
**UNIVERSIDADE FEDERAL DO CEARÁ**

**CAMPUS QUIXADÁ**

**BACHARELADO EM ENGENHARIA DE SOFTWARE**

**ANTONIO CAIO SILVA DE MELO**

# IDENTIFICAÇÃO AUTOMÁTICA DE PROBLEMAS ENCONTRADOS E MUDANÇAS SUGERIDAS POR USUÁRIOS EM SEUS COMENTÁRIOS SOBRE APLICAÇÕES MÓVEIS

**QUIXADÁ – CEARÁ**

**2017**

ANTONIO CAIO SILVA DE MELO

# IDENTIFICAÇÃO AUTOMÁTICA DE PROBLEMAS ENCONTRADOS E MUDANÇAS SUGERIDAS POR USUÁRIOS EM SEUS COMENTÁRIOS SOBRE APLICAÇÕES MÓVEIS

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Orientadora: Prof[a]. Dra. Carla Ilane Moreira Bezerra

Co-Orientadora: Prof[a]. Dra. Ticiana Linhares Coelho da Silva

QUIXADÁ – CEARÁ

2017

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

---

M485i Melo, Antonio Caio Silva de.
Identificação automática de problemas encontrados e mudanças sugeridas por usuários em seus comentários sobre aplicações móveis / Antonio Caio Silva de Melo. – 2017.
90 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2017.
Orientação: Profa. Dra. Carla Ilane Moreira Bezerra.
Coorientação: Profa. Dra. Ticiana Linhares Coelho da Silva.

1. Mineração de dados (Computação). 2. Análise por agrupamento. 3. Software-Controle de qualidade. 4. Aplicativos móveis. 5. Retroalimentação-Usuários. I. Título.

CDD 005.1

---

ANTONIO CAIO SILVA DE MELO

# IDENTIFICAÇÃO AUTOMÁTICA DE PROBLEMAS ENCONTRADOS E MUDANÇAS SUGERIDAS POR USUÁRIOS EM SEUS COMENTÁRIOS SOBRE APLICAÇÕES MÓVEIS

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Aprovado em: _____ / _____ / _____.

BANCA EXAMINADORA

---

Profª. Dra. Carla Ilane Moreira Bezerra (Orientadora)
Universidade Federal do Ceará – UFC

---

Profª. Dra. Ticiana Linhares Coelho da Silva (Co-Orientadora)
Universidade Federal do Ceará - UFC

---

Prof. Dr. Lincoln Souza Rocha
Universidade Federal do Ceará - UFC

À minha mãe, irmãs, família, padrinhos e amigos.

## AGRADECIMENTOS

Em primeiro lugar, agradeço à minha mãe, Jandira, por ser a mulher guerreira e maravilhosa que é, por sempre ter me dado tudo o que pode, por todo o amor e carinho, pelo apoio que me deu do início ao fim dessa jornada e por nunca deixar de acreditar de mim.

Aos meus padrinhos, Amanda e Orlando, pelo o imenso e quase imerecido o apoio ao longo dos últimos 4 anos, pelos conselhos, pela fé em mim e por me habilitarem a concluir este curso de graduação.

Às minhas irmãs, Karleana, Fabiana e Patrícia, e à minha família, por sempre apoiar a mim e à minha mãe, e por sempre nos amar apesar de todos os percalços e dificuldades.

À Prof[a]. Dra. Carla Ilane Moreira Bezerra e à Prof[a] Dra. Ticiana Linhares Coelho da Silva, pela magnífica orientação e auxílio durante a concepção deste trabalho, e por não terem desacreditado que eu o concluiria.

Ao professor participante da banca examinadora Lincoln Souza Rocha pelo tempo e pelas valiosas colaborações e sugestões.

Aos amigos e companheiros que fiz ao longo dos últimos 4 anos, que compartilharam diversos momentos e sentimentos comigo, sejam eles bons ou ruins, e por fazerem parte da minha história na Universidade Federal do Ceará em Quixadá. Em especial, Isaias e Letícia, aos quais credito contribuição direta à elaboração deste trabalho, mas também Ana, Gabriel, Gustavo, Júlio e Marcos. Primeiro, a luta; então, a glória.

Às minhas *waifus* 2D, em especial, Yukino, Umi, Riko, Kaori, Mio, Rem, dentre muitas outras não menos importantes, por me inspirar e auxiliar a suportar as dificuldades enfrentadas em certos momentos da minha graduação, afinal, 2D > 3D.

A todas as pessoas que compõem a Universidade Federal do Ceará, que de uma forma ou de outra influenciaram a minha passagem pela graduação, de forma direta ou indireta, e com maior ou menor intensidade.

Por último, agradeço a mim mesmo, por continuar seguindo em frente na medida do possível, por não cair em frente às barreiras impostas por mim, pelas outras pessoas ou pela vida de forma geral.

“Eu era capaz de resolver fórmulas matemática difíceis sem a ajuda de ninguém, pensei que era natural fazê-lo sozinho. Logo, eu nunca duvidei disso. Mas hoje, à medida que a estação está prestes a acabar, o único comigo é a minha própria sombra.”

(Nagi Yanagi)

## RESUMO

A popularização dos dispositivos móveis e o aumento da sua presença no cotidiano das pessoas induziu ao desenvolvimento de inúmeros sistemas de *software*, construídos especificamente para tal plataforma e conhecidos como aplicativos ou aplicações móveis (*mobile apps*), criados com o intuito de adicionar valor à vida de seus usuários, ao oferecer diversos tipos de funções e serviços. Atualmente, as lojas de aplicativos (*app stores*) centralizam várias operações relacionadas aos aplicativos, e uma delas é permitir que os usuários comentem e atribuam uma nota ao aplicativo. Os comentários podem funcionar como uma fonte de *feedback* direto aos desenvolvedores sobre os diversos aspectos referentes à suas aplicações, que vão variam entre elogios, críticas, sugestões, e assim por diante. Dentre as informações que os usuários provêm, destacam-se as que citam problemas encontrados, sugestões de possíveis mudanças ou melhorias na aplicação, que são capazes de auxiliar diretamente a evoluir seu *software*, pois explicitam ou ressaltam características específicas que ela apresenta, ou que poderiam ser adicionadas. Contudo, embora seja possível aproveitar-se do conteúdo desses comentários para incrementar e melhorar aplicação, é preciso identificar quais deles contém informações consideradas úteis, e quais tópicos os usuários citam. Tal processo exigiria grande esforço e tempo, caso executado de modo manual, principalmente para aplicações mais populares. Neste contexto, o presente trabalho propõe a utilização da clusterização, uma técnica de mineração de dados que realiza agrupamentos de objetos baseando-se no quão similares eles são entre si, para tentar reunir comentários que discorrem sobre os mesmos temas ou aspectos relacionados às aplicações. Para isso, a clusterização foi aplicada em comentários de usuários sobre os aplicativos Facebook e Telegram, em sua versão para o sistema operacional Android, considerando duas medidas de similaridade distintas, com o propósito de analisar como ele pode auxiliar na identificação automática do relato de problemas e solicitação de mudanças. Os resultados mostram que há agrupamentos bastante coesos em relação aos temas em seus comentários, permitindo a derivação de tipos ou classes de problemas a partir da descrição do *cluster*. Além disso, há indícios de que a presença de problemas e a ausência de certas funcionalidades impacta nas notas atribuída a aplicação. Por fim, foi observado que as distâncias consideradas produzem resultados distintos mas, que de modo geral, Jaccard produz *clusters* mais coesos em relação aos tópicos presentes neles.



**Palavras-chave:** Mineração de Dados. Clusterização. Identificação de problemas. Aplicações móveis. Comentários do usuário.

## ABSTRACT

The popularization of mobile devices and the increase of its presence in people's daily lives has induced the development of countless software systems, build specifically for such platform and known as mobile applications, or apps for short, created with the intent of adding value to its users' lives, by offering many kinds of funtionalities and services. Currently, the app stores centralize many operations related to apps, and one of them is allowing users to review and rate the application. Reviews can function as a source of direct feedback to the developers about various aspects that refer to ther apps, that vary between compliments, criticisms, suggestions, and so on. Among the information the users provide, stand out those that mention problems found and suggestions of possible changes or improvements to the application, which are able of directly helping the developers to evolve their software, since they make explicit or emphasize specific features it presents, or that could be added. However, though it is possible to utilize the content of these reviews to increment and improve the app, it is needed to pinpoint which of them contain said useful information, and which topics the users quote. Such process would demand great effor and time, if executed manually, mainly for more popular applications. In this context, the current work proposes the use of clustering, a data mining technique that realizes grouping of objects based on how similar they are with each other, to try gathering reviews that talk about the same themes or aspects app related. Thereunto, the clustering process was applied to user reviews over the Facebook and Telegram apps, in their versions for the Android operating system, taking into account two distinct similary measures, with the purpose of analyzing how it may help the automatic identification of problem reports e change requests. The results show that there are groups very cohesive in respect to the themes in its reviews, allowing the derivation of kinds or classes of problems from the cluster's description. Moreover, there are indications that the presence of problems and the absence of certain features impact the ratings given to the app. Lastly, it was observed that the distances taken into account produce distinct results but, in general, Jaccard generates more cohesive clusters regarding tópicos within them.



**Keywords:** Clustering. Problem identification. Mobile applications. User reviews.

## LISTA DE FIGURAS


Figura 1 – Passos que compõem o KDD. . . . . 21
Figura 2 – Representação simplificada do processo de KDD. . . . . 21
Figura 3 – Processo de Mineração de Textos . . . . . 26
Figura 4 – Procedimentos metodológicos realizados neste trabalho . . . . . 39
Figura 5 – Amostra de comentários coletados utilizando o Heedzy . . . . . 41
Figura 6 – Relação entre o número de *clusters* gerados pela distância de Jaccard e o SSE obtido para os comentários do Facebook. . . . . 47
Figura 7 – Relação entre o número de *clusters* gerados pela distância de Jaccard e o SSE obtido para os comentários do Telegram. . . . . 47
Figura 8 – Distribuição dos comentários do Facebook entre os *clusters* obtido pela distância de Jaccard . . . . . 48
Figura 9 – Distribuição dos comentários sobre o Telegram entre os *clusters* obtidos pela distância de Jaccard . . . . . 54
Figura 10 – Relação entre o número de *clusters* gerados pela distância de Levenshtein e o SSE obtido para os comentários do Facebook. . . . . 60
Figura 11 – Relação entre o número de *clusters* gerados pela distância de Levenshtein e o SSE obtido para os comentários do Telegram. . . . . 61
Figura 12 – Distribuição dos comentários do Facebook entre os *clusters* obtido pela distância de Levenshtein . . . . . 62
Figura 13 – Distribuição dos comentários do Telegram entre os *clusters* obtido pela distância de Levenshtein. . . . . 66
Figura 14 – Nota média atribuída ao Facebook por *cluster* gerado a partir da distância de Jaccard. . . . . 70
Figura 15 – Nota média atribuída ao Telegram por *cluster* gerado a partir da distância de Jaccard. . . . . 72
Figura 16 – Nota média atribuída ao Facebook por *cluster* gerado a partir da distância de Levenshtein. . . . . 74
Figura 17 – Nota média atribuída ao Telegram por *cluster* gerado a partir da distância de Levenshtein. . . . . 76

## LISTA DE TABELAS


Tabela 1 – Comparação entre os trabalhos relacionados e o presente trabalho . . . . . 18
Tabela 2 – Exemplo de representação dos comentários como um vetor de atributos binários. . . . . 34
Tabela 3 – Exemplo do cálculo da distância de Levenshtein. . . . . 36
Tabela 4 – Critérios para selecionar a ferramenta de coleta dos comentários. . . . . 38
Tabela 5 – Número de comentários restantes após o pré-processamento. . . . . 43
Tabela 6 – Valores para o número de *clusters* ($K$) e de iterações ($i$). . . . . 44
Tabela 7 – *Clusters* do Facebook por intervalo de nota média, segundo a distância de Jaccard. . . . . 69
Tabela 8 – *Clusters* do Telegram por intervalo de nota média, segundo a distância de Jaccard. . . . . 71
Tabela 9 – *Clusters* do Facebook por intervalo de nota média, segundo a distância de Levenshtein. . . . . 74
Tabela 10 – *Clusters* do Telegram por intervalo de nota média, segundo a distância de Levenshtein. . . . . 76
Tabela 11 – Exemplo que ilustra a sensibilidade de Jaccard aos termos contidos nos comentários sobre o Telegram. . . . . 81
Tabela 12 – Exemplo que ilustra a sensibilidade de Levenshtein à posição e sequência de caracteres nos comentários sobre o Facebook. . . . . 82
Tabela 13 – Exemplo que ilustra a sensibilidade de Jaccard à quantidade de palavras nos comentários sobre o Facebook. . . . . 82

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| CSV | *Comma Separated Values* ou Valores Separados por Vírgula |
| NLTK | *Natural Language Toolkit* |
| RAM | *Random Access Memory* ou Memória de Acesso Aleatório |
| SSE | *Sum of Squared Errors* ou Soma dos Erros Quadrados |

**SUMÁRIO**


**1 INTRODUÇÃO** . . . . . 13
**2 TRABALHOS RELACIONADOS** . . . . . 17
**3 FUNDAMENTAÇÃO TEÓRICA** . . . . . 20
**3.1 Descoberta do Conhecimento em Bases de Dados** . . . . . 20
**3.2 Mineração de Dados** . . . . . 22
**3.3 Mineração de Textos** . . . . . 24
**3.4 Clusterização** . . . . . 27
***3.4.1 K-medoids*** . . . . . 29
***3.4.2 Medidas de Similaridade*** . . . . . 33
*3.4.2.1 Distância de Jaccard* . . . . . 33
*3.4.2.2 Distância de Levenshtein* . . . . . 35
**3.5 Coleta de comentários em lojas de aplicativos móveis** . . . . . 36
**4 PROCEDIMENTOS METODOLÓGICOS** . . . . . 39
**4.1 Identificar aplicações** . . . . . 40
**4.2 Coletar comentários das aplicações** . . . . . 40
**4.3 Realizar o pré-processamento do conteúdo dos comentários** . . . . . 42
**4.4 Clusterizar os comentários** . . . . . 42
**4.5 Analisar os *clusters* gerados** . . . . . 45
**4.6 Analisar a relação entre os *clusters* das avaliações dos usuários** . . . . . 45
**5 RESULTADOS** . . . . . 46
**5.1 Descrição dos clusters gerados** . . . . . 46
***5.1.1 Distância de Jaccard*** . . . . . 46
*5.1.1.1 Facebook* . . . . . 48
*5.1.1.2 Telegram* . . . . . 53
***5.1.2 Distância de Levenshtein*** . . . . . 60
*5.1.2.1 Facebook* . . . . . 61
*5.1.2.2 Telegram* . . . . . 65
**5.2 Relação entre os clusters formados e as notas dos usuários** . . . . . 69
***5.2.1 Distância de Jaccard*** . . . . . 69
*5.2.1.1 Facebook* . . . . . 69
*5.2.1.2 Telegram* . . . . . 71

*5.2.2* ***Distância de Levenshtein*** . . . . . 73
*5.2.2.1* *Facebook* . . . . . 73
*5.2.2.2* *Telegram* . . . . . 75
**6 DISCUSSÃO** . . . . . 78
**6.1 Conteúdo dos clusters** . . . . . 78
**6.2 Relação entre o conteúdo e a nota média dos *clusters*** . . . . . 79
**6.3 Comparação entre as medidas de distância** . . . . . 80
**7 CONSIDERAÇÕES FINAIS** . . . . . 84
**REFERÊNCIAS** . . . . . 86

# 1 INTRODUÇÃO

Na sociedade contemporânea, o uso de *smartphones*, *tablets*, dentre outros dispositivos móveis é uma prática que faz parte do cotidiano de muitas pessoas. De fato o número de dispositivos utilizados ao longo dos últimos anos é consideravelmente grande e está em crescimento constante, uma vez que milhões de novos aparelhos são registrados a cada ano (OBILE, 2016), com estimativas de que haverão 5.5 bilhões de usuários de dispositivos móveis até 2020 (CISCO, 2017). Tal fenômeno implica no número crescente de aplicações desenvolvidas especificamente para tais dispositivos, alcançando a casa dos milhões apenas nas lojas de aplicativos dominantes (STATISTA.COM, 2017a; STATISTA.COM, 2017b; STATISTA.COM, 2017c).

Aplicações móveis são desenvolvidas com o intuito de solucionar algum problema ou propor algo que agregue valor ao cotidiano das pessoas, desde serviços comunicação, *streaming* de áudio, vídeo, jogos ou ferramentas de organização de tarefas, dentre diversas outras categorias (NICKERSON et al., 2009).

Os aplicativos são distribuídos por meio das lojas (*stores*), que oferecem diversos serviços aos desenvolvedores, permitindo-os publicar seus aplicativos e disponibilizá-los para os usuários do sistema, que poderão baixá-los e instalá-los de forma rápida e prática. Além disso, elas oferecem funções para que os usuários possam avaliar e a publicar comentários sobre a aplicação, cujo conteúdo pode, conter elogios, críticas, solicitação de melhorias e novas funcionalidades, entre outras coisas. Esse *feedback* fornecido pelos usuários possibilita traçar o perfil do aplicativo, com base na opinião e no sentimento de cada indivíduo (HAN; PEI; KAMBER, 2012), construídos a partir da experiência de uso da aplicação.

Em seus comentários, os usuários relatam sobre a sua experiência ao usar o aplicativo (NGUYEN; LAUW; TSAPARAS, 2015), e discorrem sobre como se sentiram ao usar o *software* (e.g., "Amei o aplicativo!"). Eles também provêm informações mais específicas, que agregam valor aos desenvolvedores (INUKOLLU et al., 2014), no que diz respeito à manutenção e evolução da aplicação, tais como: reporte de problemas e/ou *bugs* (PAGANO; MAALEJ, 2013); um sumário do uso de certas funcionalidades (GUZMAN; MAALEJ, 2014); requisição de melhorias (IACOB; HARRISON, 2013); e sugestão de novas funções para a aplicação (CARREÑO; WINBLADH, 2013; GUZMAN; MAALEJ, 2014).

Ainda, durante o uso das aplicações, os usuários podem experienciar diversos tipos de problemas, de lentidão até travamentos e erros de interface do usuário, entre outros. O

*feedback* provido pelos comentários desses usuários pode auxiliar os desenvolvedores a encontrar e corrigir os problemas e *bugs* contidos na seu aplicativo (GAO et al., 2015a). Dessa forma, é esperado que quanto maior o número de usuários que comentam sobre a aplicação, maior seja a quantidade de comentários considerados informativos, que reportam algum problema, requisitam alguma melhoria ou sugerem novas funcionalidades; em detrimento de comentários não informativos, que não oferecem ao desenvolvedor informações claras sobre quais aspectos da aplicação podem ser aprimorados ou incrementados.

Aplicativos podem receber centenas ou até milhares de comentários por dia (CHEN et al., 2014), normalmente variando de acordo com a sua popularidade. Consequentemente, realizar um processo de leitura e análise manual é algo que demanda bastante tempo, à medida que cresce proporcionalmente o número de comentários publicados. No entanto, nem todos os comentários podem ser considerados informativos, no sentido de indicar um *feedback* sobre as funcionalidades da aplicação (CHEN et al., 2014). Há comentários nos quais os usuários expressam a sua opinião e seus sentimentos de forma abrangente, não ressaltando nenhum aspecto específico da aplicação. Essa falta de especificidade por vezes impossibilita que o desenvolvedor identifique as funcionalidades ou características às quais o usuário está se referindo, e por isso esses comentários são não informativos.

A presença de comentários não informativos pode levar os desenvolvedores a analisar uma quantidade de comentários maior que a necessária, uma vez que eles poderiam focar apenas nos comentários que oferecem informações sobre o que pode ser mudado na aplicação. Com isso, ele economizaria não apenas o tempo de filtrar aqueles que provêm alguma informação útil à melhoria da aplicação, mas também de identificar que tipo de informação eles contêm e qual o o valor que eles agregam para a evolução do aplicativo. E mesmo após esse processo custoso, ainda é necessário identificar os problemas e *bugs* reportados, as sugestões de melhorias e de novas funcionalidades, e trabalhar no desenvolvimento de soluções adequadas (CIURUMELEA et al., 2017).

Todo o tempo e esforço designados à análise de comentários poderiam ser aplicados no desenvolvimento de novas funcionalidades, implementação de melhorias e correção de problemas e *bugs*. Portanto, é de interesse do desenvolvedor dispor de uma processo automático de identificação e agrupamento dos comentários, para identificar inicialmente quais reportam problemas, sugerem melhorias ou requisitam novas funcionalidades. Posteriormente, os comentários podem ser divididos de acordo com o tipo de problema ao qual eles se referem,

permitindo a sua priorização de acordo com algum critério, tal como a recorrência ou a sua influência na avaliação atribuída à aplicação. Com a execução desse processo de forma automática, os desenvolvedores são capazes de obter uma visão de alto-nível sobre os aspectos da aplicação que necessitam de maior atenção durante o processo de manutenção, de forma mais fácil e rápida. Assim, eles dispõem de mais tempo para realização das atividades de desenvolvimento pautadas no *feedback* dos usuários, bem como outras atividades de sua preferência (CIURUMELEA et al., 2017).

Há vários trabalhos na área de mineração de dados e obtenção de informações a partir de comentários dos usuários sobre os aplicativos móveis que eles utilizam. Dentre estes, há estudos que mineram avaliações e comentários dos usuários de forma exploratória, analisando a relação entre o tamanho e a categoria do comentário e a nota atribuída ao aplicativo (VASA et al., 2012). Outro define uma forma de classificar os comentários mais informativos para os desenvolvedores (CHEN et al., 2014). Um terceiro estudo analisa os tipos de problemas reportados nos comentários, em diversos sistemas operacionais móveis (MAN et al., 2016).

Nesse contexto, este trabalho tem como objetivo realizar um estudo para identificar e extrair automaticamente informações referentes a problemas em aplicativos, a partir de comentários publicados por usuários, a fim de facilitar a análise desse *feedback* por parte dos desenvolvedores, diminuindo assim o esforço e tempo necessários para essa tarefa. Dessa forma, os desenvolvedores poderão dispor de mais tempo para realizar outras atividades.

Os resultados deste trabalho visam auxiliar os desenvolvedores a entender quais são os tipos de comentários mais comuns sobre as suas aplicações, que problemas e *bugs* são reportados e qual a relação dos tipos de comentários e a avaliação atribuída ao aplicativo. Isso irá facilitar e agilizar a identificação dos pontos principais no *feedback* dos usuários, diminuindo o tempo necessário para análise e, portanto, aumentando o tempo disponível para a manutenção e atualização do aplicativo.

Para este estudo serão consideradas aplicações reais e comentários extraídos da *Google Play Store*, a loja do sistema operacional Android. Aplicativos selecionados foram o Facebook e o Telegram, por serem aplicativos populares, como indicam o número de vezes que cada aplicativo foi baixado; e por pertencerem à categorias que apresentam um volume relativamente maior de *feedback* dos usuários (PAGANO; MAALEJ, 2013).

O trabalho está estruturado da seguinte forma: o Capítulo 2 apresenta os trabalhos relacionados. No Capítulo 3 são expostos os conceitos básicos que são necessários para a

fundamentação do trabalho. O Capítulo 4 descreve a metodologia a ser utilizada. O Capítulo 5 expõe os resultados obtidos a partir da execução da dos procedimentos metodológicos. O Capítulo 6 discute acerca dos resultados, apresentando observações e conclusões oriundas da sua análise. Por fim, no Capítulo 7 são apresentadas considerações a respeito do trabalho, que ressaltam as contribuições e limitações deste, e discorrem sobre as possibilidades para pesquisas futuras.

# 2 TRABALHOS RELACIONADOS

Há diversos estudos voltados para a análise de comentários publicados em lojas de aplicações móveis, utilizando diferentes técnicas de análise, atentando-se a aspectos distintos que são abordados pelos usuários (CHEN et al., 2014; GAO et al., 2015b; MAN et al., 2016).

Em Man et al. (2016), foram minerados comentários de usuários de 20 aplicativos diferentes, em 3 lojas de aplicativos dos três principais sistemas operacionais móveis: *App Store* (iOS), *Google Play Store* (Android) e *Windows Store* (Windows Phone). Foram utilizadas 3 versões dos aplicativos, uma para cada sistema operacional, totalizando 60 aplicações. O estudo propõe um *framework* para analisar e entender qual a distribuição dos tipos de problemas reportados pelos usuários de cada um dos três sistemas operacionais, observando se a frequência de aparição deles é diferente de acordo com o sistema utilizado. O *framework* define sete categorias para problemas nos aplicativos: bateria, interrupção de funcionamento (*crash*), memória, rede, privacidade, *spam* e Interface do Usuário (*User interface* ou UI). Os resultados foram comparados entre os sistemas operacionais, somando as diferentes distribuições dos 20 aplicativos de cada sistema. Por fim, foi apresentada uma análise dos resultados, apontando quais são os principais tipos de problemas reportados pelos usuários em cada sistema operacional.

