maneira como os dados são normalmente apresentados neste ambiente, para a adoção de uma modelagem de dados em formato *JSON*, permitindo assim a utilização de uma única forma de apresentação de dados em todo o sistema.

Para o armazenamento dos dados coletados neste formato, o SGBD *NoSQL MongoDB* foi escolhido, por conta de sua maturidade, popularidade e facilidade de uso. O mesmo formato, *JSON*, é utilizado pelos componentes de cliente e servidor da aplicação para envio e recebimento de dados, e é utilizado também na comunicação entre os subcomponentes de aquisição de dados e o servidor da aplicação para envio dos dados adquiridos.

É necessário, ainda, identificar algumas propriedades dos objetos coletados de cada plataforma. Por exemplo, duas aquisições diferentes podem retornar o mesmo objeto de reclamação, logo, é preciso saber qual é a propriedade relacionada ao identificador único, para impedir a presença de registros duplicados no banco de dados do sistema. Outros exemplos são de propriedades que devem ser utilizadas em análises específicas, como a localização do usuário, a data da reclamação, a avaliação do usuário ao atendimento recebido *etc.* Para que essa identificação seja estabelecida, é necessário informar quais propriedades das diferentes plataformas estão associadas a uma propriedade em comum. Para isso, é cadastrado no banco de dados do sistema uma associação de propriedades e suas equivalências, como pode ser observado no objeto *JSON* abaixo, que representa a coleção *columns* do banco de dados.

```
"columns": [
  { "id":      { "consumidor": "_id",      "reclameaqui": "_id"      } },
  { "hash":    { "consumidor": "hash",    "reclameaqui": "hash"    } },
  { "theme":   { "consumidor": "company",  "reclameaqui": "company" } },
  { "message": { "consumidor": "relato",    "reclameaqui": "complaint" } },
  { "city":    { "consumidor": "city",     "reclameaqui": "city"    } },
  { "reply":   { "consumidor": "resposta",  "reclameaqui": "companyreply" } },
  { "date":    { "consumidor": "date",     "reclameaqui": "date"    } },
  { "opinion": { "consumidor": "avaliacao",  "reclameaqui": "finalReply" } },
  { "score":   { "consumidor": "nota",     "reclameaqui": "score"   } }
]
```

Uma vez que os campos correlatos das diferentes plataformas estão identificados, é

possível realizar as análises com as propriedades corretas de cada objeto armazenado no banco de dados. Por exemplo, o nome de uma propriedade *date* da plataforma *consumidor* pode ser identificado através da requisição ao banco de dados *fusao* da seguinte maneira: `fusao["columns"].find_one({"name": "date"}, {"platform": 1})["consumidor"]`.

4.2 Componentes de servidor

O servidor da aplicação é o responsável pela aquisição e processamento dos dados, bem como a disponibilização dos resultados das análises para o cliente. Para tanto, é necessário que o cliente da aplicação envie os dados necessários para a realização destas tarefas, como identificadores da empresa em cada plataforma e a quantidade de relatos para analisar.

Os componentes de servidor da aplicação são: (i) Servidor *HTTP*, (ii) Aquisição de dados, (iii) Fusão e análise de dados e (iv) Exportação de dados. Estes componentes estão descritos a seguir.

4.2.1 Servidor *HTTP*

O servidor implementado em linguagem *Python* com o auxílio do *framework Flask* oferece, para cada etapa do processo, rotas para requisições *HTTP POST* contendo os parâmetros necessários em formato *JSON*, cujas respostas serão documentos *JSON* que deverão ser consumidos pelo cliente da aplicação.

São utilizadas as seguintes rotas para o funcionamento da aplicação:

- `/` — A raiz do sistema, que permite requisições *HTTP GET*. Esta rota é a responsável pela transferência da aplicação *Front-end* ao cliente.
- `/name` — Rota que devolverá ao cliente da aplicação listas com os nomes de empresas relacionados à busca realizada. A requisição deve ser do tipo *POST* com parâmetros em *JSON*, e a resposta será enviada em formato *JSON*.
- `/analysis` — Destino para o qual os parâmetros configurados pelo cliente (como nome da empresa, quantidade de relatos a carregar e restrições) devem ser enviados. Esta rota recebe requisições *HTTP POST* com os dados necessários para a inicialização do processo de aquisição e análise dos dados em formato *JSON*. Assim que é identificada uma solicitação válida, o componente de aquisição de dados de cada plataforma é inicializado. Ao final da análise dos dados, é gerado um documento *JSON* contendo todos os dados necessários para a visualização, que deverá ser enviado ao componente de cliente da aplicação.
- `/export` — Rota responsável por exportar os dados em formato *CSV*.

No trecho de código a seguir, encontra-se um exemplo de configuração de rota para o carregamento da lista de empresas de cada plataforma cujo nome corresponda ao parâmetro `name`.

```
@app.route(
    "/name",
    methods=["POST"]
)
@cross_origin()
def name():
    try:
        if not request.is_json:
            raise Exception()
    except:
        return jsonify(
            {
                "success": False,
                "status": "Bad Request"
            }
        ), 400
    try:
        parameters = request.get_json()
        name = parameters["name"]
        platforms = parameters["platforms"]

        results = {"success": True}
        for platform in config["platforms"]:
            if platform in platforms:
                results[platform] = platform_modules[platform].name(name)
        return jsonify(results)
    except Exception as e:
        print(e)
        return (
            jsonify(
                {
                    "success": False,
                    "status": "Internal Server Error"
                }
            ), 500
        )
)
```

4.2.2 Aquisição de dados

Esses componentes são os responsáveis pela aquisição dos dados em cada plataforma. Estão associados à etapa que “consiste em obter os documentos utilizados para minerar as opiniões” (BALAZS; VELÁSQUEZ, 2016, p. 5, tradução nossa). Eles se comunicam com o servidor da aplicação através de dois métodos que devem ser fornecidos: `name()` e `crawl()`.