O trabalho de Man et al. (2016) busca comparar os tipos de problemas reportados entre os diferentes sistemas operacionais móveis, de forma a identificar quais problemas são frequentemente pautados pelos usuários em seus comentários. Já o presente trabalho trabalho busca analisar os comentários e classificá-los de acordo com o tipo de problema reportado. O objetivo principal deste estudo é agrupar e classificar os tipos de comentários de acordo com os seus tipos e os problemas e *bugs* que eles reportam, fornecendo um *feedback* aos desenvolvedores de um aplicativo específico, e não direcionado ao sistema operacional móvel como um todo.

No estudo de Chen et al. (2014), é proposto o AR-Miner, um *framework* computacional para a mineração e análise abrangente de comentários publicados pelos usuários. Tal análise tem como objetivo filtrar os comentários considerados informativos, em detrimento de comentários ruidosos e irrelevantes, ou seja, não-informativos. Em seguida, é realizado a clusterização automática dos comentários baseado em modelagem de tópicos. Posteriormente, é utilizado um esquema de ranqueamento baseado em um modelo de ranqueamento, a fim de priorizar os comentários considerados mais informativos. Por fim, os grupos de comentários classificados como mais informativos são apresentados. Para realizar o estudo, foram utilizados

4 aplicativos desenvolvidos para o sistema Android.

Chen et al. (2014) realizam a análise dos comentários de usuários, ao filtrar os comentários mais informativos, agrupá-los utilizando a modelagem de tópicos e posteriormente os ranqueá-los tanto no nível de grupo quanto no nível de instância. Este trabalho, por sua vez, realiza a classificação dos comentários em classes pré-definidas para os tipos de comentários e os tipos de problemas e *bugs* que eles reportam. Nele, também é realizado a extração dos tipos de problemas e *bugs* que são reportados nos comentários, além de analisar qual impacto eles possuem na nota atribuída ao aplicativo pelos usuários.

Tabela 1 – Comparação entre os trabalhos relacionados e o presente trabalho

| Trabalho | Local de Coleta | Ferramenta Utilizada | Procedimentos | Técnicas Aplicadas |
|---|---|---|---|---|
| (MAN et al., 2016) | Google Play Store, App Store, Windows Store | AppFigures | Extração, Priorização, Recomendação | Redes Neurais, Extração de keywords, Similaridade de cosseno |
| (CHEN et al., 2014) | Google Play Store | Não especificado | Classificação, Clusterização, Ranqueamento | *Expectation maximization* Naive Bayes, Modelagem de tópicos, Modelo de ranqueamento |
| (GAO et al., 2015b) | Google Play Store | Não especificado | Extração de tópicos, Clusterização, Ranqueamento | Modelagem de tópicos, Ranqueamento baseado em similaridade, Distância de Helling |
| Este trabalho | Google Play Store | Heedzy | Clusterização, Classificação, Extração | Similaridade de Jaccard, Classificador Bayesiano Simples |

Fonte – elaborado pelo autor.

Já Gao et al. (2015b) propõem a ferramenta AR-Tracker, desenvolvida para mineração e classificação dos comentários de usuários, com relação ao seu nível de informatividade. O objetivo da ferramenta é realizar o ranqueamento e classificação dos comentários considerando aspectos dinâmicos atrelados a eles, tais como os tópicos discutidos e a versão da aplicação em que o comentário foi publicado ou atualizado. Assim, ela propõe uma abordagem de mineração que difere das abordagens propostas por outras ferramentas e *frameworks* desenvolvidas anteriormente, que definem seu modelo de ranqueamento baseando-se apenas em aspectos estáticos e explícitos dos comentários. O AR-Tracker busca extrair tópicos automaticamente, por meio da análise dos textos nos comentários, sem a necessidade de definições categorias ou classes por parte dos seres humanos. Ademais, ela

permite uma análise de mudanças ao longo do tempo nos principais comentários, permitindo ao desenvolvedor uma visão sobre novas versões do aplicativo.

O estudo conduzido por Gao et al. (2015b) propõe a ferramenta AR-Tracker, cujo objetivo é realizar a classificação e o ranqueamento de comentários dos usuários de acordo com a sua informatividade, considerando aspectos dinâmicos como data de publicação e versão do aplicativo. Ele também realiza a análise da relação entre o conteúdo dos comentários e a nota atribuída ao aplicativo, com foco nos tópicos abordados pelos usuários de forma geral, sem especificar um conjunto finito de classes pré-determinadas. Ainda, ele busca comparar as diferentes técnicas de modelagem de tópicos, para definir qual delas é a mais apropriada para a extração desses tópicos. Em contrapartida, o presente trabalho não considera os aspectos dinâmicos do *feedback* dos usuários, ao se focar somente no conteúdo textual e as notas associadas aos comentários. Ademais, este trabalho pré-define categorias para os tipos de problemas e *bugs* reportados, que serão consideradas no processo de classificação dos comentários. Além disso, este estudo não realiza um ranqueamento, mas buscar fornecer insumos para definição de critérios de priorização ou ranqueamento dos comentários, baseados nos seus tipos e os problemas e *bugs* que eles reportam.

Na Tabela 1 é apresentada uma comparação entre alguns aspectos deste trabalho e dos trabalhos relacionados. As características importantes utilizadas como parâmetro de comparação são: onde os dados foram coletados, que ferramenta foi utilizada para coleta, quais foram os procedimentos executados pelo trabalho e quais foram as técnicas de mineração utilizadas.

# 3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são expostos os conceitos que fundamentam este trabalho. Na Seção 3.1 é apresentada a definição de Descoberta de Conhecimento em Bases de Dados (*Knowledge Discovery in Databases*) e como funciona o seu processo. Em seguida, na Seção 3.2 é discutido sobre o conceito de mineração de dados. Na Seção 3.3, é falado sobre a mineração de textos, um caso específico de mineração de dados. Na Seção 3.4, é apresentado o conceito de Clusterização e quais são algumas das suas técnicas, além do coeficiente de similaridade de Jaccard. Por fim, na Seção 3.5, são apresentadas as ferramentas de coleta dos comentários de aplicações móveis.

## 3.1 Descoberta do Conhecimento em Bases de Dados

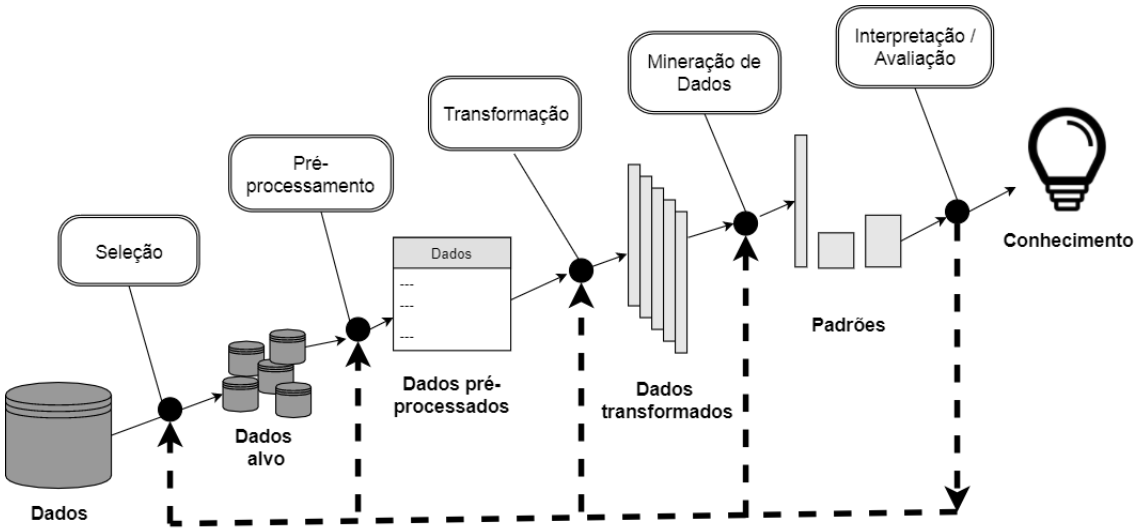
A Descoberta do Conhecimento em Base de Dados (*Knowledge Discovery in Databases* ou *KDD*) é um processo utilizado para transformar dados brutos em informações úteis ao ser humano por meio da identificação de padrões. É um processo geral e não-trivial, no sentido de que não é um somente uma computação direta de quantidades pré-definidas, havendo também a necessidade de pesquisa e inferência em sua execução (TAN; STEINBACH; KUMAR, 2009; FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Segundo Fayyad, Piatetsky-Shapiro e Smyth (1996), a Descoberta do Conhecimento "se refere ao processo geral da descoberta de conhecimento útil por meio dos dados". Eles ainda complementam alegando que "o seu objetivo unificado é extrair conhecimento de alto nível a partir de dados de baixo nível no contexto de grandes conjuntos de dados".

Fayyad, Piatetsky-Shapiro e Smyth (1996) definem o processo como "interativo e iterativo, envolvendo inúmeros passos com várias decisões feitas pelo usuário", cujo fluxo de execução é esquematizado na Figura 1.

A seguir, cada passo do processo é descrito brevemente:

1. Entendimento do domínio de aplicação e do conhecimento prévio que é relevante, e definição do objetivo do processo.
2. Definição do conjunto de dados alvo, e extração daqueles que são mais relevantes para obtenção do conhecimento.
3. Limpeza e pré-processamento dos dados, para diminuição de ruídos nos dados.
4. Redução e projeção dos dados, para tentar diminuir o número de variáveis ou encontrar representações invariantes dos dados.

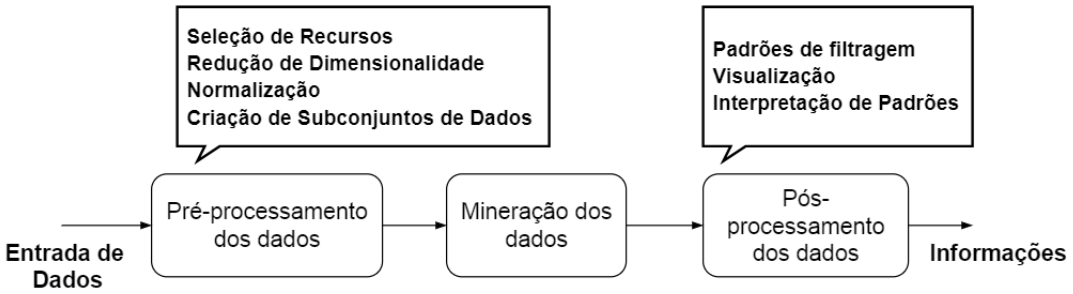
Figura 1 – Passos que compõem o KDD.




Fonte – adaptado de Fayyad, Piatetsky-Shapiro e Smyth (1996).

5. Casamento dos objetivos da processo com a técnica de mineração de dados a ser usada.
6. Análise exploratória dos dados, ao definir técnicas de mineração de dados e métodos de seleção.
7. Realizar efetivamente a mineração, a fim de encontrar padrões nos dados.
8. Interpretar os padrões minerados, através da visualização dos padrões, modelos ou mesmo dados.
9. Utilizar o conhecimento obtido para algo, seja tomada de decisão ou somente reportagem dos resultados.

Figura 2 – Representação simplificada do processo de KDD.




Fonte – adaptado de TAN, STEINBACH e KUMAR (2009).

TAN, STEINBACH e KUMAR (2009) apresentam um modelo simplificado do processo de KDD, ilustrado na Figura 2. As etapas de pré e pós-processamento abstraem parte do processamento realizado no modelo KDD original, todavia sem perder o seu significado. As

três etapas definidas são:

1. Realizar o pré-processamento ao tornar os dados brutos apropriados para análise, através da diminuição de ruído, remoção de nulos e duplicados, entre outras.
2. Executar efetivamente a mineração dos dados, de acordo com o método de mineração pré-definido.
3. Executar o pós-processamento, após a execução da mineração, realizando a apresentação dos resultados ou a sua incorporação com sistemas de apoio a decisões.

A importância do KDD está na inviabilidade dos métodos tradicionais de análise de extrair informações a partir de grandes volumes de dados. Tais métodos se apoiam em um paradigma de hipótese-e-teste, onde uma hipótese é criada acerca de algo, os dados são gerados a partir de um experimento e então comparados com as hipóteses para validá-las ou refutá-las (TAN; STEINBACH; KUMAR, 2009). Além disso, tais processos são comumente executados de forma manual, um método lento, caro e sujeito à subjetividade (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

No contexto deste estudo, o modelo de processo KDD considerado será o de TAN, STEINBACH e KUMAR (2009), pois é mais simples e direto e ainda assim atinge os objetivos do trabalho. Para tal, os passos definidos por ele serão realizados. Ele é importante, uma vez que define não somente o uso da mineração de dados, mais também etapas anteriores e posteriores que são essenciais para a execução da mineração e a obtenção do conhecimento. O processo de Descoberta do Conhecimento será utilizado para agrupar e classificar os comentários em categorias, de acordo com o seu conteúdo, além de extrair os problemas/*bugs* que estão contidos neles.

## 3.2 Mineração de Dados

Com o avanço da tecnologia, o volume de dados produzido e armazenado diariamente cresce constantemente, advindos das mais diversas áreas de atuação humana, tais como negócios, medicina, ciência e engenharia. Isso se deve à presença da computação em praticamente todas essas áreas, bem como da evolução dos métodos de coleta e armazenamento de dados (HAN; PEI; KAMBER, 2012). Dessa forma, um conjunto imenso de dados é formado, que Han, Pei e Kamber (2012) descrevem como “rico em dados, mas pobre em informação“.

Por mais que os dados brutos não agreguem muito valor por si só, eles servem como

base para extração de informações úteis às pessoas e organizações que as geram. Para tal, técnicas de análise tradicionais se mostraram não ser aplicáveis, devido à escalabilidade do conjunto dados e a sua natureza altamente dimensional, complexa e heterogênea. Isso gerou uma demanda por métodos adequados para extração automática de informações relevantes ao ser humano (HAN; PEI; KAMBER, 2012). A mineração de dados surgiu como solução para esse problema, definindo métodos para análise de dados, com o propósito de extrair informações significativas e que permitem a obtenção de um conhecimento acerca de certos aspectos e elementos do mundo real (TAN; STEINBACH; KUMAR, 2009).

A mineração de dados é multidisciplinar, englobando diversas áreas da computação, tais como aprendizagem de máquina, banco de dados e inteligência artificial; e da estatística, como amostragem, estimativa, visualização de dados e teste de hipóteses. (TAN; STEINBACH; KUMAR, 2009; HAN; PEI; KAMBER, 2012).

O processo de mineração de dados é um dos passos do processo de Descoberta de Conhecimento em Bases de Dados. Dentro do KDD, ela é utilizada para aplicação de algoritmos que tem como objetivo encontrar algum padrão relevantes no conjunto de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996).

Han, Pei e Kamber (2012) definem como mineração de dados "o processo de descobrir padrões interessantes e conhecimento de grandes volumes de dados". Já Fayyad, Piatetsky-Shapiro e Smyth (1996) possuem uma vhisão focada no desempenho computacional, em uma perspectiva de aprendizagem de máquina, afirmando que "a mineração de dados é um passo no processo de Descoberta de Conhecimento a partir de Dados que consiste da aplicação algoritmos de análise e algoritmos de descoberta que, sob limitações de eficiência computacional aceitáveis, produzem uma enumeração de padrões (ou modelos) sobre os dados".

Já para (HAND; MANNILA; SMYTH, 2001), "mineração de dados é a análise de conjuntos (frequentemente grandes) de dados observacionais para encontrar relacionamentos inesperados e para resumir os dados de novas maneiras, que sejam ambos compreensíveis e úteis ao seus donos".

A definição adotada por este estudo será aquela dada por TAN, STEINBACH e KUMAR (2009), que define mineração de dados como "o processo de descoberta automática de informações úteis em grandes depósitos de dados". Essa definição foi selecionada pois se encaixa perfeitamente com o que o presente trabalho se propõe a fazer: uma análise automática dos comentários, a fim de agrupá-los e classificá-los, extrair problemas/*bugs* e estudar a relação

entre os tipos de *bugs* reportados e a nota atribuída à aplicação.

A mineração de dados pode ser utilizada para extrair diversas informações a respeito de comentários dos usuários, tais como os sentimentos atribuídos a eles, quais os tópicos mais comumente citados ou quais as versões mais estáveis da aplicação. No contexto deste estudo, ela será utilizada para obter informações úteis em relação ao agrupamento dos comentários publicados, assim como na extração de quais problemas/*bugs* estão contidos neles e qual a sua relação com as notas atribuídas à aplicação. Para isso, serão observados os padrões de similaridade entre o conteúdo dos comentários e a frequência na qual um determinado *bug* é referenciado, e qual a relação entre os tipos de problemas/*bugs* e a nota que os usuários atribuem à aplicação.

## 3.3 Mineração de Textos

De acordo com Tan et al. (1999), "mineração de texto refere-se ao processo de extrair padrões interessantes e não triviais ou conhecimento de documentos de texto não estruturados". Essa definição engloba todos os aspectos relacionados técnicas de recuperação de informação, clusterização e classificação, além de extração de entidades, relações e eventos (KAO; POTEET, 2007). A mineração de textos possui como objetivo principal a derivação de informação de alta qualidade a partir de textos, se utilizando da aprendizagem de padrões, modelagem de tópicos e de linguagem estatística para descoberta de padrões e tendências nos dados (HAN; PEI; KAMBER, 2012).

A mineração de textos é vista como uma extensão ou especialização da mineração de dados, sendo considerada uma tarefa ainda mais complexa, devido à natureza inerentemente não estruturada e confusa dos dados textuais (TAN et al., 1999). Hearst (1999) concorda com essa visão, ao afirmar que "textos expressam um vasto, rico alcance de informações de uma forma que é difícil de se decifrar automaticamente".

Aggarwal e Zhai (2012) complementam ainda afirmando que "dados textuais são dispersos e altamente dimensionais". Com isso, eles se referem ao fato dos documentos textuais possuírem um número variável de palavras, o que torna a complexidade de análise diretamente proporcional à quantidade ou densidade de palavras nos documentos de texto e o número de documentos a serem minerados (AGGARWAL; ZHAI, 2012).

Mathiak e Eckstein (2004) observaram que o processo de mineração de textos pode ser dividido em 5 etapas, cujo fluxo é ilustrado na Figura 3. Aranha (2007) também aborda

um processo semelhante para mineração de textos em seu trabalho, ao propor uma abordagem de pré-processamento utilizando Processamento de Linguagem Natural. Ambos os processos são derivados de uma tendência observada em diversos trabalhos que envolviam mineração de textos. A seguir, os passos do processo são descritos de acordo com o Aranha (2007), Mathiak e Eckstein (2004):

- Coleta: é realizada com o objetivo de determinar o universo de atuação da mineração, determinando o domínio do dados que serão coletado. Após isso, é preciso coletar efetivamente os textos, que constituirão o corpo de documentos de texto a ser minerados. Esse passo é comumente feito através de *web crawlers*, que são aplicações responsáveis por visitar *sites* pela internet e coletar diversos tipos de dados. Como entrada, são definidas as ferramentas a serem utilizadas, o domínio e o tipo de informação que será coletada, incluindo parâmetros especiais para filtragem (MATHIAK; ECKSTEIN, 2004), e os locais de coleta. Como saída, obtém-se os textos coletados;
- Pré-processamento: o objetivo principal da execução deste passo é transformar os textos não estruturados em estruturados, tornando-os propícios para extração do conhecimento. Para isso, é preciso realizar um pré-processamento, que inclui operações como executar *parsing*, adicionar e remover características linguísticas, e outras (HAN; PEI; KAMBER, 2012);

  Isso normalmente é feito utilizando tarefas de Processamento de Linguagem Natural, tais como tokenização, stemização, lematização e *Part-of-Speech Tagging* (MATHIAK; ECKSTEIN, 2004). A entrada para esse passo são os textos brutos coletados previamente, e a sua saída são os textos estruturados, geralmente em formato de tabelas de valor-atributo.
- Indexação: esse passo é realizado com o intuito de agilizar a busca e a recuperação de dados através de certos mecanismos e técnicas, tais como listas invertidas e similaridade entre documentos de texto. A indexação é importante para grandes conjuntos de dados textuais, já que a realização de buscas em texto envolve um processo de comparação de caractere-por-caractere, a fim encontrar um termo ou palavra desejados. Devido à isso, tal processo é computacionalmente complexo e, portanto, custoso.

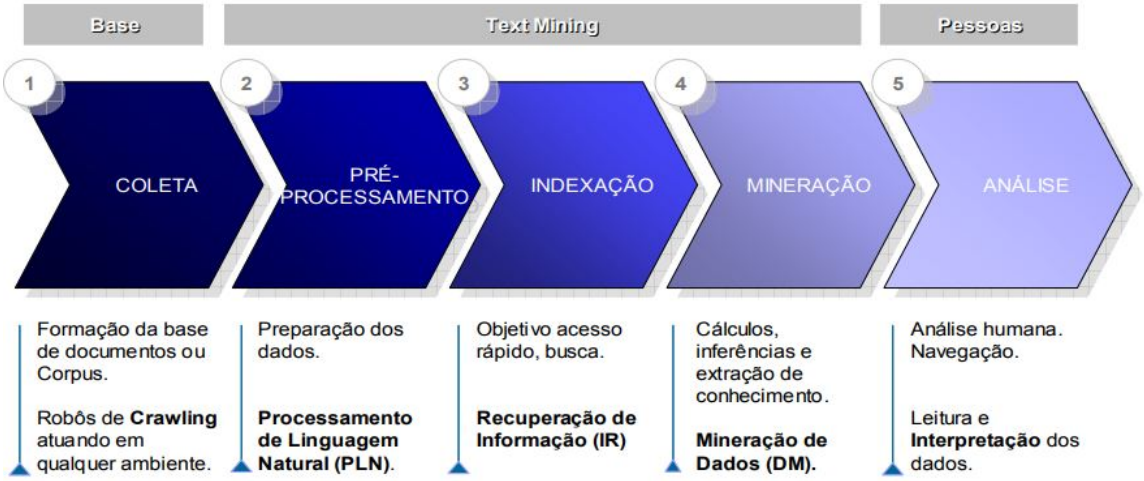
  Como entrada, o passo necessita dos textos pré-processados. E como saída, ele fornece o conjunto de termos ou índices que referenciam os documentos de texto dos quais eles foram extraídos;
- Mineração: é o passo onde são definidas quais técnicas e algoritmos serão utilizados para

minerar os dados, a fim de identificar os padrões e extrair informações. A seleção do algoritmo considera o tipo de dado e o quais informações se deseja obter a partir deles. Dentre os algoritmos de mineração de textos, temos os de classificação (Classificadores Bayesianos, Redes Neurais, *Support Vector Machines* (SVM)), de clusterização (*K-means*, *K-medoids*, DBSCAN, Agrupamento Hierárquico Aglomerativo) e de Análise de Sentimento.

Como entrada, o passo exige os documentos pré-processados e indexados (se aplicável). A saída são os resultados gerados pelos algoritmos, que dependem do objetivo da mineração: classificar, clusterizar, associar, analisar sentimentos, etc;

- Análise: o último passo envolve a análise dos resultados obtidos após a mineração. Como entrada, além das saídas geradas pelos algoritmos de mineração, obtém-se também resultados relacionados á execução da mineração, como a taxa de erro do algoritmo e tempo de CPU. Tais resultados servem como base para obtenção de conhecimento e tomada de decisões acerca do domínio selecionado. A saída desse passo é o entendimento dos resultados e o conhecimento gerado por eles.

Figura 3 – Processo de Mineração de Textos




Fonte – (ARANHA, 2007).

O processo de mineração de textos será utilizado neste estudo para identificar padrões nos comentários publicados pelos usuários sobre aplicações móveis. As técnicas de mineração aplicadas serão a clusterização dos comentários, utilizando o algoritmo *K-medoids* juntamente com coeficiente de similaridade de Jaccard; e a classificação dos comentários, primeiramente nas categorias pré-definidas para os tipos de comentários (problema/*bug*,

melhoria, nova funcionalidade e não-informativo), e posteriormente dos comentários que reportam problemas/*bugs* nas categorias de tipos de *bugs* também previamente definidas.

No que diz respeito à clusterização, após a sua execução será feita uma análise para verificar se os *clusters* gerados automaticamente correspondem ás categorias de comentários pré-definidas. Em caso afirmativo, o primeiro processo de classificação dos comentários não precisará ser executado; em caso negativo, será analisada a possibilidade da utilização de alguns dos *clusters* gerados como treino para o processo de classificação inicial.

## 3.4 Clusterização

Clusterização ou análise de *clusters* é o processo que busca dividir um conjunto de dados em grupos ou *clusters*, que apresentam algum significado ou utilidade (TAN; STEINBACH; KUMAR, 2009). Ela é feita de forma automática, por meio de algoritmos de clusterização, e se utiliza exclusivamente dos dados que descrevem os objetos e as suas relações. O objetivo da clusterização é juntar objetos semelhantes (ou relacionados) em um mesmo grupo, e objetos diferentes (ou não relacionados) em grupos distintos (TAN; STEINBACH; KUMAR, 2009). Com isso, ela permite a descoberta de grupos coesos e previamente desconhecidos do conjunto de dados (HAN; PEI; KAMBER, 2012).

A análise de *clusters* é utilizada para obtenção de *insights* sobre a distribuição dos dados, além da observação das características dos *clusters* obtidos, permitindo o enfoque em um conjunto específico deles. Ela também pode servir como pré-processamento para outras técnicas, como caracterização e classificação (HAN; PEI; KAMBER, 2012). Além disso, a clusterização pode servir como método de classificação, que considera apenas os dados que descrevem aos objetos, em detrimento da classificação, que atribui rótulos objetos conhecidos e os utiliza para classificar os demais objetos do conjunto. Por isso, a clusterização é por vezes chamada de classificação não-supervisionada ou classificação automatizada (TAN; STEINBACH; KUMAR, 2009; HAN; PEI; KAMBER, 2012).

Em um processo de clusterização, o conjunto de todos os objetos de dados é denominado agrupamento. Há diversos tipos de agrupamentos, definidos a partir de três características principais: a existência de aninhamentos, intersecções e o nível de cobertura da clusterização. A seguir, são apresentados os possíveis valores para cada uma das três características, respectivamente, de acordo com TAN, STEINBACH e KUMAR (2009):

- **Hierárquicos ou Particionados:** em um agrupamento hierárquico, os grupos podem estar

aninhados entre si, organizados uma espécie de árvore ou hierarquia, onde cada nó (grupo) da árvore é um grupo formado pela união de seus filhos (subgrupos). Dessa forma, a raiz é o grupo que contém todos os objetos, e as folhas geralmente são os grupos únicos, com objetos individuais (TAN; STEINBACH; KUMAR, 2009).

Já nos agrupamentos particionados, os grupos não podem ser aninhados e, portanto, não possuem intersecção. Assim, cada objeto pertence a apenas um conjunto (TAN; STEINBACH; KUMAR, 2009);

- **Exclusivos, Interseccionados ou Difusos:** agrupamentos exclusivos são aqueles que atribuem cada objeto a um único grupo. Em contrapartida, os agrupamentos interseccionados ou não-exclusivos permitem que um objeto pertença a mais de um grupo, simultaneamente. Há também casos de agrupamentos interseccionados onde um objeto está "entre"dois ou mais grupos, podendo ser atribuído a qualquer um deles. Nesse caso, ao invés de escolher um dos grupos, o objeto é atribuído a todos eles (TAN; STEINBACH; KUMAR, 2009).

  Já o agrupamento difuso considera o conceito matemático de conjuntos difusos, em que um objeto pode pertencer a um grupo de acordo com um peso, que varia entre 0 (não pertence totalmente) e 1 (pertence totalmente), e cuja soma deve ser exatamente 1. Isso difere do agrupamento interseccionado, uma vez que um objeto não pode pertencer totalmente a mais de um grupo (TAN; STEINBACH; KUMAR, 2009);
- **Completos ou Parciais:** os agrupamentos completos atribuem todos os objetos do conjunto a pelo menos um grupo, enquanto o agrupamento parcial permite que objetos não pertençam a nenhum grupo. Esses objetos são tidos como ruídos (*outliers*) ou elementos externos, que não se assemelham a outros objetos (TAN; STEINBACH; KUMAR, 2009).

Há diversos métodos para execução da análise de *clusters* e, para cada um deles, há um conjunto de algoritmos conhecidos que o implementa. É importante salientar que o algoritmo usado para clusterizar um conjunto de dados influencia diretamente no resultado, uma vez que os *clusters* obtidos após a clusterização variam de algoritmo para algoritmo. Os métodos básicos de clusterização, e alguns de seus algoritmos, são: particionamento ( *K-means*, *K-medoids*), hierarquia (Agrupamentos Hierárquico Aglomerativo, BIRCH), baseados em densidade (DBSCAN, OPTICS) e baseados em grade (STING, CLIQUE) (HAN; PEI; KAMBER, 2012).

O método selecionado para conduzir a clusterização neste trabalho será o

particionamento, e o algoritmo utilizando será o *K-medoids*. O conjunto de dados a ser clusterizado será considerado um agrupamento particionado, exclusivo e completo. Em outras palavras, isso significa que não haverão aninhamentos entre os *clusters* e todos os comentários deverão obrigatoriamente pertencer a um, e somente um *cluster*. A clusterização será executada como um processo de classificação não-supervisionada inicial dos comentários. Os *clusters* gerados pelo processo serão analisados, para identificar os grupos de comentários similares e sobre quais assuntos eles tratam, como problemas gerais ou específicos, entre outros.

### *3.4.1 K-medoids*

O *K-medoids* é um algoritmo de particionamento baseado em distância, bastante semelhante ao *K-means*, e baseado em protótipos, pois se utiliza objetos que representam os *clusters* - denominados *medoids* - para realizar a clusterização (TAN; STEINBACH; KUMAR, 2009). Segundo Han, Pei e Kamber (2012), "dado um conjunto de dados *D* de *n* objetos, e um número *k* de *clusters* a serem formados, um algoritmo de particionamento organiza os objetos em *k* partições, tal que ($k <= n$), onde cada partição representa um *cluster*". O número de *clusters* gerados é otimizado de acordo com algum critério de formação que, no caso dos algoritmos de baseados em distância, são as funções de similaridade/dissimilaridade (HAN; PEI; KAMBER, 2012; AGGARWAL; ZHAI, 2012).

O algoritmo foi proposto como solução para uma desvantagem que o *K-means* apresenta: a sensitividade a ruídos ou *outliers* (HAN; PEI; KAMBER, 2012). Isso se dá pois o *K-means* clusteriza os dados baseando-se na média dos valores pertencentes a cada *cluster*. Assim, no momento em que um elemento com valor muito discrepante é atribuído a um *cluster*, o valor da média é alterado drasticamente (HAN; PEI; KAMBER, 2012). Esse problema é agravado pelo uso da Soma dos Erros Quadrados (*Sum of Squared Errors*) para medição da qualidade dos *clusters* gerados durante o processo de clusterização.

A fim de solucionar tal problema, o *K-medoids* se utiliza do critério de erro absoluto (*absolute-error criterion*) para medir a qualidade dos *clusters*. O valor do critério de erro absoluto é obtido pela soma das diferenças ou dissimilaridades entre os objetos do conjunto de dados e os objetos que representam os seus respectivos *clusters*, denominados *medoids*. A sua fórmula de cálculo é dada por

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, o_i) \tag{3.1}$$

onde $k$ é o número de partições conhecidas *a priori*, $p$ é um objeto qualquer do conjunto de dados, $o_i$ é o *medoid* do *cluster* $C_i$ e $dist(x,y)$ é a função que calcula a distância entre dois pontos $x$ e $y$ (HAN; PEI; KAMBER, 2012).

Já em relação à forma como os objetos representativos são selecionados, Aggarwal e Zhai (2012) afirmam que, em um contexto de clusterização de documentos textuais, "o objetivo chave do algoritmo é determinar um conjunto ótimo de documentos representativos a partir do *corpus* original dos quais os *clusters* foram gerados". O *K-medoids* extrai os objetos representantes (*medoids*) a partir do próprio conjunto de objetos, em detrimento do *K-means*, por exemplo, cujos *centroids* são definidos à partir da média dos valores de elementos que pertencem aos *clusters*. Um dos objetivos do algoritmo é que, ao final da clusterização, todos os *clusters* formados apresentem o menor critério de erro absoluto possível.

O processo de clusterização descrito por Han, Pei e Kamber (2012) é dividido em 3 passos:

1. Um número $k$ de objetos são selecionados aleatoriamente a partir do conjunto de todos os objetos, em que $k$ é o número de *clusters* que se deseja obter. Os objetos selecionados são então divididos em $k$ *clusters*, e cada objeto é considerado o representante ou *medoid* do seu respectivo *cluster*.

   Então, calcula-se a similaridade (ou distância) entre os objetos do conjunto de dados e os *medoids*, de acordo com função de similaridade definida. Após o cálculo das similaridades, os demais objetos são atribuídos ao *cluster* representado pelo *medoid* mais similar (ou próximo) a ele. Ao final desse processo, todos os objetos terão sido atribuídos a um e somente um *cluster*.
2. Um objeto $o_{aleatorio}$ é escolhido a partir do conjunto de objetos que não são *medoids*. Ele é então usado para substituir o *medoid* $o_j$ pertencente ao conjunto $\{o_1,\ldots,o_j,o_j,o_{j+1},\ldots,o_k\}$ de *medoids*, em que $(1 \leq j \leq k)$. Com isso, é formado um novo conjunto $\{o_1,\ldots,o_j,o_{aleatorio},o_{j+1},\ldots,o_k\}$. Logo após, a similaridade (distância) entre todos os $k$ objetos e os *medoids* é recalculada, considerando o novo conjunto formado.
3. Para todos os objetos, é verificado se o *medoid* mais similar a ele mudou, e em caso afirmativo, o objeto é atribuído a ele. Assim, como o conjunto contém $o_{aleatorio}$ como *medoid*, é possível que parte dos objetos sejam atribuídos a ele.

   Após a verificação de todos os objetos, um novo valor para o critério de erro

absoluto $E$ é obtido. Caso o novo valor de E seja maior que o anterior, isso significa que ter $o_{aleatorio}$ como *medoid* diminui a dissimilaridade dos *clusters* e, portanto, que $o_{aleatorio}$ deve se tornar um *medoid* e substituir $o_j$; caso ele seja menor, isso quer dizer que $o_{aleatorio}$ aumentou a dissimilaridade e, assim, não deve se tornar um *medoid*.

Os passos 2 e 3 são iterados até que todas as possibilidades para substituições de *medoids* por objetos não-representativos tenham sido executadas. Essa é uma das principais desvantagens do *K-medoids*, pois cada iteração exige a execução da função de similaridade para todas as combinações de todos objetos do conjunto, o que torna o tempo de execução diretamente proporcional ao tamanho do conjunto. Por isso, ele é não é adequado para conjuntos de dados grandes, por ser computacionalmente complexo, da classe problemas *NP-Difícil* para $k \geq 1$ *clusters* (HAN; PEI; KAMBER, 2012). Todavia, como o conjunto de dados a ser utilizado neste estudo é relativamente pequeno, tal característica do algoritmo não será um empecilho ao seu uso.

Os algoritmos baseados em distância se utilizam da similaridade dos objetos para geração dos *clusters*, ao agrupar os objetos considerados similares. Para calcular o grau de similaridade, é preciso definir uma função objetiva de similaridade entre os objetos do conjunto de dados (AGGARWAL; ZHAI, 2012). Há diversas medidas que podem ser usadas por essas funções, como a similaridade de cosseno e a correlação. A função de similaridade utilizada neste trabalho para realizar a clusterização dos comentários será o coeficiente de similaridade de Jaccard.

No presente trabalho, o *K-medoids* será utilizado para realizar a clusterização de comentários dos usuários. Ele foi escolhido pois é baseado em distância/similaridade e, portanto, realizará o agrupamento dos comentários mais similares, dos quais espera-se que falem sobre os mesmos assuntos, tais como problemas encontrados ou melhorias sugeridas. Além disso, o algoritmo é adequado à clusterização de textos, pois os medoids que representam os *clusters* são obtidos a partir dos próprios objetos de dados, ao invés de serem criados a partir de uma média dos elementos do *cluster*, como acontece no *K-means*. Assim, ele não cria valores virtuais para os dados, utilizando os valores reais dos objetos. Outro motivo para a sua seleção é que não faria sentido se utilizar algoritmos baseados em densidade, como o DBSCAN, visto que o conhecimento sobre a densidade dos textos não é uma informação relevante no contexto deste estudo.

Uma das características do *K-medoids* é a necessidade de definição prévia do número *K* de agrupamentos que se deseja obter, antes mesmo de executar a clusterização. Isso não se apresenta como um problema quando se tem conhecimento do número de *clusters* desejado. Entretanto, nos casos em que tal número não é conhecido, não há uma regra fixa para definir o melhor número de grupos a serem gerados, uma vez que o valor ideal varia de acordo com o conjunto de dados a ser clusterizado, bem como que tipo de informações se deseja obter a partir deles.

Todavia, é possível construir uma heurística para identificação do melhor valor ou intervalo de valores para *K*, a partir da execução uma exaustiva da clusterização para diversos valores. Assim, executa-se o *K-medoids* para uma faixa pré-definida de valores e, a partir da qualidade dos resultados de cada um deles, é escolhido o melhor número ou intervalo de números de *clusters* iniciais.

Como mencionado anteriormente, a qualidade da clusterização pode ser definida por meio do Critério de Erro Absoluto, também denominado Soma dos Erros Quadrados (*Sum of Squared Errors* ou SSE), cujo cálculo é apresentado na Fórmula 3.1. Quanto menor o valor do SSE, maior é a qualidade da clusterização. O número *K* de *clusters* iniciais influencia na qualidade interna dos *clusters*, de modo há uma relação diretamente proporcional entre ambos. Segundo essa relação, o resultado de menor qualidade é aquele que considera apenas um *cluster* inicial; em contrapartida, para um número inicial de *clusters* igual ao número de todos os textos do conjunto de dados, a qualidade interna é a maior possível, já que cada texto seria um *medoid* e, portanto, seria inicialmente atribuído ao seu próprio grupo. Para o último caso, contudo, por mais que a qualidade seja a melhor possível, os grupos gerados não são úteis para análise, uma vez que cada grupo corresponde a um único texto.

Dado um conjunto de documentos de texto, ao projetar a relação entre os diferentes valores de *K* e os seus respectivos SSEs em um plano cartesiano, é possível observar que a curva de variação é decrescente e se estende até o melhor caso, onde o número de *clusters* é igual ao número de documentos e o SSE é 0. Apesar disso, há um ponto em que há certa estabilização, no qual a diferença entre os SSEs de um valor de *K* e o seu subsequente é muito pequena. Esse ponto onde a estabilização se inicia pode ser considerado o melhor valor de *clusters* iniciais para execução da clusterização, já que, por mais que os valores iniciais subsequentes possuam um menor SSE, a diferença é ínfima e não acompanha o crescimento do número de grupos iniciais.

Neste trabalho, a clusterização será executada de forma a obter os agrupamentos sem

conhecer de antemão quais são os tipos de grupos que podem ser obtidos a partir dos comentários. Dessa forma, o número ideal de grupos a serem obtidos não é conhecido de antemão. Portanto, o processo de clusterização será executado considerando uma faixa de valores para o número de *clusters* distintos, e a partir dos resultados, escolher qual valor é mais adequado para o conjunto de comentários analisado.

Outra característica do algoritmo é que, devido a sua natureza, os resultados do *K-medoids* podem variar de uma execução para outra, ainda que o valor do número de *clusters* inicias seja o mesmo e, portanto, a qualidade da clusterização também varia a cada execução. Isso se deve ao fato dos *medoids* iniciais serem atribuídos de forma aleatória. Levando isso em consideração, para cada valor de $K$, a clusterização pode executada $i$ vezes (iterações), e apenas a iteração que apresentou melhor qualidade interna dos *clusters* foi considerada.

### *3.4.2 Medidas de Similaridade*

A semelhança, similaridade ou proximidade entre dois objetos pode ser definida como o valor numérico que representa o grau em que eles se parecem (TAN; STEINBACH; KUMAR, 2009). Quanto mais parecidos, maior é a similaridade entre eles, e vice-versa. As medidas de similaridade (ou proximidade) são utilizadas para calcular o quão similares dois objetos são, e definem como os seus atributos influenciam similaridade entre eles. Para objetos simples, com apenas um atributo, a proximidade é calculada diretamente a partir do valor do atributo. Já para objetos compostos, que possuem vários atributos, ela é calculada a partir da similaridade de cada um dos atributos (TAN; STEINBACH; KUMAR, 2009).

Como discorrido na Seção 3.4, há processos de clusterização que se baseiam na similaridade entre os objetos. O nível com que dois objetos são similares pode ser definido de diversas formas, considerando aspectos distintos que os dados apresentam. Tais formas são definidas pelas medidas de similaridade. Há diversas medidas, como o Coeficiente de Correspondência Simples (SMC), a similaridade do cosseno, correlação e as divergências de Bregman (TAN; STEINBACH; KUMAR, 2009). Neste trabalho, para clusterizar os comentários, foram consideradas duas medidas: a distância de Jaccard e a distância de Levenshtein.

#### *3.4.2.1 Distância de Jaccard*

A similaridade de Jaccard é uma função utilizada para medir a semelhança entre objetos compostos de atributos binários assimétricos (HAN; PEI; KAMBER, 2012) a partir da

intersecção dos atributos que os objetos possuir. Atributos binários assimétricos são atributos que assumem apenas os valores 1 e 0, indicando sua presença e ausência, e que consideram importante qualquer valor que seja diferente de 0 (TAN; STEINBACH; KUMAR, 2009). Isso significa que um objeto possui um atributo se e somente se o seu valor for igual a 1; caso contrário, o valor do atributo é 0, indicando que o objeto não o possui.

O coeficiente de Jaccard define a similaridade dos objetos através da divisão do número de atributos que eles têm em comum pela soma de todos os seus atributos. Quanto mais atributos eles tiverem em comum, mais semelhantes eles são. Dessa forma, dois objetos são considerados completamente iguais quando possuem exatamente os mesmos atributos; em contrapartida, eles são completamente diferentes quando não possuem nenhum atributo em comum. A fórmula para cálculo do coeficiente de Jaccard é dada por

$$J(i,j) = \frac{|i \cap j|}{|i \cup j|} \tag{3.2}$$

onde $i$ e $j$ são dois objetos quaisquer que possuem atributos comparáveis entre si (HAN; PEI; KAMBER, 2012; TAN; STEINBACH; KUMAR, 2009).

Tabela 2 – Exemplo de representação dos comentários como um vetor de atributos binários.

| Comentário | Texto | "aplicativo" | "demais" | "inicial" | "na" | "o" | "tela" | "trava" |
|---|---|---|---|---|---|---|---|---|
| A | "trava na tela inicial" | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| B | "o aplicativo trava demais" | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

Fonte – elaborado pelo autor.

No contexto deste trabalho, que considera os comentários de usuários (dados textuais), cada palavra presente no texto é um atributo, e juntas formam o vetor de atributos do objeto, com valor 1 como padrão (a palavra está presente no comentário). Logo, o grau de similaridade de Jaccard entre dois comentários é medido pela entre a ao tamanho da intersecção de palavras presentes em ambos, e o tamanho do conjunto união das palavras únicas de cada comentário.

Como exemplo, observe a Tabela 2. Nela há dois comentários, A e B, e seus atributos (palavras) dispostos ao longo das colunas. Caso a palavra esteja presente no texto do comentário, ela recebe o valor 1; caso contrário, recebe o valor 0. Note que a repetição de palavras é desconsiderada. Como pode ser observado, os dois comentários possuem apenas 1 palavra em comum ("trava"), e ao todo possuem 7 palavras distintas. Logo, a similaridade entre eles é

calculada como

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|1|}{|7|} \approx 0.143$$

Assim, com um valor de similaridade de aproximadamente 0.143, é possível concluir que os comentários A e B são pouco similares.

#### *3.4.2.2 Distância de Levenshtein*

Também conhecida como Distância de Edição, a distância de Levenshtein é definida por Pieterse e Black (2015) como "o menor número de inserções, deleções e substituições necessárias par transformar um texto (*string*) em outro". Dados dois textos, a distância de Levenshtein calcula o número mínimo de operações (adição, exclusão e substituição) sobre os caracteres que as compõem é preciso para tornar os dois textos idênticos. O cálculo permite que as operações possuam diferentes pesos em relação à distância, de acordo com a necessidade ou a aplicação da medida.

Um algoritmo recursivo para cálculo de Levenshtein pode representado matematicamente por

$$L_{A,B}(i,j) = \begin{cases} max(i,j) & \text{se min(i, j) = 0} \\ min \begin{cases} L_{A,B}(i-1,j)+1 \\ L_{A,B}(i,j-1)+1 \\ L_{A,B}(i-1,j-1)+1_{(\text{se } A_i \neq B_j)} \end{cases} & \text{caso contrário.} \end{cases} \tag{3.3}$$

onde $L_{A,B}(i,j)$ é a função de cálculo da distância de Levenshtein entre dois textos $A$ e $B$, de cumprimento $i$ e $j$, respectivamente. As funções $min(i,j)$ e $max(i,j)$ retornam o maior e o menor valor entre $i$ e $j$, nessa ordem. Caso um dos textos seja vazio, o resultado é o cumprimento do outro; caso contrário, são feitas chamadas recursivas à função, sendo considerada aquela que retornar o menor valor.

O valor retornado pelo algoritmo é literalmente o número de operações para transformar um texto em outro. Entretanto, é possível normalizar tal resultado, através da sua divisão com o resultado do pior caso possível, e cujo valor é igual ao cumprimento do maior entre os dois textos. Logo, a normalização dos resultados de Levenshtein é dada por

$$L_{norm(A,B)}(i,j) = \frac{L_{A,B}(i,j)}{max(i,j)} \tag{3.4}$$

Tabela 3 – Exemplo do cálculo da distância de Levenshtein.

| **Palavra** | **Cumprimento** | **Caracteres** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| aplicativo | 10 | a | p | l | i | c | a | t | i | v | o |
| aplicavel | 9 | a | p | l | i | c | a | v | e | l | |
| **Operações** | | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

Fonte – elaborado pelo autor.

A Tabela 3 mostra um exemplo bastante simplificado de como Levenshtein calcula a distância entre as palavras "aplicacao"e "aplicavel", desconsiderando-se os caracteres especiais. Baseado no número de operações para transformar a maior palavra (aplicativo) na menor (aplicavel), é possível ver que são necessárias 4 operações. Logo, a distancia normalizada entre elas é

$$L_{norm(aplicativo,aplicavel)}(10,9) = \frac{L_{(aplicativo,aplicavel)}(10,9)}{max(10,9)} = \frac{4}{10} = 0.4$$

Vale ressaltar que, nesse exemplo, todas as operações possuem peso 1.

## 3.5 Coleta de comentários em lojas de aplicativos móveis

Até o momento da elaboração deste trabalho, o Google não disponibiliza nenhuma API ou ferramenta oficial para coleta de comentários em massa, de aplicativos distintos. Há a Google Developers API[1], que oferece funções para desenvolvedores de aplicações Android, porém em relação a comentários, ela permite apenas a coleta de comentários das aplicações que o desenvolvedor desenvolveu e publicou.

Apesar disso, atualmente há diversas opções para coleta de informações sobre aplicativos móveis publicados nas principais lojas de aplicativos do mercado. A seguir, algumas dessas ferramentas serão descritas.

AppFigures[2] é uma ferramenta paga para análise de informações sobre aplicativos móveis em diversas lojas de aplicativos. Ele oferece três produtos distintos:

- **Análise**: voltada para desenvolvedores, oferecendo um ferramentas para análise de ganhos com vendas, propaganda, entre outros.
- **Pesquisa**: voltada para pesquisadores, oferece um conjunto de ferramentas para análise, acerca de diversos aspectos dos aplicativos, como filtros e q*queries* para pesquisa,

[1] <https://developers.google.com/>
[2] <https://appfigures.com/>

estatísticas sobre preços e desempenho dos aplicativos, entre outros dados.

- **Appbase**: uma base de dados com informações sobre mais de 4 milhões de aplicativos, para quem deseja manipular tias informações de forma direta.

AppTweak[3] é outra ferramenta que oferece análise de dados sobre aplicativos móveis. Embora contenha apenas informações sobre aplicações publicadas nas lojas App Store e Google Play Store, eles oferecem diversas informações, incluindo dados sobre os comentários e avaliações. Todas essas informações podem ser acessadas através de uma API RESTful.