O primeiro método é utilizado para receber uma sequência de caracteres contendo o nome da empresa a localizar na plataforma, e deve retornar um objeto do tipo *JSON* para a aplicação em servidor. Nesta resposta, além do nome da empresa, devem estar presentes alguns pares de chave-valor necessários para o método `crawl()` funcionar, como identificadores únicos da empresa na plataforma e outros parâmetros que este método necessite para realizar a aquisição de dados (por exemplo, na plataforma ReclameAQUI é necessário saber o *shortname* da empresa para acessar o endereço correto).

O segundo método, `crawl()`, é utilizado para realizar a aquisição, a fusão e a análise de dados. Deve, portanto: (i) receber um dicionário de parâmetros; (ii) realizar a aquisição de dados; (iii) retornar à aplicação em servidor os dados adquiridos em formato *JSON* para que esta faça o armazenamento destes dados no banco de dados do sistema.

Existem várias formas de realizar aquisição de dados em mídias digitais, pois esta etapa do processo de análise de dados depende de fatores como a acessibilidade dos dados e formas de carregamento.

Quanto à acessibilidade dos dados, ela é comumente permitida através de *APIs* públicas, quando a plataforma deseja publicar estes dados. Há vários modelos de arquiteturas de *APIs*, como *SOAP* (*Simple Object Access Protocol* – Protocolo Simples de Acesso a Objeto), *REST* (*Representational State Transfer* – Transferência de Estado Representacional), *CORBA* (*Common Object Request Broker Architecture* – Arquitetura intermediária de requisição de objeto comum, em tradução livre) *etc.* Estas *APIs* podem utilizar *JSON*, *XML* (*Extensible Markup Language* – Linguagem de Marcação Extensível), ou outra forma de apresentação desses dados.

No entanto, as plataformas utilizadas neste trabalho como fonte de dados não disponibilizam nenhuma *API* pública para acesso externo aos seus dados. Logo, foi necessário recorrer a soluções alternativas, como *Web Crawlers*.

Quanto ao carregamento desses dados, ele pode estar disposto de forma estática (quando o servidor da aplicação gera, através de *templates*, a página solicitada pelo usuário) ou dinâmica (quando o carregamento de dados é realizado a partir de interações do usuário com uma página, como em casos de plataformas que utilizam aplicações em *Front-end*, como *SPAs*).

Apesar de todas as plataformas escolhidas utilizarem carregamento dinâmico dos dados, elas não o fazem da mesma maneira: enquanto o servidor do Consumidor.gov.br entrega ao seu cliente os dados de reclamações em um código contendo partes de documento *HTML* (*Hypertext Markup Language* - Linguagem de Marcação de Hipertexto), o servidor do ReclameAQUI entrega ao seu cliente os dados em formato *JSON*. Estas especificidades para aquisição de dados devem ser implementadas para cada plataforma. Por isso, para cada plataforma desejada, um subcomponente de aquisição de dados deve ser implementado.

A solução proposta para a aquisição de dados nas plataformas escolhidas é a utilização de *scripts* automatizados para a realização de busca e filtragem de relatos de consumidores. Para cada uma das plataformas, é utilizado um algoritmo diferente de aquisição de dados, com ferramentas específicas para cada plataforma. Por exemplo, para a plataforma Consumidor.gov.br, é utilizada a biblioteca *Beautiful Soup* para interpretar a resposta do servidor da plataforma e extrair os dados desejados com a utilização de seletores da linguagem *Cascading Style Sheets (CSS)*. Esta abordagem foi escolhida porque a biblioteca *Beautiful Soup* é capaz de interpretar os trechos de código *HTML* que são enviadas pela aplicação em servidor desta plataforma em resposta às requisições *HTTP*.

É importante perceber que diferentes tarefas de aquisição de dados podem conter os mesmos registros. Para evitar o armazenamento de dados duplicados, é requerida a utilização de identificadores únicos do tipo *hash* pelos subcomponentes de aquisição de dados. Assim, é possível fazer a inserção de dados utilizando o recurso *upsert*, que primeiro verifica a ausência de um documento na base de dados para poder armazená-lo. Caso o identificador já esteja presente, o *upsert* não insere um novo documento, mas atualiza os dados do documento presente na base de dados com o novo documento encontrado. A atualização é necessária em casos em que as reclamações são respondidas e/ou avaliadas. Dessa forma, mantém-se os dados atualizados a cada execução das rotinas de aquisição de dados. A plataforma ReclameAQUI contém uma propriedade de *hash* em seus dados. A plataforma Consumidor.gov.br, porém, não possui. A estratégia utilizada é adicionar uma propriedade *hash* baseada nos campos imutáveis (que permanecem os mesmos desde o cadastro da reclamação no sistema). Assim, a propriedade *hash* dos registros desta plataforma é gerada com base nas seguintes propriedades: empresa, data, local e corpo do relato.

Notou-se, porém, que a execução da atualização da base de dados por *upsert* para cada reclamação durava a mesma quantidade de tempo que a busca pelos documentos realizada pelos subcomponentes de aquisição de dados. Optou-se, portando, em montar listas de *hashes* para excluir todos os documentos com estes valores de *hashes* de uma só vez no banco de dados e posteriormente inserir todos os documentos encontrados de uma só vez, o que reduziu o tempo utilizado para menos de 3 segundos nos testes realizados com até 60 mil reclamações adquiridas.