MarketBot[4] é um projeto *open-source*, com código disponível no GitHub, e desenvolvido na linguagem Ruby.A ferramenta é um *web scraper*, ou seja, uma aplicação feita para extrair dados de websites. Ela permite a coleta de dados dos aplicativos, gráficos e desenvolvedores. Por ser o*open-source*, qualquer modificação necessária ou desejada para coleta dos dados pode ser realizada. Um dos pontos negativos da ferramenta é que ela não permite a coleta de um grande volume de dados.

A ferramenta Heedzy[5]. Ela permite o *download* de dados sobre aplicações móveis publicadas na *App Store* e *Google Play Store*, e também de postagens no Twitter. É possível baixar comentários da última semana, mês ou ano, e até todo o histórico de comentários, partindo do momento em que a ferramenta começou a acompanhar os comentários sobre a aplicação. Ela permite baixar apenas 100 comentários de qualquer aplicativo gratuitamente, porém a partir dessa quantidade é necessário ter um plano *premium*.

Há também o 42Matters[6], uma ferramenta que permite a exploração de dados sobre os aplicativos. Com ela, você pode analisar desde aspectos relacionados ao mercado de aplicações móveis, tais como *insights* sobre novas ideias ou o que é popular no momento, bem como dados específicos das aplicações, dentre eles os comentários postados sobre os aplicativos. Ela conta com uma visualização dos dados diretamente na ferramenta, e ainda possui uma API para acesso personalizado. Apesar de ser uma ferramenta paga, ela possui um período de *free-trial*.

Na Tabela 4 é possível visualizar um resumo da comparação entre as características das ferramentas. Os critérios para seleção da ferramenta são: se a ferramenta possui *free trial* ou versão *free*, as suas limitações, se ela disponibiliza uma API para acesso aos dados e o nível de esforço necessário para utilizá-la. Neste último critério, não foi considerado apenas o esforço de usar as funções da ferramenta, mas também o conhecimento necessário para utilizá-la, por

[3] <https://apptweak.io/>
[4] <www.github.com/chadrem/market_bot>
[5] <https://heedzy.com/>
[6] <https://42matters.com/>

Tabela 4 – Critérios para selecionar a ferramenta de coleta dos comentários.

| Ferramenta | *Free* ou *Free trial* | Limitações | Possui API | Nível de Esforço |
|---|---|---|---|---|
| **AppFigures** | Não | *Free trial* de apenas 14 dias | Sim | Médio |
| **AppTweak** | Não | Operações na ferramenta custam créditos. Para contas *free*, são oferecidos apenas 100 créditos/mês. | Sim | Médio |
| **MarketBot** | Sim | Ao tentar coletar um volume excessivo de dados, o IP do usuário pode ser bloqueado pelo Google. | Não | Alto |
| **Heedzy** | Não | Na versão *free*, o usuário pode acompanhar no máximo 5 aplicativos, e possui acesso aos 100 últimos comentários, apenas. | Não | Baixo |
| **42Matters** | Não | *Free trial* de apenas 14 dias, e há um limite de acesso dos dados através da API. | Sim | Médio |

Fonte – elaborado pelo autor.

exemplo, a necessidade de aprender uma nova linguagem de programação ou se e preciso estudar sua API.

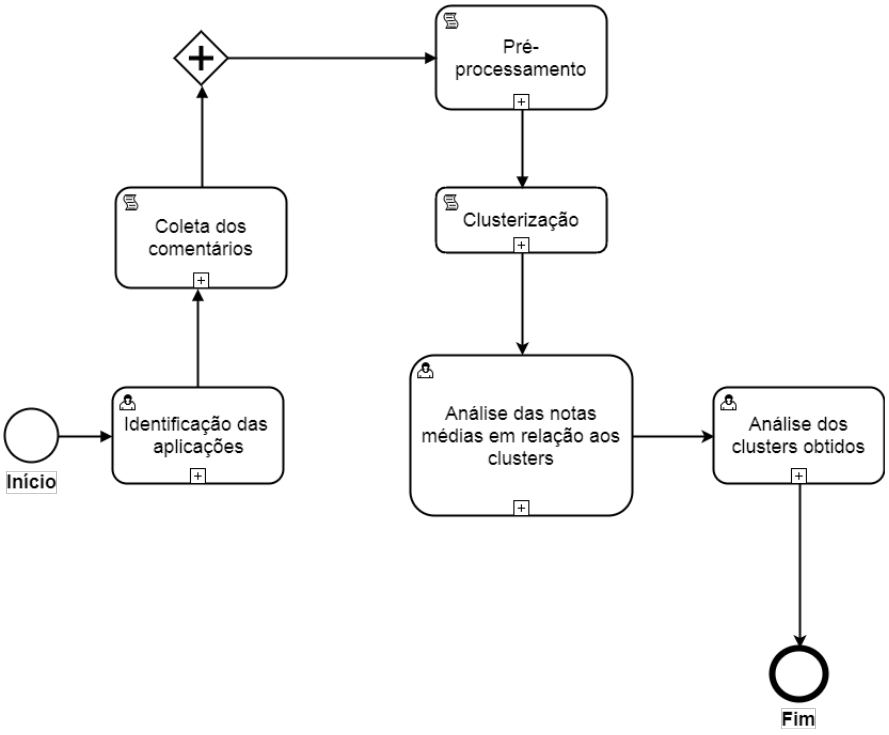
A ferramenta escolhida para este trabalho será a Heedzy, na sua versão *premium*, por ser bem simples e direto de utilizar, exigindo o mínimo de esforço, apesar de não disponibilizar uma API de acesso aos dados. Além disso, as informações que ela fornece sobre os comentários são suficientes para que o estudo possa alcançar seus objetivos. A ferramenta será utilizada exclusivamente para coleta dos comentários.

# 4 PROCEDIMENTOS METODOLÓGICOS

Neste trabalho, serão realizados procedimentos de clusterização de comentários de usuários sobre aplicações móveis, para organizar os tipos de comentários e os tipos de problemas reportados, a fim de permitir a análise do impacto que os tipos de problemas reportados possuem na avaliação das usuários acerca da aplicação.

O fluxo de execução dos procedimentos do trabalho é ilustrado na Figura 4. Inicialmente, foram identificadas as aplicações a serem utilizadas neste estudo. Em seguida, os comentários dessas aplicações foram coletados e pré-processados, preparando-os para as etapas de mineração de dados. Na sequência, o processo de clusterização foi efetivamente executado, para agrupar os comentários baseados na similaridade entre eles.

Figura 4 – Procedimentos metodológicos realizados neste trabalho




Fonte – elaborado pelo autor.

Depois, foi realizada uma análise sobre os grupos de comentários obtidos, para obter uma visão geral do significado dos grupos, e de quais informações eles podem repassar aos desenvolvedores. Por fim, foi realizada uma análise entre as relações entre os grupos e as notas atribuídas à aplicação, a fim de tentar obter uma visão superficial do impacto que os tipos de problemas reportados possuem sobre as avaliações dos usuários.

## 4.1 Identificar aplicações

Inicialmente, foram definidas quais as aplicações móveis serão consideradas neste trabalho. As aplicações a serem coletadas deveriam pertencer às categorias de “Social”, “Jogos” ou “Comunicação”, uma vez que essas comumente recebem mais *feedback* dos usuários (PAGANO; MAALEJ, 2013). Foram consideradas apenas aplicações devidamente publicadas na *Google Play Store*.

Outro critério de seleção dos aplicativos foi sua popularidade, determinada pelo número de *downloads* e comentários que elas possuem na *Google Play Store*. Ao dispor de um número maior de usuários e, consequentemente, de seus comentários sobre a aplicação, é esperado que aplicativos mais populares apresentem um número maior de comentários considerados informativos, que fornecem algum *feedback* sobre as funcionalidades da aplicação que seja de interesse dos desenvolvedores, e ajudem no processo de evolução da aplicação.

Neste estudo, foram consideradas as versões para Android das aplicações do Facebook e do Telegram, por contemplarem os critérios definidos acima, em especial o que diz respeito a sua popularidade.

## 4.2 Coletar comentários das aplicações

Esta etapa teve como objetivo a coleta do conjunto de comentários que serão utilizados ao longo de todo o trabalho. A coleta foi realizada a na *Google Play Store*, por meio da ferramenta *Heedzy*, apresentada na Seção 3.5.

A coleta foi realizada entre Julho e Dezembro de 2017, e ao todo foram coletados 4800 comentários, dos quais metade é referente ao Facebook e a outra metade referente ao Telegram.

O processo de coleta por intermédio do *Heedzy* contemplou as seguintes etapas: primeiramente, escolheu-se os aplicativos cujos comentários seriam coletados, no caso, o Facebook e o Telegram. Para cada aplicativo, houve a opção de acessar os comentários publicados durante o período de uma semana, um mês ou um ano antes da data da coleta, ou então todo o histórico de comentários. Para este estudo, será recuperado todo o histórico de comentários, uma vez que o estudo considerará o número máximo de comentários postados pelos usuários, sem distinção de tempo.

Vale ressaltar que o *Heedzy*, apesar de a ferramenta fazer distinção entre os diferentes

países, no momento da seleção dos aplicativos a serem monitorados, é possível que sejam obtidos comentários escritos em outros idiomas. Entretanto, apenas aqueles que continham palavras em inglês foram considerados.

Como saída, a ferramenta fornece um arquivo *Comma Separated Values* (CSV), contendo os dados dos comentários, que são representados pelos seguintes campos:

- **Origem:** aplicativo ao qual o comentário pertence e de onde foi extraído;
- **Data:** data de publicação (ou última atualização) do comentário;
- **Título:** título do comentário. É comumente algo opcional;
- **Conteúdo:** corpo do comentário. Contém o texto que será analisado para extração de problemas reportados pelos usuários;
- **Nome:** nome do usuário que postou o comentário;
- **Avaliação:** nota atribuída pelo usuário à aplicação. Seus valores variam entre 1 e 5;
- **Versão:** versão da aplicação em que o comentário foi postado;
- **Dispositivo:** modelo de dispositivo do usuário que postou o comentário;

A Figura 5 apresenta uma amostra de comentários coletados através do Heedzy. É possível observar que nenhum dos comentários possui dados sobre Versão ou Dispositivo, e poucos possuem Título. Esse padrão permaneceu para todos os comentários coletados e, por isso, não foi possível realizar qualquer análise que contemplasse tais atributos neste estudo.

Figura 5 – Amostra de comentários coletados utilizando o Heedzy

| 2 | Source | Date | Title | Content | Name | Rating | Version | Device |
|---|---|---|---|---|---|---|---|---|
| 3 | Facebook | 20/07/2017 | | It's ok | | 4 | | |
| 4 | Facebook | 20/07/2017 | | Thanks | Thandar Win | 5 | | |
| 5 | Facebook | 20/07/2017 | Amazing | Facebook | Reynato Solis | 3 | | |
| 6 | Facebook | 20/07/2017 | | Yes | john Jay games | 5 | | |
| 7 | Facebook | 20/07/2017 | | Loved it!ðŸ˜™ðŸ˜™ | Kiran Karashi | 3 | | |
| 8 | Facebook | 20/07/2017 | | Ok | Anh Nguyen | 5 | | |
| 9 | Facebook | 20/07/2017 | | good | | 5 | | |
| 10 | Facebook | 20/07/2017 | | Thanks | Bkt Service | 4 | | |
| 11 | Facebook | 20/07/2017 | | help connecting friends and rela | Anne Flores | 5 | | |
| 12 | Facebook | 20/07/2017 | | It has booked me from the marke | Paul Mayled | 1 | | |
| 13 | Facebook | 20/07/2017 | | Ye acha hai | ethelical expert hacking 123 | 5 | | |
| 14 | Facebook | 20/07/2017 | | Was Good facebook keeps updat | bee Jae Powell | 2 | | |
| 15 | Facebook | 20/07/2017 | | I think you suck | James Cunningham | 1 | | |
| 16 | Facebook | 20/07/2017 | | I like you | Mslove Mewada | 5 | | |
| 17 | Facebook | 20/07/2017 | | Cool Favourite | | 5 | | |
| 18 | Facebook | 20/07/2017 | | Great for contacting with friends | Quentin Henderson | 4 | | |
| 19 | Facebook | 20/07/2017 | | I loved it | | 5 | | |
| 20 | Facebook | 20/07/2017 | | videshi kumar | Baljinder Singh | 1 | | |
| 21 | Facebook | 20/07/2017 | | trsh | X 4 N T Y å¯ŸæŸ»ã€, | 5 | | |
| 22 | Facebook | 20/07/2017 | | Love it | | 5 | | |
| 23 | Facebook | 20/07/2017 | | Very nice | | 5 | | |

Fonte – <www.heedzy.com>

Todos os comentários coletados e utilizados neste estudo são podem ser acessados através da pasta compartilhada[1].

## 4.3 Realizar o pré-processamento do conteúdo dos comentários

Como apresentado na Seção 3.5, os comentários coletados possuem vários atributos. Contudo, nem todos eles são de interesse. Para o presente trabalho, os atributos de interesse são: conteúdo, que é o corpo textual do comentário; e a nota atribuída. Os demais atributos não são relevantes para a análise dos problemas que os usuários reportam, e serão desconsiderados. Com isso, haverá uma dimensionalidade dos comentários, uma vez que apenas os campos importantes serão utilizados, o que diminui a complexidade e o tempo necessário para execução dos processos de mineração.

Para realizar o pré-processamento dos comentários coletados, foi utilizado o NLTK (*Natural Language Toolkit*)[2], um conjunto de bibliotecas escrito em Python e cujo código se encontra disponível em um repositório no GitHub[3]. Ele oferece algoritmos para o pré-processamento de textos em diversos idiomas, oferecendo funções para identificação de palavras que pertencem a um idioma específico, remoção de *stopwords*, tokenização e lematização, operações básicas para limpar os dados e prepará-los para os processos de mineração, entre outras.

Foi construída uma aplicação também em Python que utiliza o NLTK para pré-processar os dados, que define as funções necessárias para a realização desse processo. No escopo deste trabalho, foram considerados apenas comentários escritos em inglês. O código-fonte do projeto pode ser encontrado no seu repositório do GitHub[4].

Depois de pré-processados, houveram comentários que acabaram ficando vazios e foram, portanto, descartados. A Tabela 5 apresenta a quantidade de comentários restantes para cada aplicação.

## 4.4 Clusterizar os comentários

Em seguida, foi executado o processo de clusterização. Para clusterizar os comentários, foi utilizado a ferramenta minetext, desenvolvido em Python e cujo código se

[1] <https://drive.google.com/drive/folders/18BKWUHLuEUy0HWE5_Jhsz8_TIrE3sJ23?usp=sharing>
[2] <http://www.nltk.org/>
[3] <https://github.com/nltk/nltk>
[4] <https://github.com/CaioMelo8/android-comments-miner>

Tabela 5 – Número de comentários restantes após o pré-processamento.

| Aplicação | Total de Comentários | Comentários Restantes |
|---|---|---|
| Facebook | 2400 | 2123 |
| Telegram | 2400 | 1866 |

Fonte – Elaborado pelo autor.

encontra disponível no GitHub[5]. Ela implementa várias técnicas de mineração de textos, dentre elas a clusterização. Ela possui suporte a diversos algoritmos de clusterização, como o DBSCAN e *K-means*/*K-medoids*, além de implementar o cálculo de várias medidas de distância, entre elas o coeficiente de Jaccard, a função de Fading e a distância Euclidiana.

Neste trabalho, o algoritmo selecionado para clusterizar os comentários será o *K-medoids*, que é apresentado na subseção 3.4.1. Como já foi dito anteriormente, ele agrupa textos baseados na distância entre eles. Dados dois comentários quanto maior a distância, menor a similaridade, e vice-versa.

A distância entre dois textos pode ser definida de diversas formas distintas. Há diversos algoritmos para cálculo de distância entre textos, a exemplo da distância Euclidiana e de Fading. No contexto deste trabalho, foram consideradas dois tipos de distância: a distância de Jaccard e de Levenshtein, que são explanadas nas Seções 3.4.2.1 e 3.4.2.2, respectivamente. Para cada medida de distância, foi executado um processo separado de clusterização. O objetivo em realizar execuções distintas é comparar os agrupamentos de comentários obtidos para ambas as medidas, a fim de analisar que tipos de resultados eles fornecem e buscar identificar qual delas é mais adequada para clusterização dos comentários.

Os resultados do *K-medoids* podem variar de uma execução para outra, por mais que o valor do número $K$ de *clusters* iniciais seja o mesmo. Logo, a qualidade da clusterização também varia a cada execução. Levando isso em consideração, para cada valor de $K$, a clusterização foi executada em um número $i$ de iterações, e a iteração que apresentou melhor qualidade dos *clusters* foi considerada.

Assim, a clusterização foi executada para diversos valores $K$ de grupos iniciais e, para cada um deles, foi considerada a melhor dentre as $i$ iterações executadas. Por motivos de tempo de execução e obtenção dos resultados, cada medida de distância considerou intervalos diferentes para $K$ e $i$. Os intervalos são apresentados na Tabela 6.

Após a clusterização, a aplicação salvou os resultados em um conjunto de arquivos

[5] <https://github.com/gustavoaires/minetext>

Tabela 6 – Valores para o número de *clusters* ($K$) e de iterações ($i$).

| **Distância** | **Valor mínimo de $K$** | **Valor máximo de $K$** | **Número de iterações** |
|---|---|---|---|
| Jaccard | 3 | 80 | 40 |
| Levenshtein | 3 | 40 | 20 |

Fonte – Elaborado pelo autor.

CSV, e eles foram divididos tanto por valor de $K$ quanto pelo valor de $i$, onde *cluster* dá origem a um arquivo CSV. Ainda, para cada *cluster*, para facilitar o processo de análise do conteúdo seu conteúdo, foram identificados os 100 comentários mais similares ao *medoid*, uma vez que, por estarem mais próximos ao *medoid*, é natural pensar que eles tratam sobre assuntos ou contém termos semelhantes e, portanto, conseguem caracterizar ou descrever o grupo ao qual pertencem.

Em seguida, seguindo a estrutura descrita na subseção 3.4.1, uma heurística foi executada para identificar o valor ou os valores de $K$ que indicam estabilidade na relação entre o número de *clusters* e o SSE. Devido à forma como é dado o cálculo das distâncias de Jaccard e Levenshtein, os seus resultados são obtidos em escalas distintas. Assim, foi necessário normalizar os valores para uma única escala, que varia entre 0 e 1.

No contexto deste trabalho, a heurística buscou por um intervalo de 5 valores de $K$ consecutivos cuja diferença entre os SSEs entre cada valor fosse menor ou igual a 7.5. O tamanho do intervalo considerado foi definido a partir do tamanho do conjunto de resultados para as duas medidas de distância; já o valor do aumento no SSE foi obtido pela própria heurística, que buscava pela menor diferença que retornasse um intervalo válido tanto para as distâncias e quanto para as aplicações. O valor de $K$ considerado para análise foi aquele que apresentou o menor SSE dentre os valores pertencentes ao intervalo obtido.

Depois de definir os valores de $K$ a serem analisados, foram calculadas as estatísticas para cada um dos $K$ *clusters* gerados que são: número de documentos; média das notas; e distribuição de frequência das notas. A partir delas, gerou-se gráficos para visualização de certas características dos grupos, por meio da biblioteca Matplotlib[6]. Tais gráficos complementaram o processo de a análise dos *clusters*, descrito nas próximas seções.

[6] <https://github.com/matplotlib/matplotlib>

## 4.5 Analisar os *clusters* gerados

Após obter os resultados da clusterização, foi realizada uma análise dos *clusters* obtidos, na qual foi observado o conteúdo dos comentários pertencentes a eles. O objetivo principal da análise é buscar descrever cada *cluster* a partir do conjunto de comentários que os compõem, para identificar sobre o que eles tratam, com foco especial na reportagem de problemas relacionados às aplicações.

Como dito na Seção 3.5, para cada aplicação, foram considerados mais de 200 comentários. Realizar um processo de análise manual de todos os comentários seria relativamente custoso. Dessa forma, para otimizar a análise dos *clusters*, foram considerados somente os 100 comentários mais similares (ou menos distantes) ao medoid, devido à sua capacidade de descrever sucintamente o conteúdo do *cluster*.

## 4.6 Analisar a relação entre os *clusters* das avaliações dos usuários

A nota do usuário é um dos atributos mais importantes e que normalmente representam bem a sua opinião a respeito da aplicação. É natural pensar que comentários que falam de problemas e contém críticas ou opiniões negativas possuam notas mais baixas; em contraste, é de se esperar que comentários que contenham elogios ou opiniões positivas apresentem notas mais altas. Para averiguar se isso se mostra verdade no conjunto de comentários considerados, foi feita uma análise da relação entre o conteúdo de um *cluster* e como ele pode ser descrito, e as notas atreladas aos comentários que o pertencem.

Anteriormente, falou-se sobre a construção dos gráficos que apresentavam as médias das notas de cada *cluster* e suas distribuições de frequência. Nesta etapa, a partir dos grupos gerados, comparou-se as estatísticas observadas e a descrição de cada grupo, a fim de verificar se a relação entre ambos realmente se comporta da forma descrita acima.

# 5 RESULTADOS

Neste capítulo são apresentados os principais resultados obtidos após a execução dos passos planejados, descritos no Capítulo 2. Entre eles, estão o produto da clusterização dos comentários, as estatísticas obtidas a partir dos grupos gerados e o que foi descoberto através da análise de ambos.

## 5.1 Descrição dos clusters gerados

Aqui, serão apresentados os resultados referentes à clusterização e à análise dos grupos gerados e como eles podem ser descritos. Na subseção 5.1.1 são descritos os resultados para a execução que considerou a distância de Jaccard, para os aplicativos do Facebook e do Telegram. Na sequência, a subseção 5.1.2 apresenta os resultados para a execução que utilizou a distância de Levenshtein, também para ambas as aplicações.

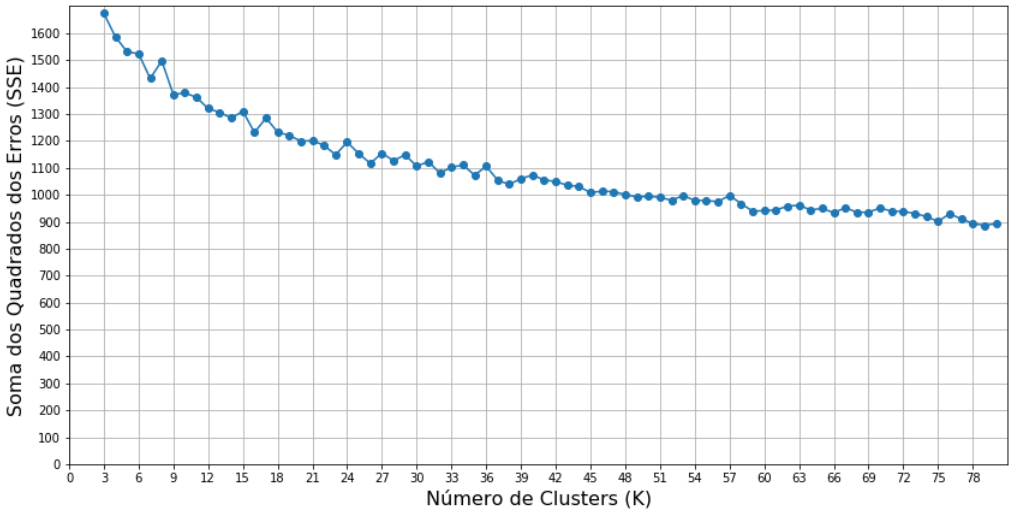
### *5.1.1 Distância de Jaccard*

A relação entre o número de clusters e o SSE obtido para ambos os aplicativos pode ser observada nas Figuras 6 e 7. Nos gráficos, é possível observar que a linha do gráfico é decrescente e que a variação entre os pontos segue padrões bastante semelhantes. Nota-se também como a diferença no SSE entre os pontos diminui à medida que os valores de $K$ aumentam. O valor de $K$ no qual essa estabilização na variação do gráfico começa a ser percebida é um valor bom para o número de *clusters*, segundo a heurística definida.

O limite máximo para o eixo de valores de SSE difere nos dois gráficos, pois reflete o maior valor obtido dentre todos os resultados. De forma geral, o Facebook apresentou SSEs maiores que o Telegram, de modo que a diferença entre eles flutua entre 200 e 500, para todos os valores de K. Isso poderia indicar que os grupos de comentários coletados acerca do Facebook são menos coesos que aqueles sobre o Telegram.

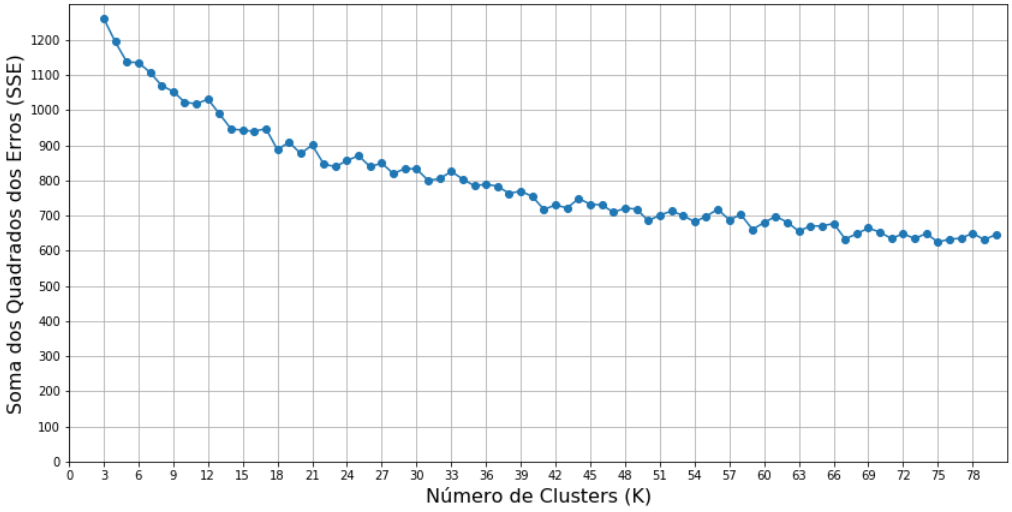
Entretanto, é preciso ressaltar que, devido à etapa de pré-processamento, o número de comentários clusterizados para cada aplicação foi diferente: o Facebook permaneceu com mais comentários válidos que o Telegram. O número de comentários considerados pode afetar diretamente o valor do SSE absoluto, que não se preocupa com a quantidade de documentos processados. Logo, não é possível afirmar com confiança que os comentários de uma aplicação são mais ou menos coesos que o da outra.

Figura 6 – Relação entre o número de *clusters* gerados pela distância de Jaccard e o SSE obtido para os comentários do Facebook.




Fonte – elaborado pelo autor.

Figura 7 – Relação entre o número de *clusters* gerados pela distância de Jaccard e o SSE obtido para os comentários do Telegram.



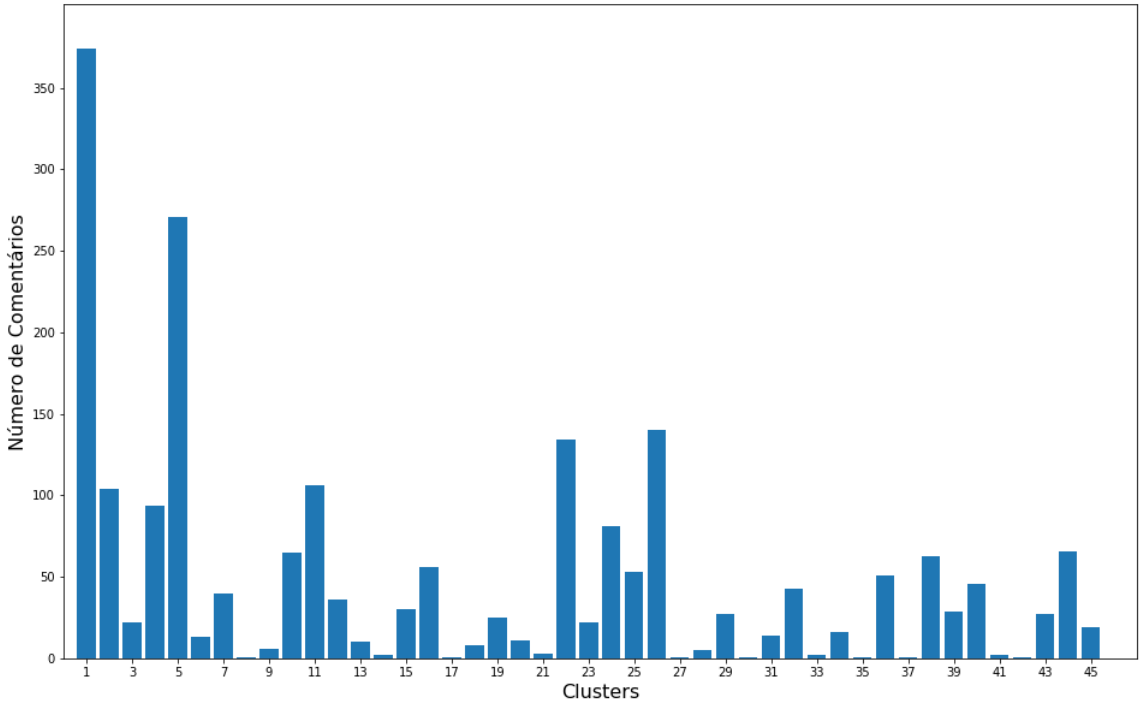

Fonte – elaborado pelo autor.

O melhor valor de $K$ pode variar entre as aplicações, uma vez que os comentários publicados sobre elas podem tratar de assuntos distintos, por exemplo, um aplicativo apesentar mais problemas ou possibilidades de melhorias que outro; ser mais criticado ou elogiado; e assim por diante. Nas seguintes subseções são apresentados os resultados das análises dos *clusters* para ambos os aplicativos.

#### *5.1.1.1 Facebook*

Considerando os comentários sobre o Facebook, a heurística determinou que $K = 45$ foi o melhor valor para o número de *clusters*. A Figura 8 mostra a distribuição do número de comentários entre os grupos gerados.

Figura 8 – Distribuição dos comentários do Facebook entre os *clusters* obtido pela distância de Jaccard




Fonte – elaborado pelo autor.

Observando a distribuição, é fácil notar que o *cluster* 1 é o maior, e que poucos *clusters* possuem uma quantidade relativamente grande de comentários, se comparada ao total considerado; em contrapartida, há vários grupos compostos por um número pequeno de comentários. É fácil notar que os *clusters* 1 e 5 são extremamente mais volumosos que os demais.

No que diz respeito ao conteúdo dos clusters, a análise dos 100 comentários mais similares ao *medoid* mostrou que foram gerados grupos com comentários majoritariamente positivos, em detrimento de outros que apresentavam conteúdo que relatava problemas e solicitava mudanças.

Entre os clusters com comentários que não apresentaram comentários considerados

informativos, estão:

- **Cluster 5:** Comentários em sua maioria concordam que a aplicação é boa (*good*), mas há alguns que afirmam o contrário.
- **Cluster 9:** Pouquíssimos comentários, e todos falam que a aplicação era boa (*good*), mas com a palavra soletrada erroneamente (*gud*).
- **Cluster 11:** *Cluster* relativamente grande, com comentários que declaram amar (*love*) a aplicação, além de distribuir adjetivos: legal (*nice*), incrível (*awesome*), adorável (*lovely*), entre outros.
- **Cluster 14:** Apenas 2 comentários, que alegam ter gostado (*enjoy*) de usar aplicação.
- **Cluster 26**: Grupo com muitos comentários, e afirmam que o aplicativo é legal (*nice*).

Já entre os agrupamentos que detém comentários que apresentam informações sobre problemas e melhorias, tem-se:

- **Cluster 1:** Contém a maior quantidade de comentários, e cuja esmagadora maioria pode ser considerada um *outlier*, que não apresenta nenhuma similaridade com o *medoid* (distancia igual a 1), mesmo entre os comentários mais similares. Apesar disso, os comentários mais semelhantes ao *medoid* são os que exprimem positividade, gratidão (*thanks*) e elogiam o aplicativo (*awesome*).
- **Cluster 2:** Diversos comentários que alegam gostar (*like*) do aplicativo, mas há uns que afirmam o oposto. Apresentam comentários sobre a problemas com função "Curtir". Alguns discorrem sobre outros tipos de problemas.
- **Cluster 3:** Grupo com poucos comentários, em maior parte, falam sobre ao ato de pensar (*think*). Contém elogios e críticas, e alguns comentários reclamam sobre a quantidade de permissões, aplicação fechando automaticamente e outros.
- **Cluster 4:** Quantidade razoável de comentários, que falam sobre problemas de travamento, congelamento, parada e reinicialização forçada e lentidão da aplicação. Foco especial em problemas ao visualizar vídeos. Alguns comentários sugerem novas funcionalidades, como *download* de vídeos e busca em grupos.
- **Cluster 6:** Poucos comentários, e quase todos tratam sobre mudanças (*change*). Alguns afirmam problemas ao mudar a senha, informações e configurações da conta. Alguns reclamam sobre permissões para mudar configurações do sistema e controle de acesso remoto.
- **Cluster 7:** Grupo pequeno cuja maioria dos comentários reclamam sobre problemas após

atualizações da aplicação, falando também sobre o volume excessivo de atualizações. Há ainda relatos sobre outros problemas, não diretamente ligados às atualizações, como lentidão, propagandas excessivas e erros em certas funcionalidades.

- **Cluster 10:** Grupo com número razoável de comentários que, em sua grande maioria reclamam sobre travamento repentino (*crash*) e congelamento (*freeze*) do aplicativo, mas diversas funcionalidades diferentes. Alguns reportam lentidão no carregamento, e outros se queixam de propagandas excessivas.
- **Cluster 12:** *Cluster* pequeno, em que a maioria dos comentários trata sobre problemas decorridos da atualização (*update*) da aplicação. São relatados também outros problemas não diretamente relacionados às atualizações.
- **Cluster 13:** Número muito pequeno de comentários, cujo tema em comum é falar sobre nada (*nothing*). Alguns reclamam de problemas que persistem após o tempo o após realizar ações como desinstalar o aplicativo, alegando que nada acontece
- **Cluster 15:** Grupo com número moderado de comentários majoritariamente positivos, que chamam a aplicação de excelente/formidável (*super*). Todavia contém alguns poucos comentários falando de travamentos e enfatizando na lentidão (*super slow*).
- **Cluster 16:** O *cluster* apresenta uma número considerável de comentários. A maioria deles reporta diversos tipos de problemas: travamento (*crash*), congelamento (*freeze*), entre outros. Alguns usuários atribuem o surgimento dos problemas às atualizações. Alguns citam funcionalidades específicas da aplicação.
- **Cluster 18:** Grupo com poucos comentários, dos quais a maioria se queixa de drenagem excessiva da bateria pela aplicação. Uma minoria reclama de problemas de travamento (*crash*).
- **Cluster 19:** *Cluster* com poucos comentários. Boa parte deles trata sobre o aplicativo estar melhor que antes, que era melhor antes ou que poderia melhorar. Há alguns poucos que reportam problemas, e reclamam do tamanho e da aplicação ser lenta (*slow*).
- **Cluster 20**: Grupo apresenta um pequeno número de comentários. Todos eles expressam a necessidade (*need*) de melhorias, algumas novas funcionalidades (tal como tradutor) e de conserto de *bugs* e problemas encontrados na aplicação. Certos comentários reclamam especificamente da quantidade de memória (RAM e armazenamento) utilizada pelo aplicativo.
- **Cluster 22:** *Cluster* grande, cujos comentários explicitam o aplicativo (*app*), mas tratam

de assuntos diversos: legal (*nice*); o melhor (*best*); alguns relatam problemas como travamento (*crash*) e congelamento (*freeze*).

- **Cluster 23:** Grupo com poucos comentários. Parte deles reporta problemas de diversos tipos. É notável a presença das palavras tempo/vezes (*time/times*) nos conteúdo dos comentários.
- **Cluster 24:** Esse grupo apresenta uma alta quantidade de comentários, em que boa parte discorre sobre usar (*use*) a aplicação, o quão útil (*useful*), incrível (*awesome*) e fácil (*easy*) de usar ela é. Há também comentários que reclamam do alto uso de memória (*memory*), experiência de usuário, entre outros problemas.
- **Cluster 25**: Todos os comentários tratam sobre o quão incrível (*awesome*) é o aplicativo. Apenas um deles relata um problema de lentidão (*slow*), ainda que ache aplicação incrível.
- **Cluster 28:** Pouquíssimos comentários, entretanto todos eles falam especificamente de problemas ao executar (*play*) vídeos, seja por que não funciona ou por que executam automaticamente.
- **Cluster 29:** Quantidade pequena de comentários, onde os usuários discorrem sobre a importância do Facebook para se comunicar com família (*family*) e amigos (*friends*). A maioria dos comentários elogia o aplicativo, mas há uns poucos que citam problemas, como falha no carregamento (*load*) e do número excessivo de inserção de senha (*password*).
- **Cluster 31:** O grupo reúne um número pequeno de comentários. A maior parte deles exprime a ideia de que o aplicativo como um todo, a sua versão atual ou algumas das funcionalidades são ruins (*bad*); já uma minoria indica que a aplicação não é nada mal (*not bad*).
- **Cluster 32:** Contém um número razoável de comentários, e quase todos falam o quanto a aplicação é *OK*. Todavia, há um comentário específico que contém o termo *OK*, mas reclama do volume de uso de dados.
- **Cluster 34:** Apresenta uma baixa quantidade de comentários, que falam sobre diversos assuntos: elogios, críticas e problemas relacionados a vídeos e fotos. Todos os comentários ou citam os termos *Facebook* ou livro *book* (e derivados).
- **Cluster 36:** *Cluster* que contém um número moderado de comentários, cujo conteúdo expressa que a aplicação é a melhor (*best*), ou que versões anteriores eram melhores.
- **Cluster 38:** Contém uma quantidade razoável de comentários. A maioria deles afirma que o aplicativo é ótimo (*great*). Uns poucos comentários reportam problemas na interface

e experiência do usuário, além de reclamar sobre travamentos (*crash*) e problemas ao visualizar vídeos ao vivo (*live*).

- **Cluster 39:** O grupo apresenta poucos comentários, dos quais a maior parte relata problemas após atualizações (*update*). Dentre os problemas, está parada forçada (*stop*), travamento (*crash*), lentidão (*slow*), congelamento (*freeze*) e carregamento (*load*). Outros reclamam sobre problemas na interface, ausência de som e impossibilidade de uso de algumas funcionalidades (comentar e usar *emojis*).
- **Cluster 40:** Apresenta uma quantidade considerável de comentários. A maioria deles fala sobre o quão legal (*cool*) é a aplicação. Apenas um deles afirma o contrário, ao reclamar da execução automática de vídeos com áudio ao rolar pelo *feed* do Facebook.
- **Cluster 43:** *Cluster* que apresenta um volume moderado de comentários, em que todos discorrem sobre atualização (*update*). Em grande parte, eles se queixam do tamanho de cada atualização e da memória ocupada pela aplicação (em termos de armazenamento), mas que não veem mudanças significativas a cada nova versão. Há aqueles que afirmam que a atualização trouxe problemas, como impossibilidade de entrar na aplicação (*login*), lentidão e presença de *bugs*.
- **Cluster 44:**: Outro grupo que contém diversos comentários relacionados à atualizações (*update*). Usuários reclamam da atualização, afirmando que ela trouxe *bugs*, que fez o aplicativo ficar mais lento (*slow*) e travar (*crash*) ou congelar (*freeze*) mais constantemente. Outros reclamam de mudanças na interface ou em funcionalidades específicas do Facebook.
- **Cluster 45:** Conta com poucos comentários, cujo conteúdo expressa amor (*love*) ao aplicativo. Entretanto, alguns poucos reclamam de problemas relacionados a travamentos (*crash*) e impossibilidade de entrar na conta do Facebook.

Há *clusters* que não foram descritos, ou por apresentarem comentários ilegíveis (escritos em outro idioma) ou conter apenas o *medoid* como único elemento do grupo.

Nota-se que a maioria dos grupos continha algum comentário que relatava problemas ou sugeria melhorias, em face de poucos grupos que não apresentaram comentários desse tupo. Em vários dos grupos obtidos, os comentários falam sobre problemas diversos, geralmente sobre assuntos distintos, como problemas após atualização, falha na execução de vídeos, mudanças nas funcionalidades ou na interface do aplicativo, entre inúmeros outros. Na realidade, quase nenhum *cluster* conseguiu reunir comentários que apontavam para os mesmos problemas, embora o tema geral do cluster fosse respeitado.

Esses resultados permitem inferir que o Jaccard consegue agrupar comentários que tratam de problemas e melhorias relacionadas, mas que o conteúdo de parte dos grupos obtidos não se limita exclusivamente a tais assuntos. Entre os problemas mais citados, estão travamentos, congelamentos da aplicação e reclamações sobre a frequência e tamanho das atualizações.

Como mencionado anteriormente, a distribuição dos comentários evidencia que os *clusters* 1 e 5 são os maiores. No caso do *cluster* 1, a maior parte dos seus comentários não possui similaridade alguma entre si, podendo ser considerados outliers. Uma explicação para a sua concentração nesse grupo é que, caso eles sejam totalmente dissimilares a todos os *medoids*, eles serão alocados no primeiro agrupamento.

Já em relação ao *cluster* 5, todos os comentários tratam sobre a aplicação ser ou não boa, sendo que termo bom/boa (*good*) é especialmente citado em todos eles. Ambas são palavras bastante comuns e utilizadas em diversos contextos, com sentidos diversos, embora geralmente denotem aprovação de algo. Assim, é possível teorizar que o grande número de comentários do grupo é causa da presença de um termo bastante comum nos comentários que o compõem.

#### *5.1.1.2 Telegram*

No caso do Telegram, o melhor valor encontrado para o número de *clusters* foi $K = 47$. Na Figura 9 é possível visualizar a distribuição dos comentários entre os *clusters* gerados.

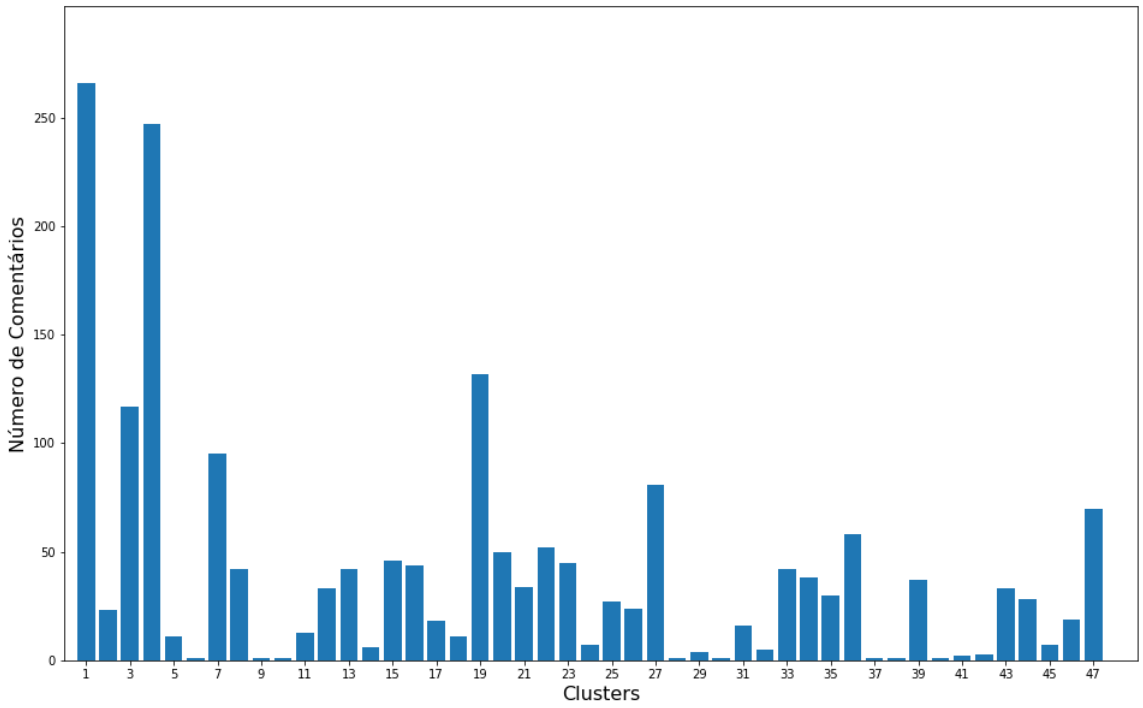
Analisando o gráfico e comparando-o com a Figura 9, é possível observar que distribuição dos comentários entre os diferentes grupos é similar: poucos grupos contém um número grande de comentários, enquanto muitos grupos contém poucos comentários. Novamente, o *cluster* 1 é o mais volumoso, mas dessa vez ele é seguido pelos *clusters* 3 e 19.

Para cada grupo, foi feita uma análise dos 100 comentários mais similares ao *medoid*, a fim de definir uma descrição geral do *cluster*.

Considerando os grupos em que não foram identificados comentários informativos, viu-se que:

- **Cluster 2:** *Cluster* com poucos comentários, em que todos afirmam que o Telegram é excelente (*excellent)*.
- **Cluster 4:** Apresenta muitos comentários, que se referem à aplicação como boa (*good*).
- **Cluster 19:** *Cluster* com muitos comentários, em que todos afirmam que o aplicativo é legal (*nice*).

Figura 9 – Distribuição dos comentários sobre o Telegram entre os *clusters* obtidos pela distância de Jaccard




Fonte – elaborado pelo autor.

- **Cluster 21:** Grupo de tamanho moderado, em que todos os comentários expressam aprovação (*OK*).
- **Cluster 29:** Pouquíssimos comentários, que veem o aplicativo como o notável (*greatest*).

Em contrapartida, entre os *clusters* cujos comentários apresentam informações úteis aos desenvolvedores, observou-se:

- **Cluster 1:** Grupo muito grande. Os comentários mais similares ao *medoid* tratam sobre o uso/utilidade (*use/useful*) da aplicação, em que a maioria afirma que ele é fácil de usar e muito útil. Entretanto, há comentários que tratam de diversos problemas encontrados durante o uso, como falhas (*glitch*) e travamentos (*crash*). Há alguns que sugerem funcionalidades, por exemplo, bloquear usuários específicos nos *chats* em grupo. Há a presença de um número grande de *outliers*, que não possuem similaridade alguma com o *medoid* (distância igual a 1).
- **Cluster 3:** Grupo com quantidade considerável de comentários, cuja maioria elogia a aplicação, a definindo como legal(*nice*), formidável (*super*), fantástico (*fantastic*) esplêndido (*superb*). Há vários comentários que falam que a origem do aplicativo ser Indiana. Todavia, há uns poucos comentários que reclamam de problemas de conexão

(*connection*) e a necessidade de mudanças, como adição de chamadas (*calls*. Todos os comentários ressaltam a aplicação (*app*).

- **Cluster 5:** *Cluster* com poucos comentários, em que parte deles afirmam que o Telegram é ruim (*bad*), e outra parte discorda dessa afirmação. Desses últimos, alguns relatam problemas com mensagens do *chat* e a função de chamada (*call*).
- **Cluster 7:** Grupo com uma grande quantidade de comentários. Quase todos categorizam o aplicativo como bom (*good*), seguro (*secure*) e alguns até afirmam que o amam (*love*). Contudo, há múltiplos usuários que solicitam chamadas de vídeo (*video calls*). Outros se queixam de problemas relacionados à consumo de bateria (*draining battery*), super aquecimento (*heating*) do aparelho, falha na conexão e no *log in*, e erro ao exibir notificações (*notifications*).
- **Cluster 8:** Contém um número moderado de comentários, em que todos expressão o quão melhor (*better*) o Telegram é especificamente em relação ao seu principal concorrente, o Whatsapp. Usuários citam funcionalidades que consideram superiores, como interface de usuário e sincronização, enquanto outros sugerem novas funcionalidades como chamadas e organização dos *chats*.
- **Cluster 11:** Grupo com poucos comentários, em que todos os comentários dizem que o Telegram é incrível (*amazing*). Todavia, alguns deles solicitam adições, tais como mais idiomas (*language*), ou reclamam do desempenho do aplicativo.
- **Cluster 12:** *Cluster* que contém poucos comentários, cuja maioria expressa gratidão (*thanks*), e distribui elogios: útil (*helpful*), poderoso (*powerful*), o melhor (*best*), perfeito (*perfect*), fácil de usar, entre outros. Há usuários que, mesmo elogiando, pedem novas funcionalidades, como mais idiomas (língua persa) e sugerem separação na organização dos *chats* de grupos, super grupos e canais. Há alguns que falam sobre problemas de conexão e envio de mensagens.
- **Cluster 13:** Grupo com número razoável de comentários, que em maior parte dizem que o aplicativo é formidável (*super*). Há apenas um deles que pede novas funcionalidades, como silenciar mensagens de grupos; outro reclama de problemas de conectividade e não recebimento de notificações.
- **Cluster 14:** Apresenta pouquíssimos comentários. Todavia, quase todos falam sobre dar ou não 5 estrelas à aplicação: a maioria afirma que daria 5 estrelas caso algo mudasse na aplicação; há um comentários que afirma não atribuir nota 5 por que os desenvolvedores

não aceitaram contribuições no projeto, que é *open-source*.