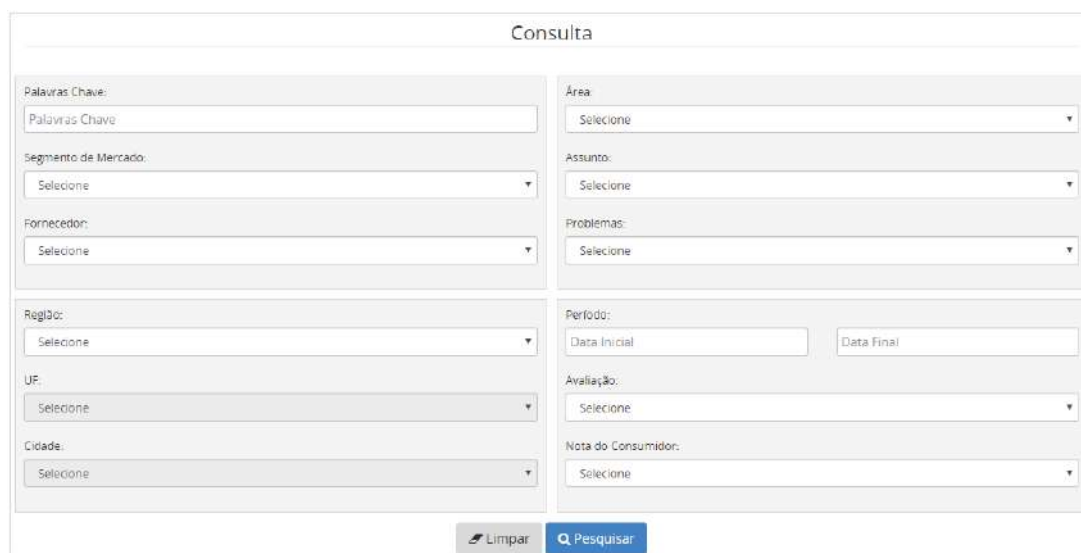
Mais detalhes de implementação dos subcomponentes de aquisição de dados são apresentados nas subseções a seguir.

4.2.2.1 Web Crawler Consumidor.gov.br

Os dados disponibilizados na plataforma Consumidor.gov.br devem ser, exclusivamente, reclamações, não sendo permitido ao usuário publicar elogios. Ao relatar um problema, o consumidor pode anexar documentos que comprovem os dados informados. A

empresa pode, então, responder ao consumidor na plataforma, além de entrar em contato através do telefone; celular e/ou *e-mail* informado. Quando os consumidores encerram um atendimento na plataforma, podem escrever uma resposta final e avaliar o atendimento com uma nota no intervalo de 1 a 5. As interações (reclamação, resposta da empresa, resposta final do usuário, e avaliação) são, então, publicadas pela própria plataforma em seu *website* no *link* Indicadores, disponível no menu de opções.

Na visualização dos indicadores, o usuário que acessa a plataforma pode filtrar os dados exibidos. Para isso, a plataforma disponibiliza uma vasta gama de filtros que podem ser aplicados, como pode ser visto na Figura 10. São estes os filtros preenchidos na plataforma utilizada pelo *web crawler* que irá adquirir os dados desta plataforma, pois a mesma não disponibiliza uma *API* para acesso público aos dados.



O formulário, intitulado "Consulta", está organizado em duas colunas principais. A coluna da esquerda contém os seguintes campos: "Palavras Chave:" com um campo de texto; "Segmento de Mercado:" com uma lista suspensa; "Fornecedor:" com uma lista suspensa; "Região:" com uma lista suspensa; "UF:" com uma lista suspensa; e "Cidade:" com uma lista suspensa. A coluna da direita contém: "Área:" com uma lista suspensa; "Assunto:" com uma lista suspensa; "Problemas:" com uma lista suspensa; "Período:" com campos para "Data Inicial" e "Data Final"; "Avaliação:" com uma lista suspensa; e "Nota do Consumidor:" com uma lista suspensa. Na base do formulário, há dois botões: "Limpar" com um ícone de lixeira e "Pesquisar" com um ícone de lupa.

Figura 10 – Opções de filtros para dados a serem exibidos

Na aquisição de dados na plataforma Consumidor.gov.br é utilizado um *script* com tarefas automatizadas para filtragem e requisição de dados que faz uso de um navegador *headless* para o acesso à plataforma. A filtragem e a coleta de dados dar-se-ão através do acesso aos campos de formulários e links para carregamento de relatos. Esta interação ocorre através de consultas por seletores *CSS* aos elementos da página, pela *API* em *JavaScript* de acesso ao *DOM* do navegador. Assim, o *script* deve acessar a página *Indicadores* para o preenchimento automatizado do formulário disponível. Em seguida, serão injetados códigos em *JavaScript* na página para efetuar a repetição das buscas automaticamente até que as demandas solicitadas sejam satisfeitas (como a quantidade de reclamações a adquirir). O resultado da requisição é, novamente com o auxílio de seletores *CSS*, capturado e estruturado em formato *JSON*.

Para o método `name()`, será utilizada a biblioteca *Beautiful Soup* para interpretar a resposta da plataforma à requisição de uma lista de empresas, apresentada em formato

HTML. Esta biblioteca permite realizar consultas aos elementos em seletores *CSS*, sendo possível assim extrair os dados desejados.

Para o método `crawl()`, será utilizada a ferramenta *Selenium*, que faz uso do *Devtools Protocol* de um navegador *Chromium*¹.

4.2.2.2 Web Crawler ReclameAQUI

A plataforma ReclameAQUI disponibiliza relatos referentes exclusivamente a reclamações, também não permitindo relatos do tipo elogios. Estes relatos estão disponíveis na página da empresa, mais especificamente na opção *Reclamações*. O ReclameAQUI disponibiliza uma busca textual das reclamações, além de opções de filtros por categoria, produtos e serviços, tipo de problema e por estado da reclamação (todas, não respondidas, respondidas e avaliadas).

Para a aquisição de dados na plataforma ReclameAQUI, é também utilizado um *web scraper*, uma vez que a plataforma não disponibiliza acesso público à sua *API*. Versões anteriores desse subcomponente realizavam uma requisição *HTTP* diretamente à rota do servidor responsável por disponibilizar os relatos, com os parâmetros necessários contidos no cabeçalho de cada requisição *HTTP*. O servidor da plataforma retorna um *JSON* contendo uma lista de relatos com seus respectivos dados e um identificador único para cada um. Contudo, sucessivos bloqueios eram realizados pelo servidor da plataforma a essas requisições. Por fim, optou-se pela utilização de *web crawlers* para gerar essas mesmas requisições a partir do cliente da aplicação, com injeção de código para realização de requisições popularmente conhecidas como *Ajax* (sigla para *Asynchronous Javascript and XML* – Javascript Assíncrono e XML). Com o acesso às respostas das requisições *HTTP* disponibilizado pela *API* do *Devtools Protocol*, é possível, então, capturar os dados enviados pelo servidor da aplicação. Através dessa estratégia, é possível adquirir dados de até 50 relatos por requisição *HTTP*.