- **Cluster 15:** Grupo de tamanho moderado, onde os comentários se queixam de diversos problemas, tais como falha ao abrir e se conectar à aplicação, lentidão ao abrir certas seções e *bugs* na interface. Há alguns que solicitam uma versão do aplicativo para cada cartão SIM, em dispositivos *dual* SIM.
- **Cluster 16:** Grupo composto por um número médio de comentários, no qual todos se referem à aplicação como incrível (*awesome*). Alguns elogiam características como privacidade (*privacy*), simplificação (*simplify*) e por ser grátis (*free*). Contudo, há uns poucos comentários que solicitam a emissão de notificações quando capturas de tela (*screenshot*) forem realizadas, enquanto outros se queixam sobre problemas de conexão.
- **Cluster 17:** Grupo de comentários pequeno, onde todos possuem conteúdo relacionado ao *chat* da aplicação: alguns elogiam tal funcionalidade, mas a maior parte reporta problemas: alguns usuários alegam não conseguir receber/enviar mensagens, e que as notificações (*notifications*) de mensagens recebidas não estão sendo exibidas.
- **Cluster 18:** Apresenta um volume muito pequeno de comentários. Boa parte deles elogia diversos aspectos da aplicação. Os comentários são bastante extensos e contém um texto bem elaborado. Há, porém, alguns que reportam problemas de travamento (*crash*), lentidão (*lag*) ao realizar video chamadas, demora na conexão e problemas ao realizar *log-in*.
- **Cluster 20:** Contém uma quantidade razoável de comentários, dos quais a maior parte diz gostar (*like*) do aplicativo ou compará-lo (*like*) a outros aplicativos (e.g. Whatsapp). Desses, alguns elogiam principalmente a boa segurança (*security*) da aplicação. Alguns usuários solicitam funcionalidades, como uso de *widgets* ou *backup* local. Outros reclamam de problemas nas ligações (*calls*) ou consumo excessivo de bateria (*drains battery*).
- **Cluster 22:** *Cluster* de tamanho médio que apresenta muitos comentários se referindo ao aplicativo como legal (*nice app*), dos quais alguns o comparam com concorrentes (Whatsapp, Facebook Messenger), o definindo como mais rápido (*faster*) e melhor (*better*). Há alguns usuários que reclamam que a aplicação não funciona (*not working*), não é útil (*not useful*) ou cometeu banimento arbitrário de número.
- **Cluster 23:** O grupo detém um volume razoável de comentários, cuja grande maioria se relaciona a vídeos. Alguns usuários solicitam que sejam adicionadas vídeo chamadas (*video calls*), e que os dados do vídeo sejam baixados em paralelo a sua execução; outros reclamam de problemas de conexão. São feitas também outras solicitações relacionadas à

interface das mensagens e o uso de *stickers/emojis*.

- **Cluster 24:** Contém muito poucos comentários, que enfatizam o termo mensageiro (*messenger*). Alguns o elogiam, dizendo que ele é bom (*good*), completo (*complete*) e melhor (*better*) que o Whatsapp. Entretanto, há um comentário que vê o aplicativo como não-seguro (*unsecured*)
- **Cluster 25:** Apresenta uma quantidade razoável de comentários. Dentre eles, há vários que reclamam sobre problemas nas notificações (*notifications*). Outros se queixam de que o Telegram não funciona em seu dispositivo, ou que estão com problemas de conexão. Alguns comentários solicitam funcionalidades para o *chat*, como deletar mensagens antigas nos grupos.
- **Cluster 26:** *Cluster* de tamanho médio, cuja maioria dos comentários trata sobre adição (*add*) de funcionalidades e características ao aplicativo. A solicitação mais comum é a adição de outros idiomas, em especial a língua Persa. Outros comentários pedem adição de *bookmarks*, possibilitar adicionar contatos aos favoritos (*favorites*), permitir deletar mensagens específicas e proibir capturas de tela (*screenshots*).
- **Cluster 27:** Grupo apresenta um número grande de comentários, dos quais a maioria concorda que o Telegram é o melhor (*best*) ou melhor que os concorrentes. Mas há alguns comentários que se queixam do aplicativo não mudar o *status* para *offline* ao sair, problemas de *log-in*, dificuldade em encontrar grupos e presença de *spam*.
- **Cluster 31:** O agrupamento contém poucos comentários, dos quais parte discorre sobre a função de envio de arquivos/imagens: alguns comentários identificam problemas, como o teclado não ser minimizado durante o envio de fotos. o envio ser lento ou não funcionar; outros aprovam a possibilidade de enviar arquivos grandes. Há ainda comentários que fazem elogios ao aplicativo,
- **Cluster 32:** *Cluster* com poucos comentários, que apresentam a presença da palavra sim (*yes*). Um dos comentários reclama de não conseguir conectar (*connect*) à aplicação.
- **Cluster 33:** Contém um volume razoável de comentários. Uma boa porção deles requisita novas funções, tais como *backup offline*, chamadas de vídeo (*video calls*), destaque (*bookmark*) de mensagens importantes, categorização de *chats*. Alguns usuários sugeriram mudanças nas funções de encaminhamento de mensagens, *download* de arquivos e execução de vídeo. Alguns reportaram lentidão (*slow*) ao baixar o aplicativo, travamento (*crash*) e falha na conexão.

- **Cluster 34:** *Cluster* composto por um número moderado de comentários. Enquanto uma parcela dos usuários se mostra satisfeita com as funcionalidades adicionadas anteriormente, outros sugerem a adição de funcionalidades, como chamadas de vídeo (*video calls*), autenticação em 2 passos (*2-step authentication*), exibição de progresso durante *downloads*, compartilhamento recursivo de pastas, dentre várias outras. Há reportagem de alguns problemas de conectividade (*connection*) e lentidão (*slow*).
- **Cluster 35:** O grupo apresenta quantidade razoável de comentários. Todos eles elogiam o aplicativo, dizendo que ele é bom (*good/gud*) e ótimo (*great*), bem desenvolvido (*well developed*), seguro (*secure*), focado em privacidade (*privacy*) e amigável ao usuário. Todavia, alguns deles reportam problemas com notificações e *emojis*. Há um que sugere algo quanto à função de "cutucar"(*nudge*).
- **Cluster 36:** Grupo grande, cuja maioria dos comentários se referem ao Telegram como o melhor (*best*) aplicativo, e esbanjam elogios às suas funcionalidades. Contudo, há usuários que solicitam mudanças em certas funções, como a aumentar a quantidade de contatos/grupos que podem ser fixados e separar contatos, grupos e canais.
- **Cluster 39:** Contém uma quantidade considerável de comentários. O foco principal dos usuários está no uso (*use*) do Telegram: muitos afirmam que ele é útil (*useful*); outros tratam da usabilidade, afirmando que ele é fácil/difícil (*easy/hard to use*); há ainda aqueles que sugerem que outras pessoas o usem. É possível observar alguns que reclamam de não conseguir usar certas funções, como chamadas de voz (*voice call*). Um usuário se queixa que, após atualizar, mensagens salvas perderam os *links* dos canais aos quais elas pertenciam.
- **Cluster 43:** Grupo de tamanho médio, que apresenta vários comentários que definem o Telegram como um ótimo (*great*) aplicativo. Ainda assim, há alguns usuários que reportam problemas referentes a atrasos (*delays*) em notificações de mensagens e problemas com o time de moderação (*moderating team*) do aplicativo.
- **Cluster 44:** *Cluster* de tamanho médio. Vários deles sugerem mudanças em funções já existentes na aplicação: salvamento automático de imagens na galeria do sistema, opção para definição manual de local para salvamento de áudios, distinção entre os *status* das mensagens (enviada, recebida e lida), opção de aplicar função a todos os elementos de uma lista (e.g. baixar todas as mídias de um grupo) com um toque, adição de chamadas de vídeo, grupos públicos, entre várias outras.

- **Cluster 45:** Apresenta um número muito pequeno de comentários, dos quais boa parte expressa aprovação (*well/very well*) e elogia o aplicativo. Há apenas um usuário que alega não estar recebendo notificações. Alguns reclamam que o aplicativo é lento (*slow*).
- **Cluster 46:** Grupo com baixo volume de comentários. A maioria deles expressa amor (*love*) pela aplicação, e ressaltam aspectos positivos como rapidez (*fast*), segurança e privacidade, e também por ser multi-plataforma. Poucos comentários reportam problemas e sugerem novas funcionalidades, como o caso de um que pede para fixar (*pin*) canais ou os marcar como favoritos.
- **Cluster 47:** Contém um número considerável de comentários, cuja maioria diz amar (*love*) Telegram. Há apenas alguns usuários que pedem novas funções: chamada de vídeo ou deleção de itens nos grupos.

Assim como no aplicativo do Facebook, foram obtidos alguns *clusters* de comentários sobre o Telegram que não foram descritos. Como observado na Figura 9, os *clusters* 1, 4 e 19 são os mais volumosos, sendo o *cluster* 1 o maior de todos. Seus comentários que são totalmente dissimilares ao seu *medoid* e, assim, o motivo para o tamanho do grupo é semelhante ao do *cluster* 1 do Facebook: todos os *outliers* foram agrupados aqui. Ainda assim, ele detém alguns comentários coesos que discorrem sobre a utilidade do Telegram.

Novamente, a maioria dos agrupamentos contém comentários que relatam problemas e solicitam melhorias. Da mesma forma, foi visto que os assuntos dos grupos variam, assim como os problemas e melhorias mencionados neles. Logo, reforça-se a ideia de que Jaccard gera grupos coesos até certo ponto, cujos comentários tratam de temas semelhantes ou relacionados, mas que já casos onde isso não ocorre. Tais resultados indicam que Jaccard pode não ser indicado para agrupar comentários que falam dos mesmos temas, já que a coesão dos grupos gerados é varia bastante.

Em relação ao *clusters* 4 e 19, todos os seus comentários tratam sobre o aplicativo ou algumas funcionalidades ser ou não boas (*good*), e são ou não são legais (*nice*). Tais termos são bastante gerais e podem ser utilizados para inúmeros assuntos ou contextos distintos, denotando positividade ou negatividade, entre outras coisas. Logo, entende-se que a grande quantidade de comentários presentes no grupo advém do aumento na similaridade entre ambos, devido à alta recorrência de tais palavras. Vale lembrar que foi obtido ao menos um *cluster* semelhante a partir dos comentários sobre o Facebook.

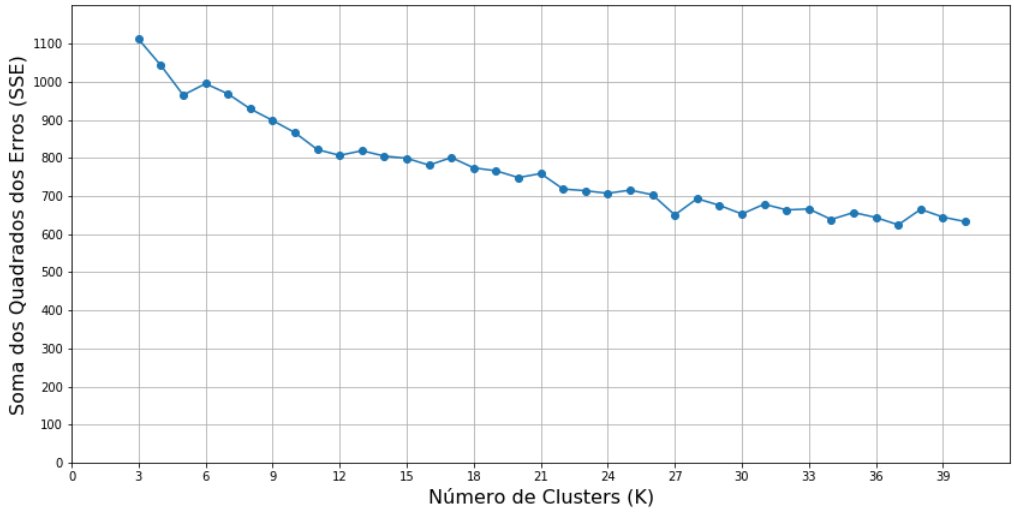
### *5.1.2 Distância de Levenshtein*

Considerando a distância de Levenshtein como medida de similaridade, a clusterização foi executada para valores do número $K$ que pertencem ao intervalo entre 3 e 40, com 20 iterações para cada. Como explicado na Seção 4.4, o alcance dos valores considerados para o número de *clusters* e iterações para Levenshtein é menor que para Jaccard, por limitações em relação ao tempo de execução do algoritmo.

As Figuras 10 e 11 ilustram os resultados do SSE para cada valor de $K$. Assim como nos resultados para Jaccard, a variação que ocorre nos pontos dos dois gráficos segue um padrão bastante similar. Vale notar que, apesar de contemplar menos valores para o número de *clusters*, a estabilização do gráfico já começa a ser percebida.

De modo geral, ao comparar a diferença entre os resultados do SSE entre as duas aplicações, é possível perceber que há uma pequena diferença entre eles, de forma o maior e o menor valor não são os mesmos: o Facebook apresenta SSEs mais altos, em comparação ao Telegram. É preciso lembrar, todavia, que o número de comentários sobre o Telegram é menor, e isso afeta diretamente os seus resultados. Ainda assim, a diferença observada entre os aplicativos é comparativamente menor no caso do Telegram, no qual ela flutua entre os valores de 100 e 150.

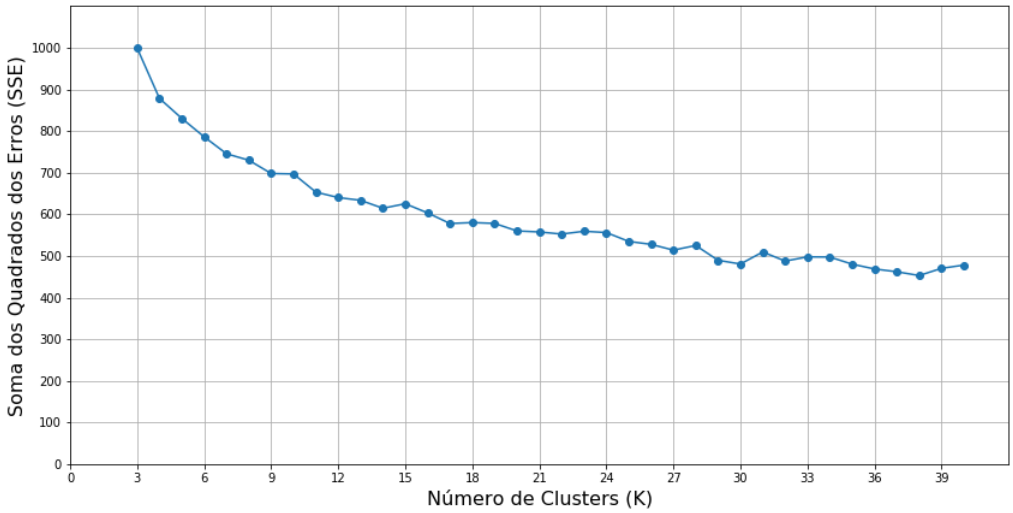
Figura 10 – Relação entre o número de *clusters* gerados pela distância de Levenshtein e o SSE obtido para os comentários do Facebook.




Fonte – elaborado pelo autor.

A seguir, são apresentados os resultados da análise dos *clusters* para os dois aplicativos.

Figura 11 – Relação entre o número de *clusters* gerados pela distância de Levenshtein e o SSE obtido para os comentários do Telegram.




Fonte – elaborado pelo autor.

#### *5.1.2.1 Facebook*

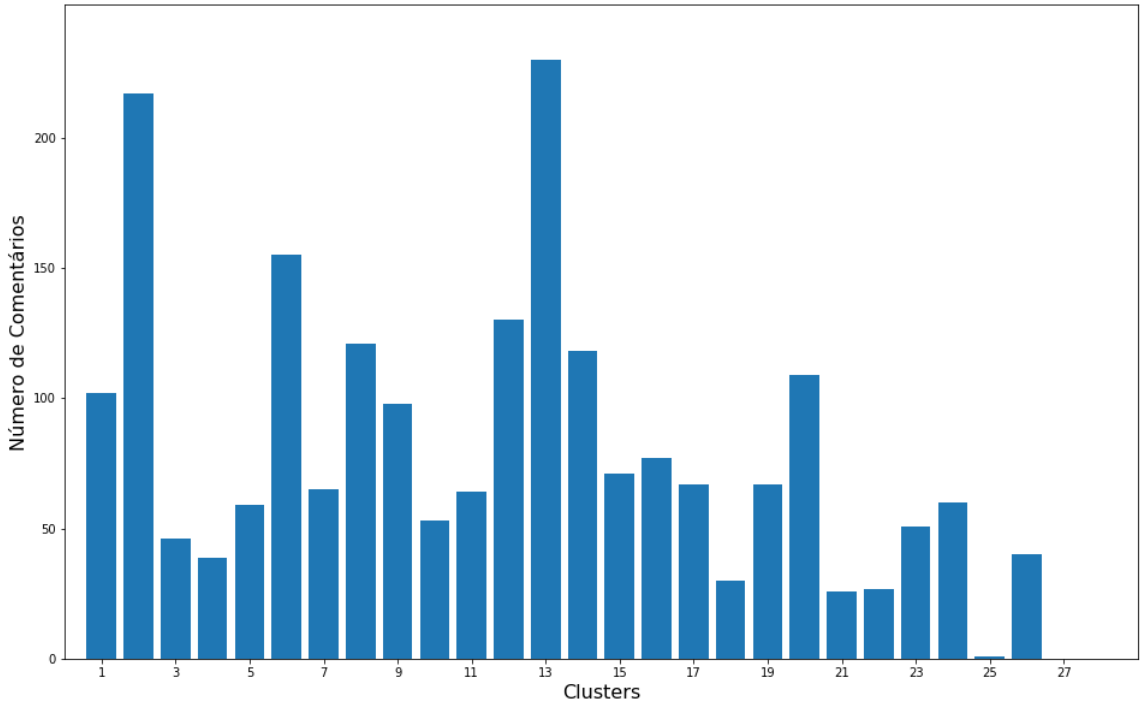
Ao considerar os comentários sobre o Facebook, a heurística definiu que o melhor valor para número *clusters* obtidos ao usar a medida foi $K = 26$, um número bem menor que o melhor valor resultante do uso da medida de Jaccard. Na Figura 12 pode ser observada a distribuição dos documentos entre os grupos obtidos.

É possível notar no gráfico que a distribuição dos comentários entre os grupos varia bastante, e que alguns poucos deles possuem uma quantidade mais expressiva de comentários, enquanto muitos apresentam tamanho pequeno e médio. Diferentemente da distribuição visualizada nos *clusters* obtidos a partir de Jaccard, o primeiro grupo não é muito maior que os outros, apesar de haverem alguns grupos bastante volumosos.

Ao analisar os 100 comentários mais similares ao *medoid* de cada *cluster*, também foram observados grupos que contém ou não comentários informativos. Em relação os que não os contém, destacam-se:

- **Cluster 1:** *Cluster* com grande quantidade de comentários, cuja maioria elogia a aplicação, definindo-a como ótima (*great*), a melhor (*best*), interessante (*interesting*) e brilhante (*brilliant*).
- **Cluster 2:** Grupo muito grande, em que todos os comentários se referem à aplicação como legal (*nice*) e útil (*useful*).

Figura 12 – Distribuição dos comentários do Facebook entre os *clusters* obtido pela distância de Levenshtein




Fonte – elaborado pelo autor.

- **Cluster 6:** Grupo que apresenta uma grande quantidade de comentários, e todos expressam amor (*love*) pelo aplicativo.
- **Cluster 13:** Apresenta um alto volume de comentários, nos quais predomina a opinião que o aplicativo é bom (*good*), mas que também conta com opiniões contrárias (*not good*).

Já entre os *clusters* que apresentam comentários contendo problemas e melhorias, nota-se que:

- **Cluster 3:** Apresenta tamanho médio, e se divide entre comentários que afirmam que o Facebook é ruim (*bad*), e outra parte discorda, expressando que o aplicativo não é tão ruim (*not bad*) ou é incrível (*amazing*). Há ainda outros comentários que se queixam de diversos problemas, entre eles vídeos apresentando falhas (*glitches*) e borrões (*blurs*).
- **Cluster 4:** Grupo pequeno, onde cerca de metade dos comentários se referem à aplicação como boa (*good*) e o restante são palavras sem significado aparente. Há um número ínfimo de comentários que citam problemas com vídeos e bloqueio de tela do dispositivo.
- **Cluster 5:** *Cluster* com quantidade razoável de comentários. Grande parte deles expressa a vantagem do Facebook em conectar pessoas, porém existem comentários que relatam problemas como: falha ao executar vídeos, alto consumo de memória (*memory*) e paradas

(*stop*) repentinas.

- **Cluster 7:** Contém quantidade razoável de comentários, e a maioria deles extensos e expressam opiniões sobre as pessoas acharem (*think*) o aplicativo bom (*good*) ou útil (*useful*), além de expressar gratidão (*thanks*). Contudo, há comentários que reclamam das novas atualizações (*updates*).
- **Cluster 8:** *Cluster* com grande quantidade de comentários. Nele, todos os comentários são extensos, e em sua grande maioria relatam problemas de variados tipos, dentre eles: falha ao realizar *log-in* e ao conectar (*connect*) a aplicação; erros na execução de vídeos, não atualização das postagens do *feed* de notícias *news*; lentidão (*slow*) e travamento (*crash*); impossibilidade de visualizar, interagir e publicar comentários (*comments*); e outros. Entretanto, um pequeno conjunto de comentários afirma que o Facebook é bom (*good*), útil (*useful*) e que o ama (*love*).
- **Cluster 9:** Apresenta um número grande de comentários, que se referem ao aplicativo como legal (*nice*), fácil de usar (*easy to use*) e útil (*useful*). Contudo, alguns usuários se queixam de problemas ao executar vídeos no aplicativo.
- **Cluster 10:** Grupo de tamanho moderado, no qual a maioria dos usuários expressam gostar (*like*) do Facebook. Todavia, há relatos de alguns tipos de problemas, em especial lentidão (*slow*) e problema na reprodução de vídeos, além de relatos que alegam que versões (*version*) anteriores da aplicação eram melhores.
- **Cluster 11:** *Cluster* de tamanho razoável, onde boa parte dos comentários são extensos e elogiam ou reclamam de algum aspecto ou funcionalidade da aplicação. Elogios utilizam adjetivos como bom (*good*), e falam que o aplicativo funciona (*work*) e que está aprovado (*OK*). Já as reclamações explicitam lentidão (*slow*) no uso do aplicativo.
- **Cluster 12:** Possui um número grande de comentários, de teor majoritariamente negativo em relação ao aplicativo. Usuários reclama de problemas como travamento (*crashing*), vírus, lentidão (*slow*), alto consumo de memória (*memory*) RAM e bateria (*battery*), e caracterizam o aplicativo como cheio de *bugs* (*buggy*, exprimindo desejo que eles sejam consertados (*fix*).
- **Cluster 14:** *Cluster* com muitos comentários, em sua grande maioria reclamando sobre atualizações constantes (*constant updates*) e o de falhas na execução dos vídeos no aplicativo do Facebook.
- **Cluster 15:** Apresenta um tamanho considerável e é composto por uma mistura de

comentários positivos e negativos, contendo reclamações sobre atualizações (*update*) e sobre a versão (*version*) atual do aplicativo. Entretanto, A maior parte dos comentários falam que o Facebook é formidável (*super*), esplêndido (*superb/splendind*), e o classificam como o melhor *better*.

- **Cluster 16:** Grupo com bastantes comentários. A maior porção dos comentários discorre sobre as frequentes atualizações (*update*) e reclamam sobre erros de instalação (*install*). Outros mencionam ainda que o aplicativo não funciona (*does not work*).
- **Cluster 17:** Apresenta uma quantidade razoável de comentários. Dentre eles, a maior parte tem conotação negativa e alguns que reportam problemas, além mesmo de comentários irônicos. Nos comentários positivos, predominam adjetivos como o melhor (*best*). Contudo, a maior parte dos comentários desse *cluster* é composta de reclamações ou relatos sobre algum problema, em especial congelamentos (*freeze*), travamentos (*crash*) e alto consumo de recursos como bateria (*battery*) e memória (*memory*).
- **Cluster 18:** Contém poucos comentários, dos quais a maioria não possui significado expressivo, e alguns detém apenas o termo "nada"(*nothing*). Há a reportagem de alguns problemas relacionados a travamentos (*crash*).
- **Cluster 19:** *Cluster* de tamanho médio, cuja grande maioria dos seus comentários possui palavras similares á *awesome*. Há alguns poucos comentários que desviam desse padrão, e entre eles há usuários que reclamam de travamentos (*crash*) na aplicação.
- **Cluster 20:** Grupo relativamente grande, que apresenta muitos comentários positivos, onde os usuários dizem gostar (*like*) e amar (*love*) o Facebook, e o descrevem como um aplicativo bom (*good*) e legal (*cool*). Há, entretanto, comentários que se queixam de diversos problemas, como lentidão (*slow*), ocupa muito espaço (*too many space*) de armazenamento e erros após atualizar (*update*) a aplicação.
- **Cluster 21:** *Cluster* com relativamente poucos comentários, que se dividem entre muitos aqueles que elogiam a aplicação, afirmando que ela é incrível (*amazing*), incrível (*awesome*) e rápido (*fast*); e poucos que discordam, dizendo que ele é ruim (*bad*) e horrível (*awful*).
- **Cluster 22:** Detém um volume moderado de comentários, que relatam de mal funcionamento (*malfunction*) do aplicativo, afirmando que ele não funciona (*not working*), é lento (*slow*) e trava (*crash*). Alguns reclamam da aplicação ser muito grande (*too big*).
- **Cluster 23:** Apresenta um número razoável de comentários, que em sua maioria definem

o Facebook como o melhor (*best*), fantástico (*fantastic*). Há, todavia, alguns poucos comentários que reclamam de travamento (*crash*) e da alta frequência de atualizações (*update*).