Para os métodos `match()` e `crawl()` são utilizadas a linguagem *Python* e a ferramenta *Selenium*.

4.2.3 Fusão e análise de dados

O componente de fusão e análise dos dados é iniciado pelo servidor *HTTP*, que recebe os dados dos subcomponentes de aquisição de dados e envia-os para a análise em um dicionário *Python*, que contém uma chave para cada plataforma (“consumidor” e “reclameaqui”). Também é enviado como parâmetro o mapeamento de colunas salvo no banco de dados, para que o componente de fusão e análise identifique quais propriedades

¹ *Chromium* e *Devtools Protocol* são projetos de código aberto mantidos pela *Google*. Mais detalhes podem ser encontrados em <<https://www.chromium.org/Home>> e <<https://chromedevtools.github.io/devtools-protocol/>>

dos dados de cada plataforma possuem correlação. A partir destas informações então, são extraídas as seguintes informações: quantidade de reclamações obtidas por plataforma, distribuição cronológica das reclamações por plataformas (agrupadas por meses e anos) e distribuição geográfica das reclamações (agrupadas por Unidade Federativa). A aprendizagem de máquinas (AM) fornece métodos e ferramentas para a descoberta de conhecimento em bancos de dados. Dentre as técnicas de AM disponíveis, optou-se por implementar a modelagem de tópicos e análise de sentimentos, uma vez que são amplamente utilizados em análise de mídias sociais (CIRQUEIRA et al., 2017).

Para a descoberta de conhecimento nos textos adquiridos, é necessário, em primeiro lugar, limpá-los. Nesta etapa de pré-processamento, são realizadas tarefas como transformação dos caracteres maiúsculos em minúsculos, remoção de tags *HTML*, *links*, *stopwords*, números, pontuações, acentuações etc. Com o texto pré-processado, seguem-se as análises de texto propostas.

Para entender a percepção dos usuários acerca de um tema, é necessário saber os assuntos sobre os quais os usuários estão comentando. Esse conhecimento pode ser alcançado usando uma estratégia para descobrir os tópicos presentes em uma coleção de documentos, representando-os em termos de temas latentes descobertos (WANG; BLEI, 2011), técnica chamada de modelagem de tópicos. Nesta técnica, podem ser utilizadas abordagens como a correlação de palavras para identificar padrões de texto latentes no conteúdo (WALLACH, 2006). Neste trabalho, foi utilizado o modelo *LDA* (BLEI; NG; JORDAN, 2003).

A escolha pelo algoritmo *LDA* deu-se por consequência de um levantamento sistemático realizado previamente, no qual objetivou-se identificar as bases de dados, as ferramentas e os algoritmos mais utilizados (além de como se relacionam os tópicos de pesquisa) em trabalhos de análise de dados em mídias digitais publicados nas conferências de maior relevância na área. Percebeu-se que este algoritmo, próprio para modelagem de tópicos, estava presente entre os dez algoritmos mais utilizados nos total de 440 trabalhos analisados das conferências *International Conference on Web and Social Media*, *International Conference on Social Media & Society*, *ACM Conference on Hypertext and Social Media* e *Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis* nas edições de 2016 a 2019 (JÚNIOR et al., 2020).

Além disto, para saber como o usuário interage com os elementos do cenário em análise, é realizada uma análise de sentimento. Esta forma de análise é amplamente utilizada para tratar computacionalmente opinião, sentimento e subjetividade em texto (PANG; LEE, 2009).

Esta técnica permite analisar avaliações, atitudes e emoções das pessoas em relação a produtos, serviços, organizações, tópicos e seus atributos, entre outros (LIU, 2012). Os sentimentos identificados podem ser classificados como Positivo, Neutro e Negativo,

baseado na polaridade de um segmento textual. Para realizar esta tarefa, é utilizado o auxílio da ferramenta Polyglot². A escolha pela ferramenta *Polyglot* ocorreu devido à sua abordagem léxica. Desta forma, não é necessário treinar modelos para classificação das avaliações dos atendimentos recebidos nas plataformas.

4.2.4 Exportação de dados

Este componente é responsável por exportar os dados capturados pelo usuário para uma planilha em formato *CSV*, ou valores separados por vírgula. Esta funcionalidade é útil para permitir ao usuário realizar suas próprias análises com os dados adquiridos.

Com a ajuda do mapeamento da Tabela 3, os dados das reclamações são obtidos do banco de dados do sistema, e, então unificados em uma estrutura homogênea, possível de ser carregada em um *dataset* único, e, por fim, exportada em um arquivo *CSV* que é enviado ao cliente como resposta à requisição.

4.3 Componente de cliente

O cliente da aplicação é o responsável pela interação do usuário com o sistema. Nele, são realizadas a escolha da empresa que terá os dados analisados, a seleção de parâmetros de restrição para a aquisição dos relatos (como quantidade e filtros de relatos), e a exibição das visualizações gráficas resultantes da análise dos dados obtidos, conforme indicado no início deste capítulo. Esse componente é responsável pelo envio dos parâmetros de requisição dos dados às rotas `/name` (busca de empresas), `/analysis` (requisição de análise) e `/export` (exportação de dados).

Um exemplo da requisição *HTTP* enviada pelo componente de cliente ao componente de servidor da aplicação pode ser verificado no exemplo de código a seguir.

² <<https://pypi.org/project/polyglot>>

```
POST /analysis HTTP/1.1
Accept: application/json, */*
Content-Type: application/json
Host: www.servidor.com

{
  "consumidor": {
    "nome": "Americanas.com",
    "quantity": 50
  },
  "reclameaqui": {
    "id": 97826,
    "name": "Americanas Marketplace",
    "shortname": "americanas-marketplace",
    "quantity": 50,
    "offset": 0
  }
}
```

5 Resultados

Neste capítulo, é apresentada a ferramenta proposta, bem como as interfaces gráficas para visualização das análises de dados realizadas através da mesma. A ferramenta de *software* fornece, para a empresa pesquisada, visualização gráfica das análises realizadas nos dados obtidos das plataformas.

A Figura 11 ilustra como um usuário escolhe uma empresa para ter os dados de reclamações analisados. A tela de escolha das plataformas a buscar (11a) é utilizada para a inserção do nome da empresa e de quais plataformas realizar a busca. A submissão dos parâmetros gera uma requisição *HTTP POST* ao servidor da aplicação semelhante ao código abaixo.

```
POST /analysis HTTP/1.1
Accept: application/json, */*
Content-Type: application/json
Host: www.servidor.com

{
    "name": "Americanas",
    "platforms": ["consumidor", "reclameaqui"]
}
```

A aplicação em servidor então identifica as duas plataformas e seus respectivos subcomponentes de aquisição de dados, e executa o método `name()` de cada uma, conforme código visto na subseção 4.2.1. O código abaixo ilustra algumas opções retornadas pela aplicação em servidor para seleção de empresa na aplicação cliente.

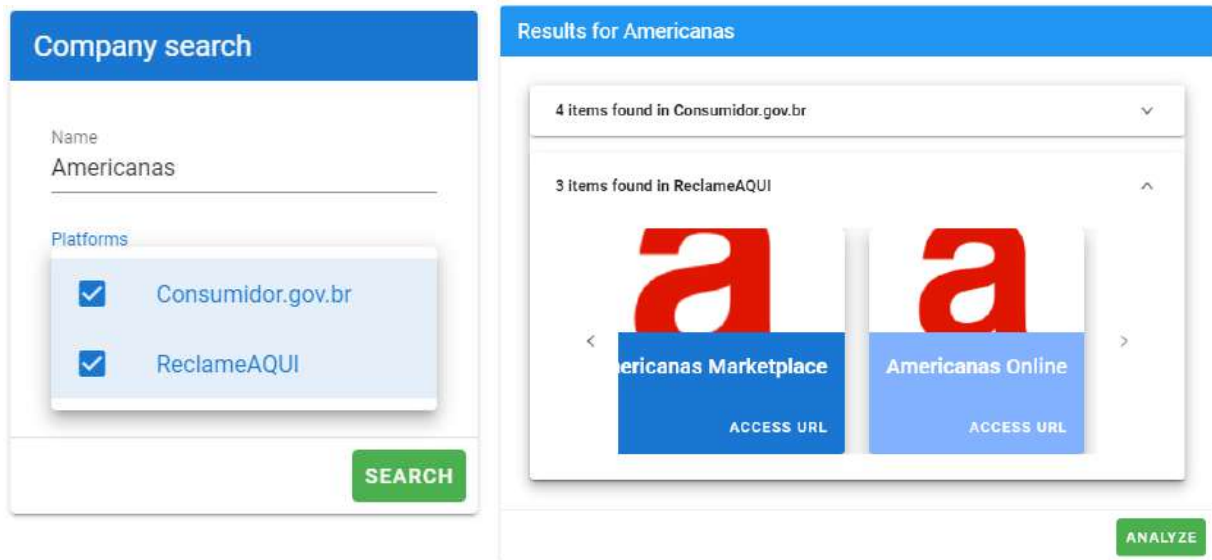
```

{
  "consumidor": [
    {
      "title": "Americanas Viagens",
      "url": "https://www.consumidor.gov.br/pages/empresa/20140206000001411/perfil"
    },
    {
      "title": "Americanas.com",
      "url": "https://www.consumidor.gov.br/pages/empresa/20140206000001407/perfil"
    },
    {
      "title": "Cartão Americanas.com",
      "url": "https://www.consumidor.gov.br/pages/empresa/201502030000053006/perfil"
    },
    {
      "title": "Lojas Americanas",
      "url": "https://www.consumidor.gov.br/pages/empresa/20140206000001402/perfil"
    }
  ],
  "reclameaqui": [
    {
      "companyName": "Americanas Marketplace",
      "count": 77904,
      "created": "2020-03-24T17:50:04",
      "fantasyName": "Americanas Marketplace",
      "finalScore": "7.5",
      "hasVerificada": false,
      "id": "97826",
      "logo": "https://raichu-uploads.s3.amazonaws.com/company_142ed4d8-665f-4124-b401-796783e932a7.png",
      "shortname": "americanas-marketplace",
      "solvedPercentual": "82.4",
      "status": "GOOD"
    },
    {
      "companyName": "americanas - Loja Online",
      "count": 19511,
      "created": "2020-04-23T08:27:00",
      "fantasyName": "americanas - Loja Online",
      "finalScore": "8.5",
      "hasVerificada": false,
      "id": "6446",
      "logo": "https://raichu-uploads.s3.amazonaws.com/logo_americanas-com-loja-online_AbTWm9.png",
      "shortname": "americanas-com-loja-online",
      "solvedPercentual": "91.5",
      "status": "RA1000"
    }
  ],
  "success": true
}

```

A resposta do servidor é utilizada pelo cliente para exibir a tela de seleção da

empresa (11b), que é utilizada para enviar os dados como nome e identificadores da empresa (além da quantidade de reclamações a buscar) para o servidor da aplicação realizar a captura e a análise dos dados. Os dados enviados são semelhantes ao código da seção 4.3.



(a) Tela de busca por nome.