- **Cluster 26:** Contém um número moderado de comentários, dos quais a maioria apresenta teor positivo, em que demonstram gratidão (*thanks*) e dizem que o Facebook é um aplicativo 5 estrelas (*5 stars*). Apesar disso, há alguns usuários que reclamam de travamento (*crash/crashing*) da aplicação.

Apenas o *cluster* 25 não pode ser descrito, pois, analogamente a alguns dos grupos gerados a partir da distância de Jaccard, ele era composto apenas por seu próprio *medoid*. Contudo, é perceptível que a presença desses tipos de *clusters* é relativamente menor quando a distância de Levenshtein é aplicada.

Ao realizar a comparação entre o número de *clusters* com comentários não informativos e informativos, respectivamente, é possível ver que predominam os informativos, que relatam problemas e sugerem melhorias, assim como em Jaccard.

Como observado na Figura 12, os *clusters* 2, 6 e 13 são consideravelmente mais volumosos que os demais. Em todos eles, é observada a predominância de elogios e comentários positivos sobre o Facebook, afirmando que ele é bom (*good*), legal (*nice*) e útil (*useful*). Contudo, especificamente no *cluster* 13, são encontrados comentários com sentido negativo, falando que a aplicação não é boa (*not good*).

Observa-se que, de modo semelhante a Jaccard, os grupos gerados por Levenshtein apresentam de comentários que expressam contextos variados, ainda mais que os de Jaccard. Foram observados comentários que desviam bastante to tema geral do *cluster*, embora a sua frequência não seja muito grande. Isso indica que Levenshtein pode ser ainda menos adequado que Jaccard para agrupar comentários que discorrem sobre problemas e melhorias similares.

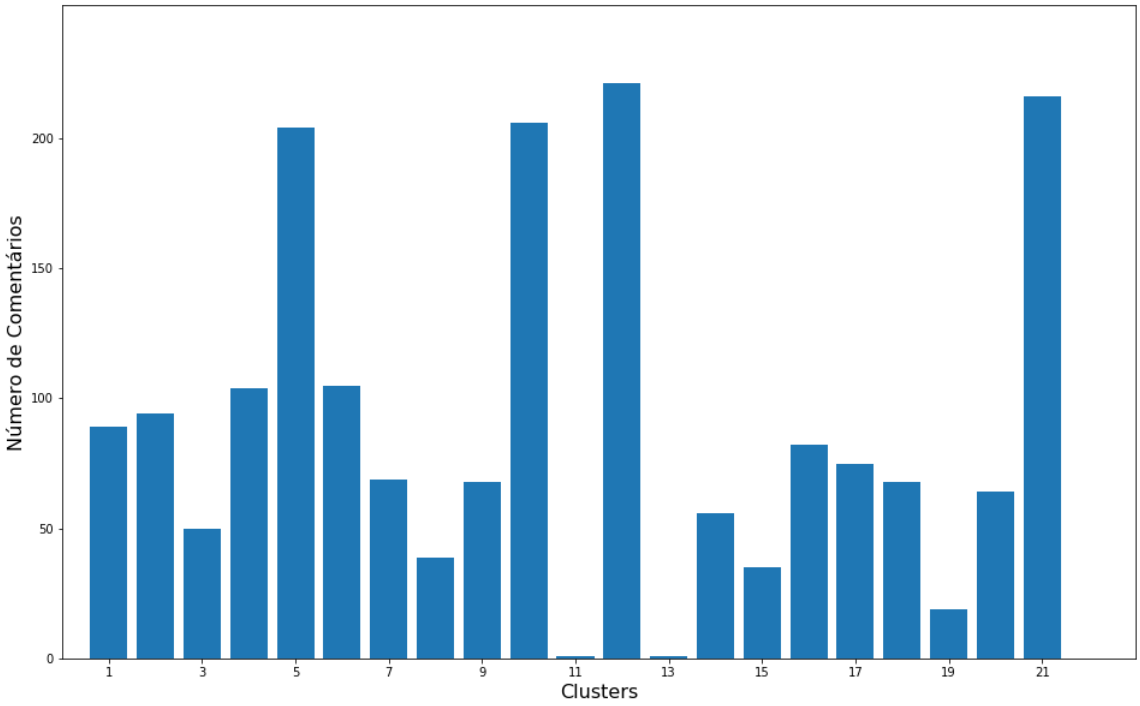
#### *5.1.2.2 Telegram*

Em relação aos comentários sobre o Telegram, o melhor número de *clusters* encontrado foi $K = 21$. A distribuição dos comentários por *cluster* pode ser visualizada na Figura 13.

Comparando-se o tamanho dos grupos com os tamanhos dos *clusters* gerados através da medida de Jaccard, é possível notar que a distribuição se assemelha em ambos, uma vez que os grupos apresentam tamanhos variados, onde há poucos *clusters* com muitos comentários,

e um número maior de grupos com relativamente poucos comentários. Todavia, é importante ressaltar que nos grupos gerados por Levenshtein, não foi observado um grupo grande e com comentários extremamente dissimilares (possíveis *outliers*).

Figura 13 – Distribuição dos comentários do Telegram entre os *clusters* obtido pela distância de Levenshtein.




Fonte – elaborado pelo autor.

A partir da análise dos 100 comentários mais similares aos *medoid* de cada *cluster*, a fim descrever o seu conteúdo e os assuntos que eles tratam. Atentando-se aos grupos compostos apenas por comentários não informativos, é possível ver que:

- **Cluster 2:** Grupo com grande quantidade de comentários, que em sua maioria diz que o aplicativo é formidável (*super*), útil (*use*) e maravilhoso (*wonderful*).
- **Cluster 3:** Apresenta quantidade razoável de comentários, que em sua maioria expressa aprovação (*OK*).
- **Cluster 4:** *Cluster* com grande quantidade de comentários, dos quais a maior parte expressa amor (*love*) e que gosta (*like*) do aplicativo.
- **Cluster 5:** Contém muitos comentários, em que todos expressam que o aplicativo é legal (*nice*).
- **Cluster 12:** Apresenta uma grande quantidade de comentários, em que todos falam que o

Telegram é bom (*good*).

- **Cluster 19:** *Cluster* com poucos comentários. Neles, os usuários dividem opiniões sobre o aplicativo ser bom (*gud*), nada mal (*not bad*) ou mesmo fabuloso (*fabulous*); e ser ruim (*bad*).
- **Cluster 21:** Grupo bastante volumoso, cuja maioria dos comentários elogia o Telegram, ao afirmar que ele é o melhor (*best*), útil (*useful*), fantástico (*fantastic*) e ótimo (*great*).

Já para os grupos que apresentam comentários informativos, tem-se que:

- **Cluster 1:** *Cluster* com grande quantidade de comentários, dos quais a maioria são comentários extensos e de conteúdos diversos. Predominam comentários que expressam elogios ao aplicativo, porém existem reclamações e relatos de problemas, como a falta de suporte a video chamada (*video call*), sugestões de melhoria na segurança (*security*), e suporte a outras linguagens (*languages*).
- **Cluster 6:** Grupo grande, com comentários extensos e diversos. A maioria deles apresentam elogios seguidos de reclamações ou sugestões de melhorias.
- **Cluster 7:** *Cluster* de tamanho razoável, que possui conteúdos diversos, entre elogios, reclamações e sugestões, cuja maioria diz que o Telegram é um bom (*good*) aplicativo, e o compara com o Whatsapp e a outras aplicações. Reclamações tem foco principal na falta da funcionalidade de vídeo chamada (*video call*) e suporte a outras linguagens (*language*).
- **Cluster 8:** *Cluster* de tamanho relativamente pequeno, em que possui elogios diversos, com alguns poucos que expressam sugestões a aplicação, como suporte a vídeo chamada, e outros dois relatam problemas de notificações (*notifications*) que não são imediatas.
- **Cluster 9:** Apresenta tamanho razoável, que em sua maioria são elogios, como formidável (*super*). Porém, há comentários que contém sugestões de melhorias, como adicionar suporte a outras linguagens (*language*), vídeo chamada (*video call*) e opções de *status*; e um comentário que relata problemas de conexão (*connection*).
- **Cluster 10:** Grupo com grande quantidade de comentários, onde quase todos descrevem o aplicativo como bom (*good*), ou afirmam que gostaram (*good*) e amaram (*love*). Contudo, um comentário expressa que o aplicativo é bom para nada (*good for nothing*).
- **Cluster 14:** Grupo com quantidade considerável de comentários. Em sua grande maioria, os usuários elogiam a aplicação, se referindo a ela como incrível (*awesome*) e surpreendente (*amazing*). Há alguns comentários que se enfocam sobre a versão da aplicação, afirmando que a nova/última versão (*new/latest version*) é ruim.

- **Cluster 15:** Contém um pequeno número de comentários, dos quais a maior parte elogia o aplicativo, ressaltando o quão fácil de usar (*easy to use*) e rápido (*fast*) ele é, e que é o melhor (*best*), comparando-o com o seu principal concorrente, o Whatsapp. Contudo, há uma minoria que solicita novas funções como vídeo chamadas (*video calls*) e organização dos *chats*.
- **Cluster 16:** Contém um número grande de comentários extensos, que está dividido entre comentários que aprovam o aplicativo, e aqueles que pedem melhorias às funções existentes ou requisitam novas, tais como baixar e executar o vídeo simultaneamente, adicionar vídeo chamadas (*video calls*) e melhorar a organização dos *chats*. Alguns se queixam de atrasos/não recebimento de notificações de mensagens, e de experienciar falhas de conexão (*connection*), lentidão (*slow*) e durante ligações (*calls*), entre diversos outros.
- **Cluster 17:** Apresenta de tamanho médio. Entre assuntos falados nos comentários, predominam a solicitação de novas funções e características, como chamadas de áudio/vídeo (*audio/video calls*), suporte a mais idiomas (*languages*) e organização dos *chats*. Certos usuários reportam erros de conexão (*connection*), lentidão (*slow*) e problemas na execução de vídeos.
- **Cluster 18:** Grupo de tamanho moderado, cuja maior parte dos comentários é positivo e elogia a aplicação, atribuindo a ela adjetivos como ótima (*great*), grandiosa (*greatest*) e perfeita (*perfect*). Alguns ressaltam o fato do Telegram ser seguro (*secure*). Há uns poucos comentários que reclamam de problemas de conexão (*connection*) e que solicitam adição de vídeo chamadas (*video calls*).
- **Cluster 20:** Contém um número considerável de comentários, que se dividem entre relatos de diversos tipos de problemas, em especial problemas de conexão (*connection*) e notificações (*notifications*); e sugestão de melhorias e adição de inúmeras funcionalidades.

Assim como para Jaccard, a clusterização utilizando Levenshtein gerou *clusters* que continham apenas *medoids* ou comentários não legíveis, embora a frequência de aparição desses tipos de *clusters* seja relativamente menor.

De acordo com a Figura 13, os grupos 5, 10, 12 e 21 são os que apresentam maior número de comentários. Em todos eles, praticamente todos os comentários mais similares expressam sentimento positivo e distribuem elogios ao Telegram, se referindo a ele como bom (*good*), ótimo (*great*), fantástico (*fantastic*), útil (*useful*), entre vários outros adjetivos.

Em relação à quantidade de grupos com comentários não informativos e informativos, é possível observar que Levenshtein gerou mais *clusters* não informativos que Jaccard. Já no que diz respeito ao conteúdo dos *clusters*, assim como nos comentários do Facebook, os assuntos dos clusters varia bastante e mais que os gerados por Jaccard. Tais resultados reforçam a suspeita de que Levenshtein não seja adequado para clusterização dos comentários para agrupar tipos de problemas e melhorias, principalmente se comparado à Jaccard.

## 5.2 Relação entre os clusters formados e as notas dos usuários

Esta seção expõe os resultados para as médias dos *clusters* obtidos a partir das distâncias consideradas, buscando relacioná-las à descrição de cada *cluster*, a fim de entender quais tipos ou grupos de comentários podem influenciar positiva e negativamente as avaliações dos usuários.

### *5.2.1 Distância de Jaccard*

Esta subseção apresenta os resultados obtidos para as notas médias dos *clusters* gerados a partir da medida de Jaccard, a relação entre o valor das notas e o conteúdo dos *clusters*.

#### *5.2.1.1 Facebook*

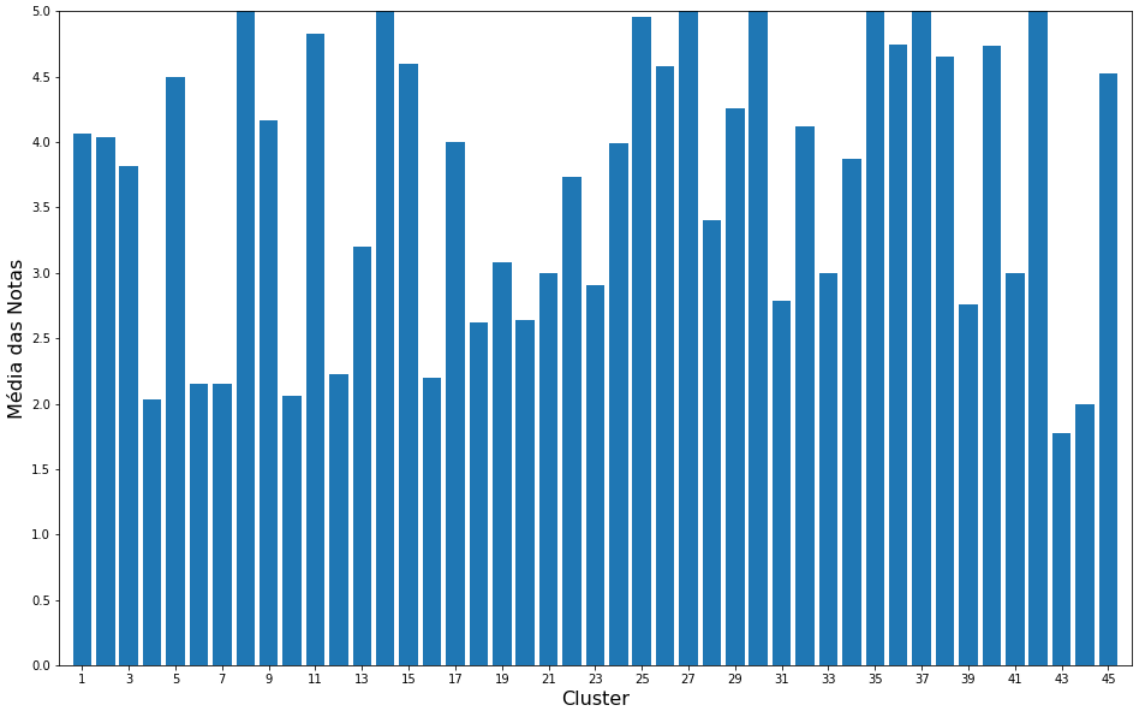
A Figura 14 apresenta as médias para os *clusters* obtidos ao utilizar a distância de Jaccard, em relação aos comentários sobre o Facebook. Na *Google Play*, a escala das notas vai de 1 a 5, não podendo ser atribuída uma nota 0. Como complemento, a Tabela 7 divide os *clusters* por valor da sua nota média.

Tabela 7 – *Clusters* do Facebook por intervalo de nota média, segundo a distância de Jaccard.

| **Nota média (M)** | ***Clusters*** | **Porcentagem (%)** |
|---|---|---|
| $1 <= M < 2$ | 43 | 0.03 |
| $2 <= M < 3$ | 4, 6, 7, 10, 12, 16, 18, 20, 23, 31, 39 e 44 | 0.27 |
| $3 <= M < 4$ | 3, 13, 19, 21, 22, 24, 28, 33, 34 e 41 | 0.23 |
| $4 <= M < 5$ | 1, 2, 5, 9, 11, 15, 17, 25, 26, 29, 32, 36, 38, 40 e 45 | 0.34 |
| $M = 5$ | 8, 14, 27, 30, 35, 37 e 42 | 0.16 |

Fonte – elaborado pelo autor.

Figura 14 – Nota média atribuída ao Facebook por *cluster* gerado a partir da distância de Jaccard.




Fonte – elaborado pelo autor.

Ao observar ambos, é possível visualizar que a distribuição é bastante variada: há múltiplos agrupamentos cuja média alcança a nota máxima; entretanto, também há vários clusters com médias baixas, embora nenhum tenha média igual a 1.

Ao analisar a divisão dos grupos por suas notas médias, foi descoberto o seguinte:

- **Nota média entre 1 e 2:** categoria composta apenas pelo *cluster* 43, que ele possui a menor valor de nota média. Representa somente 3% de todos os grupos. De acordo com a descrição feita na subseção 5.1.1.1, nele praticamente todos os comentários mais similares se referem a atualizações (*update*) na aplicação. Os usuários reclamam da frequência e do tamanho das atualizações, e de como elas parecem não trazer melhorias, além de reportar diversos tipos de problemas decorrentes delas.
- **Nota média entre 2 e 3:** representa pouco mais de um quarto (27%) de todos os *clusters* gerados. A categoria detém *clusters* pequenos e médios, nos quais há uma predominância de comentários que relatam problemas na aplicação, em especial travamentos (*crash*), congelamento (*freeze*) e lentidão (*slow*). Na maioria dos grupos, os problemas são relacionados às atualizações (*updates*) realizadas.
- **Nota média entre 3 e 4:** categoria que contém em torno de um quarto (25%) dos grupos

obtidos, que se dividem entre comentários positivos, que elogiam a aplicação; e negativos, que reclamam de vários tipos problemas experienciados no aplicativo. O tamanho dos *clusters* varia de muito pequenos a muito grandes, dentre os quais os grupos 21 e 41 contém apenas os seus próprios *medoids*. Compreende pouco menos de um quarto (23%) dos *clusters*.

- **Nota média entre 4 e 5:** apresenta a maior parte dos *clusters*, totalizando pouco mais de um terço (34%) dos agrupamentos. De forma geral, todos contém comentários bastante positivos, que afirmam gostar (*like*) da aplicação, e distribuem elogios como bom (*good/gud*), legal (*nice*), incrível (*awesome*), formidável (*super*), o melhor (*best*), entre outros. Há, todavia, comentários que falam de problemas variados. Somente o *cluster* 17 é composto apenas pelo *medoid*.
- **Nota média igual a 5:** atentando-se aos *clusters* com nota média máxima, todos são compostos apenas pelo *medoid*, que avaliavam o aplicativo com nota 5, exceto o 14 que contém somente dois comentários. Em ambos, os usuários alegam apreciar (*enjoy*) o Facebook. Representa 16% do conjunto de *clusters*.

#### *5.2.1.2 Telegram*

No caso do Telegram, a Figura 15 apresenta as notas médias dos seus *clusters*, gerados a partir da distância de Jaccard. Ademais, a Tabela 8 apresenta uma visão alternativa, ao dividir os *clusters* por intervalos de notas.

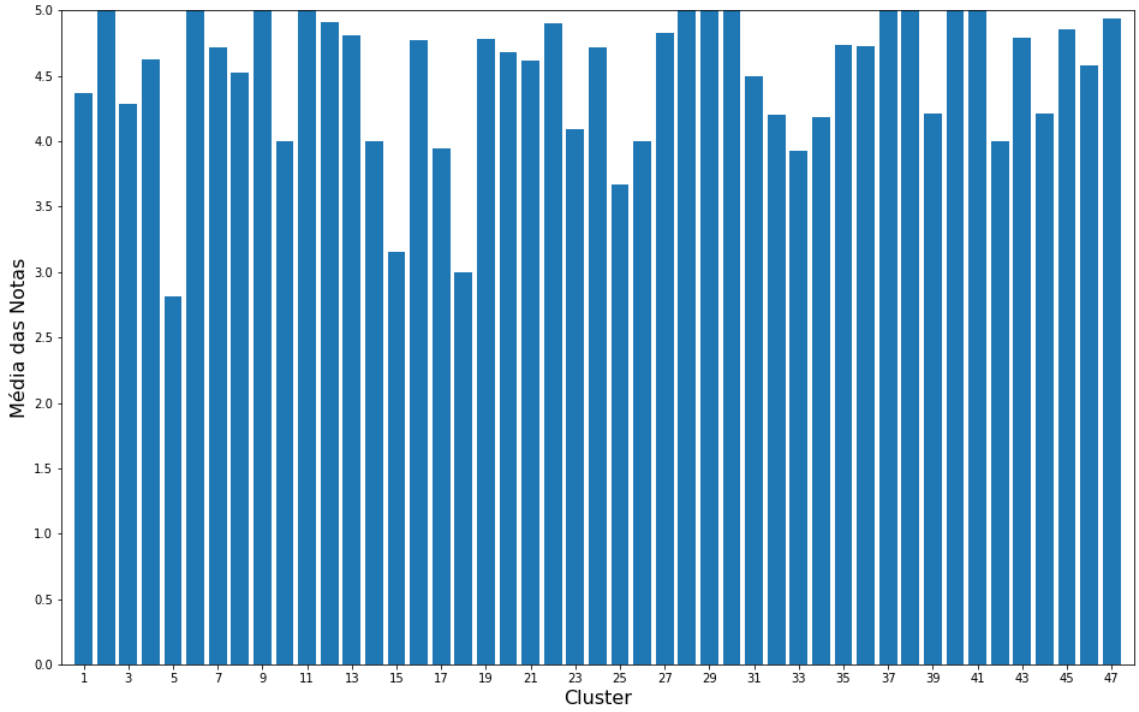
Tabela 8 – *Clusters* do Telegram por intervalo de nota média, segundo a distância de Jaccard.

| **Nota média (M)** | ***Clusters*** | **Porcentagem (%)** |
|---|---|---|
| $1 <= M < 2$ | - | 0.00 |
| $2 <= M < 3$ | 5 | 0.03 |
| $3 <= M < 4$ | 15, 17, 18, 25 e 33 | 0.11 |
| $4 <= M < 5$ | 1, 3, 4, 7, 8, 10, 12, 13, 14, 16, 19, 20, 21, 22, 23, 24, 26, 27, 31, 32, 34, 35, 36, 39, 42, 43, 44, 45, 46 e 47 | 0.64 |
| $M = 5$ | 2, 6, 9, 11, 28, 29, 30, 37, 38, 40 e 41 | 0.24 |

Fonte – elaborado pelo autor.

É possível visualizar que a maioria dos grupos apresenta médias muito altas, dos quais muitos deles alcançam a nota máxima. E mesmo os grupos com médias baixas dentre os

Figura 15 – Nota média atribuída ao Telegram por *cluster* gerado a partir da distância de Jaccard.




Fonte – elaborado pelo autor.

grupos possuem notas relativamente altas, se comparados aos **clusters** do Facebook que detém as menores médias. Entretanto, assim como o Facebook, nenhum *cluster* do Telegram apresenta a nota média mínima (igual a 1).

Aprofundando-se nos resultados para as médias de cada *cluster* e as classes às quais eles pertencem, foi observado o seguinte:

- **Nota média entre 1 e 2:** todos os grupos possuem nota média acima de 2. Logo, nenhum dos grupos se encaixou nesta categoria.
- **Nota média entre 2 e 3:** apenas o *cluster* 5 pertence a essa categoria, que representa apenas 3% do total de *clusters*. O agrupamento tem poucos comentários, dos quais metade descreve o Telegram como ruim (*bad*), e reporta problemas com o *chat* e com a função de chamada (*call*); a outra metade acha que o aplicativo não é nada mal *(not bad)*.
- **Nota média entre 3 e 4:** esta categoria conta com pouco mais de um décimo (11%) dos *clusters* gerados, que apresentam tamanhos que variam entre pequeno e médio. Na maioria dos grupos, prevalecem comentários que reportam diversos problemas, dos quais os mais recorrentes são travamentos (*crash*), lentidão (*slow*), falhas na conexão (*connection*) e no *log-in*, problemas com o *chat* e as notificações (*notificações*). Há ainda usuários que

sugerem mudanças e novas funções, especificamente relacionadas ao *chat* e à adição de chamadas de vídeo (*vídeo calls*). Parte dos comentários elogia algumas funcionalidades e a aplicação como um todo.