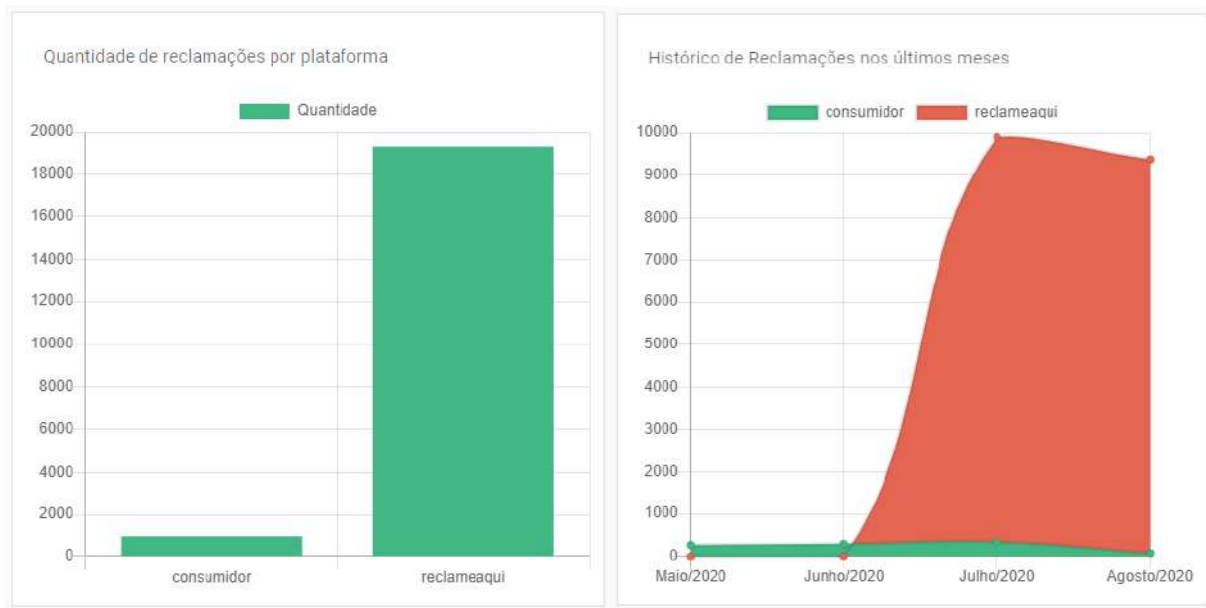
(b) Tela de escolha da empresa.

Figura 11 – Telas relacionadas à busca e seleção da empresa.

A resposta da aplicação em servidor é, finalmente, os dados para a visualização gráfica dos resultados obtidos na etapa de fusão e análise de dados. Estes dados são capturados pela aplicação cliente e destinados aos subcomponentes de visualização de dados.

As Figuras 12, 13 e 14 ilustram os gráficos do *dashboard* com os resultados das análises realizadas para a empresa Nubank com uma solicitação de 20.000 reclamações para cada plataforma. Nela, é possível visualizar os seguintes componentes gráficos: reclamações por plataformas, distribuição temporal das reclamações nas plataformas, índice de solução das reclamações e distribuição geográficas das reclamações em território nacional por mapa de calor, além do resultado da análise de tópicos realizada através do algoritmo *LDA*. Como pode ser observado, neste exemplo foram gerados 6 tópicos com diversas palavras correlacionadas. A identificação da coerência entre as palavras de um determinado tópico deve ser realizada junto a um especialista do domínio (LI et al., 2019), sendo que, neste estudo de caso, seria importante a participação de profissionais da área comercial/vendas das referidas empresas.

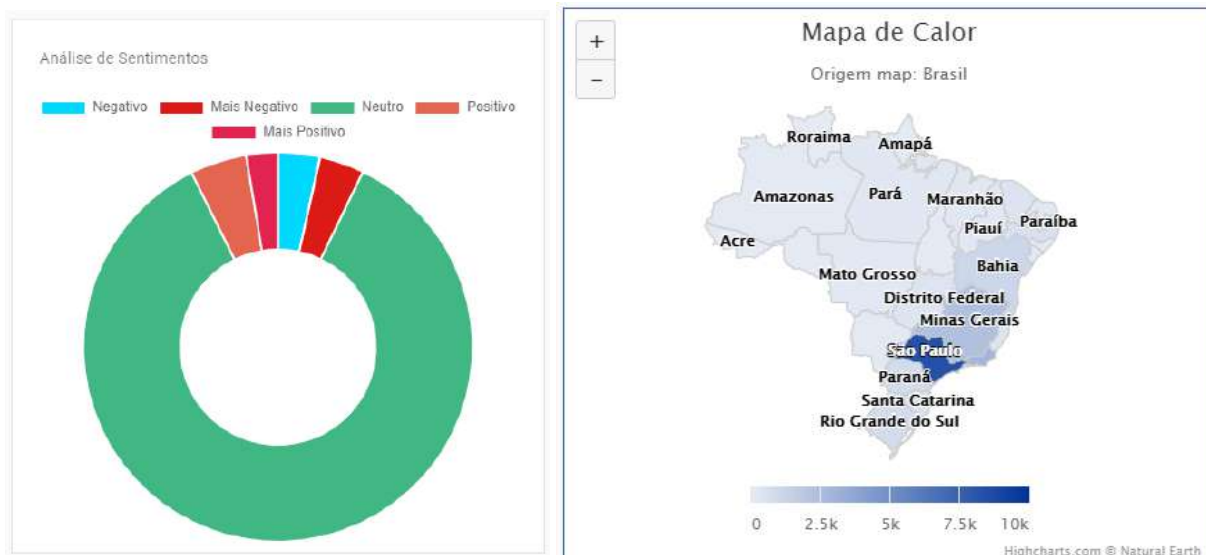
Na Figura 12 é possível perceber que a plataforma ReclameAQUI é mais utilizada pelos consumidores da Nubank para a realização de reclamações, e que o mês de julho foi o mais utilizado pelos consumidores para registro de reclamações nesta plataforma, totalizando 9.894 reclamações somente no ReclameAQUI.



(a) Distribuição das reclamações por plataforma. (b) Distribuição das reclamações por meses.

Figura 12 – Visualização gráfica de distribuição das reclamações.

Na Figura 13 é possível perceber que a estratégia utilizada identificou como sentimento *neutro* a maior parte das avaliações dos consumidores ao atendimento recebido, e que São Paulo é o estado de onde se originam mais reclamações (8.346 das 20.257 coletadas).



(a) Análise de sentimentos.

(b) Distribuição geográfica.

Figura 13 – Análise de sentimentos e distribuição geográfica.

Finalmente, na Figura 14 é possível perceber os tópicos identificados nas reclamações cadastradas nas plataformas analisadas, bem como uma nuvem das palavras mais frequentes em cada tópico.



(a) Nuvem de palavras para o tópico 1.



(b) Nuvem de palavras para o tópico 2.



(c) Nuvem de palavras para o tópico 3.



(d) Nuvem de palavras para o tópico 4.

Figura 14 – Nuvens de palavras por tópicos.

6 Conclusões

Este trabalho propõe a criação de uma ferramenta para aquisição, fusão, processamento e análise de dados de reclamações publicadas nas plataformas Consumidor.gov.br e ReclameAQUI. Nele, foram utilizadas tecnologias web para disponibilizar o sistema com a arquitetura cliente-servidor. Os dados foram extraídos das plataformas com a utilização de *scripts* para *web crawling*, e em seguida foram pré-processados e analisados. A fusão de dados ocorreu baseada nas propriedades análogas das reclamações em cada plataforma. Foram realizadas análises das distribuições por plataforma, cronológica e geográfica, além da aplicação de métodos de aprendizagem de máquinas para realização de análise de sentimentos da avaliação dos clientes ao atendimento recebido e modelagem de tópicos das reclamações, além da visualização de nuvens de palavras, aplicada a cada tópico identificado. O sistema conta com um *dashboard* para visualização das análises realizadas, além de uma opção de exportação dos dados obtidos através do mesmo. A interface intuitiva do sistema desenvolvida sobre o *framework* *Vue.js* permite a busca por empresas e a seleção da empresa correta em cada plataforma, além da seleção da quantidade de reclamações a buscar em cada plataforma.

Dentre as dificuldades encontradas no desenvolvimento deste trabalho, estão a definição da quantidade de tópicos para identificar nas reclamações e o aprimoramento da técnica aplicada para análise de sentimentos, uma vez que o método utilizado identificou um grande percentual de sentimentos neutros. Diante disto, são propostos como trabalhos futuros a realização de outras análises, como a média das notas dadas pelos clientes aos atendimentos recebidos nas plataformas. Propõe-se, também, a publicação de um artigo relatando o processo de desenvolvimento da aplicação em *front-end* utilizada neste sistema.

Este trabalho resultou em duas publicações futuras de artigos científicos. O primeiro, *Ferramentas para Análise de Mídias Sociais: Um levantamento sistemático* será publicado nos anais da conferência *Computer On The Beach*, em setembro de 2020 (JÚNIOR et al., 2020). O segundo, *Uma solução em fusão de informação para gerenciamento de reclamações feitas na web*, será publicado na revista *Soluções para o Desenvolvimento do País*, edição nº 176, de agosto de 2020.

Referências

- ALI, M.; JOORABCHI, M. E.; MESBAH, A. Same app, different app stores: A comparative study. In: IEEE PRESS. *Proceedings of the 4th International Conference on Mobile Software Engineering and Systems*. [S.l.], 2017. p. 79–90. Citado na página 27.
- AUTHORITY, I. A. N. *Message Headers*. 2019. Disponível em: <<https://www.iana.org/assignments/message-headers/message-headers.xhtml>>. Citado na página 41.
- BALAZS, J. A.; VELÁSQUEZ, J. D. Opinion mining and information fusion: a survey. *Information Fusion*, Elsevier, v. 27, p. 95–110, 2016. Citado 4 vezes nas páginas 15, 30, 31 e 47.
- BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *Journal of machine Learning research*, v. 3, n. Jan, p. 993–1022, 2003. Citado na página 52.
- BOSTRÖM, H. et al. *On the definition of information fusion as a field of research*. [S.l.]: Institutionen för kommunikation och information, 2007. Citado na página 30.
- BRASIL. *LEI Nº 9.307, DE 23 DE SETEMBRO DE 1996*. 1996. Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/l9307.htm>. Citado na página 24.
- CAMPBELL, W. M. et al. Cross-domain entity resolution in social media. *arXiv preprint arXiv:1608.01386*, 2016. Citado na página 28.
- CIRQUEIRA, D. et al. Improving relationship management in universities with sentiment analysis and topic modeling of social media channels: learnings from ufpa. In: *Proceedings of the International Conference on Web Intelligence*. [S.l.: s.n.], 2017. p. 998–1005. Citado na página 52.
- CONSELHO NACIONAL DE JUSTIÇA. *Resolução nº 125, de 29 de novembro de 2010*. 2010. Disponível em: <<http://www.cnj.jus.br/busca-atos-adm?documento=2579>>. Citado na página 24.
- DOMO. *Data never sleeps 7.0 Infographic / DOMO*. 2020. Disponível em: <<https://www.domo.com/learn/data-never-sleeps-7>>. Citado na página 23.
- DONELAN, H.; KEAR, K.; RAMAGE, M. *Online communication and collaboration: A reader*. [S.l.]: Routledge, 2012. Citado na página 23.
- ESTEBAN, J. et al. A review of data fusion models and architectures: towards engineering guidelines. *Neural Computing & Applications*, Springer, v. 14, n. 4, p. 273–281, 2005. Citado na página 29.
- FAN, R. et al. Anger is more influential than joy: Sentiment correlation in weibo. *CoRR*, abs/1309.2402, 2013. Disponível em: <<http://arxiv.org/abs/1309.2402>>. Citado na página 23.
- GRÉGOIRE, Y.; SALLE, A.; TRIPP, T. M. Managing social media crises with your customers: The good, the bad, and the ugly. *Business Horizons*, Elsevier, v. 58, n. 2, p. 173 – 182, 2015. ISSN 0007-6813. EMERGING ISSUES IN CRISIS MANAGEMENT.

Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0007681314001566>>. Citado 2 vezes nas páginas 23 e 24.

HALL, D. L.; LLINAS, J. An introduction to multisensor data fusion. *Proceedings of the IEEE*, IEEE, v. 85, n. 1, p. 6–23, 1997. Citado 2 vezes nas páginas 29 e 30.

HU, M.; LIU, B. Mining and summarizing customer reviews. In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2004. p. 168–177. Citado na página 23.

JOIA, L. A.; SOARES, C. D. Social media and the trajectory of the “20cents movement” in brazil: An actor-network theory-based investigation. *Telematics and Informatics*, v. 35, n. 8, p. 2201 – 2218, 2018. ISSN 0736-5853. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0736585318303277>>. Citado na página 23.

JÚNIOR, E. G. S. L. et al. Ferramentas para análise de mídias sociais: Um levantamento sistemático. *Anais do Computer on the Beach*, p. 389–396, 2020. Citado 3 vezes nas páginas 34, 52 e 61.

KAPLAN, A. M.; HAENLEIN, M. Users of the world, unite! the challenges and opportunities of social media. *Business Horizons*, v. 53, n. 1, p. 59 – 68, 2010. ISSN 0007-6813. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0007681309001232>>. Citado na página 23.

LAMBERT, D. A. An exegesis of data fusion. *STUDIES IN FUZZINESS AND SOFT COMPUTING*, PHYSICA-VERLAG, v. 127, p. 68–75, 2003. Citado na página 29.

LI, D. et al. Integration of knowledge graph embedding into topic modeling with hierarchical dirichlet process. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. [S.l.: s.n.], 2019. p. 940–950. Citado na página 57.

LIU, B. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, Morgan & Claypool Publishers, v. 5, n. 1, p. 1–167, 2012. Citado na página 52.

LLINAS, J. et al. *Revisiting the JDL data fusion model II*. [S.l.], 2004. Citado na página 29.

LOBATO, F. M. F. et al. Vamos falar sobre deficiência? uma análise dos tweets sobre este tema no brasil. In: SBC. *Anais do VII Brazilian Workshop on Social Network Analysis and Mining*. [S.l.], 2018. Citado na página 23.

LYTRIVIS, P.; THOMAIDIS, G.; AMDITIS, A. Sensor data fusion in automotive applications. In: _____. [S.l.: s.n.], 2009. ISBN 978-3-902613-52-3. Citado 2 vezes nas páginas 15 e 28.

MINISTÉRIO DA JUSTIÇA E SEGURANÇA PÚBLICA. *Problemas com alguma empresa? Consumidor.gov.br resolve!* 2019. Disponível em: <<https://www.justica.gov.br/news/collective-nitf-content-1552426609.41>>. Citado 2 vezes nas páginas 24 e 37.

- MINISTÉRIO DA JUSTIÇA E SEGURANÇA PÚBLICA. *Senacon lança Consumidor em Números*. 2019. Disponível em: <<https://www.justica.gov.br/news/collective-nitf-content-1552676889.94>>. Citado na página 24.
- Mozilla Developer Network. *Cabeçalhos HTTP - HTTP / MDN*. 2019. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Headers>>. Citado na página 41.
- Mozilla Developer Network. *uma visão geral do HTTP - HTTP / MDN*. 2019. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTTP/overview>>. Citado na página 40.
- NETTO, J. S. D. et al. Melhorando sistemas de social crm por meio de eletronic word-of-mouth. *Revista Eletrônica de Iniciação Científica em Computação*, v. 17, n. 4, 2019. Citado 3 vezes nas páginas 15, 32 e 34.
- PANG, B.; LEE, L. Opinion mining and sentiment analysis. *Comput. Linguist*, v. 35, n. 2, p. 311–312, 2009. Citado na página 52.
- Reclame AQUI. *Sobre o Reclame AQUI - Reclame Aqui*. 2019. Disponível em: <<https://www.reclameaqui.com.br/institucional/>>. Citado na página 38.
- RODRIGUES, L. D.; JÚNIOR, J. L. da S.; LOBATO, F. M. A culpa é dela! e isso o que dizem nos comentários das notícias sobre a tentativa de feminicídio de elaine caparroz. In: SBC. *Anais do VIII Brazilian Workshop on Social Network Analysis and Mining*. [S.l.], 2019. p. 47–58. Citado na página 23.
- Sarma, V. V. S.; Raju, S. Multisensor data fusion and decision support for airborne target identification. *IEEE Transactions on Systems, Man, and Cybernetics*, v. 21, n. 5, p. 1224–1230, Sep. 1991. ISSN 0018-9472. Citado na página 27.
- SOUSA, G. N. d. et al. Análise do setor de telecomunicação brasileiro: Uma visão sobre reclamações. *RISTI-Revista Ibérica de Sistemas e Tecnologias de Informação*, Associação Ibérica de Sistemas e Tecnologias de Informação (AISTI), n. 37, p. 31–48, 2020. Citado 4 vezes nas páginas 15, 32, 33 e 34.
- TEIXEIRA, M. A. M. et al. Um sistema de aquisição e análise de dados para extração de conhecimento da plataforma ebit. *arXiv preprint arXiv:1811.11121*, 2018. Citado 4 vezes nas páginas 15, 31, 32 e 34.
- TRIPP, T. M.; GRÉGOIRE, Y. When unhappy customers strike back on the internet. *MIT Sloan Management Review*, Massachusetts Institute of Technology, Cambridge, MA, v. 52, n. 3, p. 37–44, 2011. Citado 2 vezes nas páginas 23 e 24.
- TUMASJAN, A. et al. Predicting elections with twitter: What 140 characters reveal about political sentiment. In: *Fourth international AAAI conference on weblogs and social media*. [S.l.: s.n.], 2010. Citado na página 23.
- VELASQUEZ, J. D.; GONZALEZ, P. Expanding the possibilities of deliberation: The use of data mining for strengthening democracy with an application to education reform. *The Information Society*, Taylor & Francis, v. 26, n. 1, p. 1–16, 2010. Citado na página 23.

- WALLACH, H. M. Topic modeling: beyond bag-of-words. In: *Proceedings of the 23rd international conference on Machine learning*. [S.l.: s.n.], 2006. p. 977–984. Citado na página 52.
- WANG, C.; BLEI, D. M. Collaborative topic modeling for recommending scientific articles. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. [S.l.: s.n.], 2011. p. 448–456. Citado na página 52.
- WE ARE SOCIAL. Global digital report 2020. *Erişim: <https://wearesocial.com/digital-2020>*, 2020. Citado na página 23.
- WHITE, F. E. *Data fusion lexicon*. [S.l.], 1991. Citado na página 29.
- WIRTH, R.; HIPPEL, J. Crisp-dm: Towards a standard process model for data mining. In: SPRINGER-VERLAG LONDON, UK. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. [S.l.], 2000. p. 29–39. Citado na página 36.
- ZHANG, W.; SKIENKA, S. Trading strategies to exploit blog and news sentiment. In: *Fourth international AAAI conference on weblogs and social media*. [S.l.: s.n.], 2010. Citado na página 23.