- **Nota média entre 4 e 5:** categoria que contém a grande maioria dos *clusters*, e representa mais da metade (64%) do total. Por conter muitos grupos, os assuntos tratados são bastante variados. De forma geral, os usuários aprovam e elogiam o Telegram, o definindo como bom (*nice*) e o melhor (*best*), e melhor (*better*) que os concorrentes (em especial o Whatsapp). Alguns vão além e citam características específicas, dizendo que ele é seguro (*secure*), respeitoso à privacidade (*privacy*), útil (*useful*) e fácil de usar e rápido (*fast*). Contudo, há uma parte dos comentários que enfoca nos problemas e nas adições que poderiam ser feitas. As reclamações mais recorrentes dizem respeito ao aplicativo ser lento (*slow*), à conexão (*connection*), *log-in*, travamentos (*crash*) e notificações (*notifications*). Dentre as funções sugeridas e requisitadas, aparecem frequentemente a possibilidade de organizar e fixar os *chats*, adição de vídeo chamadas (*video calls*) e suporte a novos idiomas (*languages*), emitir notificações ao tirar capturas de tela (*screenshots*), entre várias outras.
- **Nota média igual a 5:** para os grupos que apresentaram nova média máxima, todos são clusters compostos apenas pelo *medoid* ou que apresentam poucos comentários ilegíveis, com exceção dos *clusters* 2, 11 e 29. Nestes há pouquíssimos comentários, nos quais predominam elogios ao aplicativo, sobre como ele é excelente (*excellent*), incrível (*amazing*), e notável (*greatest*). Entretanto, há alguns poucos comentários que reclamam de lentidão (*slow*) e pedem que sejam adicionados novos idiomas.

### *5.2.2 Distância de Levenshtein*

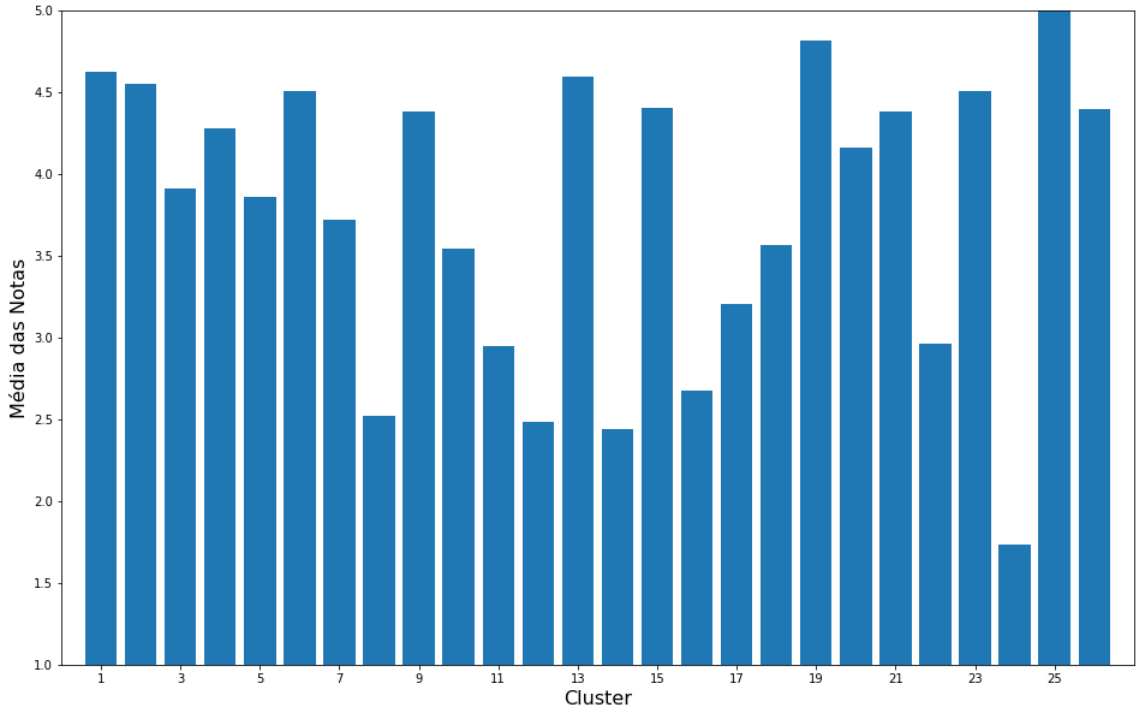
Esta subseção apresenta os resultados obtidos para as notas médias dos *clusters* gerados a partir da medida de Levenshtein, a relação entre o valor das notas e o conteúdo dos *clusters*.

#### *5.2.2.1 Facebook*

A Figura 16 apresenta a distribuição das médias para os *clusters* com comentários sobre o Facebook, gerados a partir da medida de Levenshtein. Assim como para Jaccard, nenhum dos grupos apresentou nota média igual a 1. Adicionalmente, a Tabela 9 exibe a divisão dos

*clusters*, levando em consideração as suas notas médias.

Figura 16 – Nota média atribuída ao Facebook por *cluster* gerado a partir da distância de Levenshtein.




Fonte – elaborado pelo autor.

Tabela 9 – *Clusters* do Facebook por intervalo de nota média, segundo a distância de Levenshtein.

| **Nota média (M)** | ***Clusters*** | **Porcentagem (%)** |
|---|---|---|
| $1 <= M < 2$ | 24 | 0.04 |
| $2 <= M < 3$ | 8, 11, 12, 14, 16 e 22 | 0.24 |
| $3 <= M < 4$ | 3, 5, 7, 10, 17 e 18 | 0.24 |
| $4 <= M < 5$ | 1, 2, 4, 6, 9, 13, 15, 19, 20, 21, 23 e 26 | 0.47 |
| $M = 5$ | 25 | 0.04 |

Fonte – elaborado pelo autor.

Considerando-se as categorias de *clusters* delimitadas pelos intervalos de nota média e a descrição de cada *cluster*, viu-se que:

- **Nota média entre 1 e 2:** apenas o *cluster* 24 possui nota média entre esta faixa de valores, abrangendo apenas 4% do total de *clusters*. Os comentários que o compõem são, em sua maioria, reclamações dos usuários sobre diversos problemas na aplicação, tais como

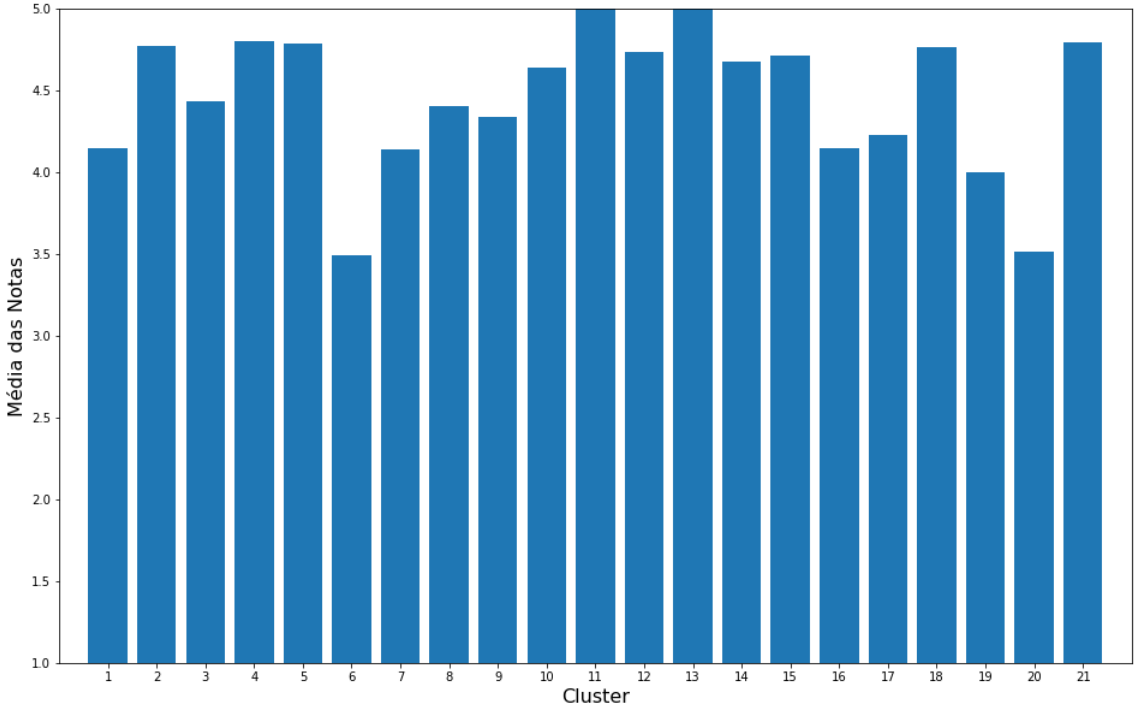
travamento (*crash*), congelamento (*freeze*), uso excessivo da bateria, erros nas notificações (*notifications*) e na execução automática de vídeos.

- **Nota média entre 2 e 3:** representa aproximadamente um quarto (25%) dos grupos gerados. A maioria *clusters* presentes nessa categoria contém comentários que se queixam de diversos tipos de problemas relatados pelos usuários. Entre os problemas mais comumente citados, estão lentidão (*slow*), travamento (*crash*), além de erros ao entrar (*log-in*) ou tentar conectar-se ao aplicativo, falhas de execução de vídeos, publicação de comentários e visualização do *news feed*. A maioria das reclamações são atribuídas às atualizações (*updates*) da aplicação, e há comentários que se queixam do tamanho delas e da frequência com que elas ocorrem.
- **Nota média entre 3 e 4:** detém pouco menos de um quarto dos *clusters*, que em sua grande maioria são divididos entre comentários que elogiam a aplicação, afirmdando que o aplicativo é bom (*good*), útil (*useful*), incrível (*amazing*) e útil (*useful*); e comentários onde os usuários reportam diversos problemas, sendo os mais frequentes travamentos (*crash*), congelamentos (*freeze*), lentidão (*slow*) e erros na execução de vídeos. Alguns usuários afirmam que versões (*versions*) da aplicação eram melhores.
- **Nota média entre 4 e 5:** a categoria com mais grupos, compreendendo quase metade (47%) deles. Em todos os grupos, prevalecem comentários positivos, que expressam gratidão (*thanks*) e distribuem elogios à aplicação, com destaque para boa (*good*), a melhor (*best*), fantástica (*fantastic*), incrível (*amazing/awesome*) e rápida (*fast*). Porém, há uma quantidade razoável de comentários que criticam o aplicativo, definindo-o como ruim (*bad/not good*), e reclamam de vários problemas, em especial travamento (*crash*), lentidão (*slow*), congelamento (*freeze*), exige muita memória e bateria, e que as atualizações (*updates*) são muito frequentes.
- **Nota média igual a 5:** contém apenas o *cluster* 25, que é composto apenas pelo seu *medoid*. Ainda assim, representa 4% dos *clusters* gerados.

#### *5.2.2.2 Telegram*

Para os grupos de comentários sobre o Telegram, a Figura 17 apresenta a distribuição das suas notas médias, enquanto a Tabela 10 apresenta os *clusters* divididos por intervalos de notas médias. Assim como todos os resultados expostos anteriormente, nenhum dos *clusters* apresentou média igual a 1.

Figura 17 – Nota média atribuída ao Telegram por *cluster* gerado a partir da distância de Levenshtein.




Fonte – elaborado pelo autor.

Tabela 10 – *Clusters* do Telegram por intervalo de nota média, segundo a distância de Levenshtein.

| **Nota média (M)** | ***Clusters*** | **Porcentagem (%)** |
|---|---|---|
| $1 <= M < 2$ | - | 0.00 |
| $2 <= M < 3$ | - | 0.00 |
| $3 <= M < 4$ | 6 e 20 | 0.10 |
| $4 <= M < 5$ | 1, 2, 3, 4, 5, 7, 8, 9, 10, 12, 14, 15, 16, 17, 18, 19 e 21 | 0.81 |
| $M = 5$ | 11 e 13 | 0.10 |

Fonte – elaborado pelo autor.

No que diz respeito aos tipos de *clusters* presentes em cada intervalo, foi observado o seguinte:

- **Nota média entre 1 e 2:** nenhum grupo apresentou nota entre a faixa de valores desta categoria.
- **Nota média entre 2 e 3:** nenhum grupo apresentou nota entre a faixa de valores desta categoria.
- **Nota média entre 3 e 4:** compreende um décimo (10%) de todos os grupos. Apenas os

*clusters* 6 e 20 pertencem a essa categoria,

- **Nota média entre 4 e 5:** representa mais de 80% de todos os agrupamentos obtidos. Possui diversos *clusters* que somente elogiam a aplicação, a definindo como boa (*good/gud*), ótima (*great*), a melhor (*best*), útil (*useful*), seguro (*secure*), entre outros adjetivos. Contudo, há a presença de *clusters* que são divididos entre comentários que elogiam e outros que reportam problemas ou solicitam novas funcionalidades. Dentre os problemas citados, os mais comuns são falhas de conexão (*connection*), lentidão (*slow*), erros ao executar vídeos e nas notificações (*notifications*). Já para os pedidos de mudanças e adição de funções, destacam-se a extensão do suporte a novas linguagens (*languages*), adição de vídeo chamadas (*video calls*) e mudança na organização dos *chats* (contatos, grupos e canais).
- **Nota média igual a 5:** categoria representada somente pelos *clusters* 11 e 13, que são compostos apenas por seus *medoids*, abrangendo um décimo (10%) do total de *clusters*.

# 6 DISCUSSÃO

Na heurística para definição do valor ideal para $K$, o intervalo de possíveis valores de $K$ considerado foi menor para Levenshtein do que para Jaccard, devido a questões de desempenho: a distância de Levenshtein é bastante custosa, exigindo alto processamento computacional e, logo, não se mostra aplicável a um conjunto de dados muito grande, ou caso se considere muitos valores para $K$. Tal problema não foi observado ao utilizar Jaccard, que apresentou tempos de execução extremamente baixos.

Apesar de tal fato parecer um problema, foi visto que os valores para Facebook e Telegram variam pouco para as duas medidas de distância entre si, mas que Levenshtein apresentou valores de $K$ menores que os definidos por Jaccard. Isso permite concluir que, para os comentários de ambas as aplicações, a relação entre os SSEs e número de *clusters* se estabiliza mais cedo utilizando Levenshtein, em detrimento de Jaccard. Logo, mesmo que valores maiores de $K$ fossem considerados para Levenshtein, o ponto de estabilização seria o mesmo e, portanto, não há prejuízo na execução da heurística.

## 6.1 Conteúdo dos clusters

A análise dos agrupamentos resultantes da clusterização utilizando as duas medidas de distância permite observar que a quantidade de comentários em cada um deles varia bastante. Foi visto que poucos *clusters* apresentam muitos comentários, enquanto os demais contém um número moderado e poucos deles. Além disso, os resultados apontam que ambas as distâncias podem gerar grupos compostos apenas pelo seu *medoid*, embora a sua frequência de aparição seja menor para a distância de Levenshtein.

No que diz respeito ao conteúdo dos *clusters*, por meio da análise dos comentários mais similares, foi visto que há *clusters* que apresentam apenas comentários positivos e que elogiam a aplicação, mas também podem ser obtidos grupos apenas com comentários que reclamam e relatam problemas encontrados no aplicativo, ou sugerem mudanças e melhorias. Os elogios e problemas mencionados variam bastante entre os *clusters*, e também entre as aplicações. Entretanto, houveram grupos com intersecções entre si, que falavam de um mesmo assunto ou se enfocavam nos mesmos aspectos.

Os problemas relatados podem ser classificados em dois tipos: gerais e específicos da aplicação. Os problemas gerais são comumente voltados às características de qualidade do

aplicativo, como desempenho, frequência ou tolerância a falhas, travamentos e paradas, tamanho e frequência das atualizações, assim como a memória (RAM ou armazenamento) utilizada pela aplicação, e outras. Dentre elas, há ainda as diretamente ligadas à plataforma na qual o *software* é executado, que no caso desse estudo são os dispositivos móveis. Tais problemas abrangem: consumo de bateria, defeitos na interface do usuário, notificações, e outros.

Já problemas específicos da aplicação dizem respeito exclusivamente às funcionalidades que ela oferece. No caso do Facebook, tem-se como exemplos *bugs* com os *likes*, comentários e postagens, *news feed* não sendo atualizado, e outros que são intrinsecamente direcionados às funções que compõem o aplicativo. No caso do Telegram, podem ser citados reclamações sobre a organização dos *chats* e erros ao realizar chamadas de áudio e ausência de vídeo chamadas. É natural que haja intersecções entre as funções oferecidas pelas aplicações, a exemplo de travamentos na execução de vídeos ou no envio e recebimento de mensagens, que são oferecidas por ambas.

## 6.2 Relação entre o conteúdo e a nota média dos *clusters*

Em relação às notas atribuídas às aplicações, para ambas as medidas de distância consideradas, descobriu-se que o Telegram apresentou *clusters* com médias mais altas que os *clusters* do Facebook, além de uma menor variação das médias dos seus *clusters*. Todavia, de modo geral, em ambas as aplicações prevaleceram agrupamentos com notas medianas e altas, dos quais alguns até mesmo apresentaram nota média máxima, embora a maioria deles fosse composto apenas pelo seu próprio *medoid*.

Para os *clusters* com médias mais baixas do Facebook, viu-se que a maior parte apresentava predominância de comentários onde os usuários relatavam diversos tipos problemas, de travamentos a abuso de recursos do dispositivo, e todos atribuíam os problemas às frequentes atualizações sofridas pelo aplicação. Isso permite inferir que, à medida que o aplicativo recebeu atualizações, novos problemas foram introduzidos, provocando o efeito contrário do esperado pelas atualizações, que é sempre buscar trazer melhorias ao *software*.

Já no caso do Telegram, pouquíssimos agrupamentos apresentavam médias baixas, e dentre os que o faziam, houve a predominância de *clusters* pequenos, dos quais parte dos comentários citavam alguns problemas, entre eles erros de conexão, lentidão e travamentos; enquanto a outra parte solicitava novas funcionalidades ou mudanças nas já existentes, como adição vídeo chamadas e mudanças na apresentação dos diferentes tipos de *chats* definidos pela

aplicação.

Atentando-se aos *clusters* com médias mais altas, foi observado que, nos dois aplicativos, os usuários comumente distribuíam inúmeros elogios e utilizavam palavras que demonstravam apreço e aprovação da aplicação. Entretanto, no caso do Facebook, foi identificado que mesmo esses grupos apresentavam vários comentários com reclamações e relato de problemas, em detrimento do Telegram, onde tais comentários foram bem menos frequentes nesses *clusters*. Vale ressaltar que tais *clusters* eram comumente os que apresentavam o maior volume de comentários, principalmente nos originados a partir da distância de Levenshtein.

## 6.3 Comparação entre as medidas de distância

No processo de clusterização, são considerados ruídos ou *outliers* os documentos que não apresentam nenhuma similaridade com os seus respectivos *medoids*, ou seja, cuja distância entre ambos é igual a 1. Nas execuções em que Jaccard foi empregado, o primeiro *cluster* foi sempre muito grande e, mesmo entre os comentários mais similares, foram encontrados *outliers*. A explicação para isso é que, nos casos onde um comentário é totalmente dissimilar a todos *K* *medoids*, o algoritmo por padrão o atribui ao "primeiro *medoid* verificado". E como as distâncias são calculadas na ordem em que os *clusters* são organizados, o *medoid* do *cluster* inicial é sempre o primeiro a ser verificado e, portanto, todos os *outliers* são alocados nele.

Ademais, em Jaccard os ruídos tendem a ser comentários com poucas palavras, uma vez que quanto menos termos um texto tiver, maior a variação na distância entre ele e outros textos.

Já para Levenshtein, tal comportamento não foi observado. Porém, é preciso considerar que a probabilidade da distância ser 1 é extremamente baixa, e diminui à medida que o tamanho dos textos aumenta. Logo, caso se queira eliminar possíveis *outliers*, é preciso utilizar uma heurística que define uma distância máxima entre um documento e o seu respectivo *medoid* para que ele não seja considerado um ruído. Para este trabalho, tal heurística não foi aplicada.

Nos comentários, houveram *clusters* compostos apenas por seu próprio *medoid*, tanto para Jaccard quanto para Levenshtein, embora o primeiro tenha gerado uma quantidade relativamente maior deles. Tal efeito também é fruto da escolha do "primeiro cluster verificado", nos casos onde há *medoids* muito similares ou idênticos entre si. Por exemplo, se considerarmos dois medoids A e B, que são iguais. Um comentário C qualquer terá a mesma distância tanto em relação a A quanto a B e será, portanto, atribuído ao *cluster* do *medoid* verificado primeiro. Por

mais que não seja um erro em si, esse efeito influencia diretamente os resultados obtidos.

Uma possível medida para evitar tal problema é garantir que os *medoids* selecionados aleatoriamente não sejam idênticos, ou que haja um limite máximo de similaridade entre eles. Com isso, além de inibir a criação de *clusters* formados apenas pelo *medoid*, consegue-se aumentar a dissimilaridade entre os *clusters* que, em teoria, aumenta a qualidade da clusterização como um todo.

Os assuntos abordados nos *clusters* gerados pela distância de Jaccard podem variar bastante, mesmo entre os comentários de um *cluster*. É perceptível que tais comentários usam os mesmos termos, mas os temas, o sentido ou o contexto são diferentes para parte ou todos eles. A Tabela 11 ilustra um exemplo onde isso ocorre. Nele, são listados alguns comentários, o *cluster* ao qual ele pertence e seu respectivo medoid. Os comentários são oriundos dos textos sobre o Telegram, e todos solicitam a adição de suporte novas linguagens, em especial a Língua Persa. Apesar de todos possuírem um tema em comum, eles foram atribuídos a grupos distintos, cujos *medoids* não possuem um tema em comum.

Tabela 11 – Exemplo que ilustra a sensibilidade de Jaccard aos termos contidos nos comentários sobre o Telegram.

| **Cluster** | **Medoid** | **Comentário** |
|---|---|---|
| 11 | Amazing app | Amazing but please add more languages to the app. |
| 12 | Thanks for all | most of user are iranian but !! why you did not support persian language ???? please increase persian language thanks telegram team |
| 26 | Why not add to the Persian language ? | Please add persian language for new updates. |

Fonte – elaborado pelo autor.

No caso da distância de Levenshtein, foi identificado, apesar serem obtidos *clusters* que apresentam majoritariamente elogios, problemas e sugestão de mudanças, os assuntos presentes em vários deles varia mais ainda que em Jaccard. Além disso, é perceptível a sensibilidade a termos parecidos, mas não idênticos e a caracteres em sequências semelhantes, que chegam a possuir mais impacto na similaridade que a presença de palavras iguais nos comentários. Adicionalmente, a quantidade de caracteres pode afetar a distância os comentários.

A Tabela 12 apresenta um exemplo de como textos com temas similares podem ser mais discrepantes que outros não relacionados. Os comentários que ele mostra são referentes ao *cluster* 11, gerado por Levenshtein e cujo *medoid* é “Facebook app stopped working“:

Tabela 12 – Exemplo que ilustra a sensibilidade de Levenshtein à posição e sequência de caracteres nos comentários sobre o Facebook.

| **Comentário** | **Distância do *medoid*** |
|---|---|
| Network | 0.62 |
| I hate this app bc it's trying to copy off of Instagram | 0.65 |
| Way to stay connected | 0.69 |
| Everything is OK | 0.69 |
| The app keeps on crashing everything I open it and nothing is working - Samsung Note 4 | 0.69 |
| My app is always showing unfortunately app is stopping | 0.72 |

Fonte – elaborado pelo autor.

É possível identificar comentários que apresentam assuntos extremamente distintos ao do *medoid*, e que também não possuem palavras em comum com ele, mas são considerados mais similares que outros que discutem sobre o mesmo tema e apresentam até mesmo palavras em comum.

O mesmo pode ser observado para Jaccard, onde os agrupamentos gerados por Jaccard, há casos onde dois comentários que tratam de um mesmo assunto são considerados menos similares que outros dois que discorrem sobre assuntos distintos. Na Tabela 13 são listados comentários do *cluster* 15, gerado a partir dos comentários do Facebook e cujo *medoid* é "It boots me out every time I try to open the app and when it does happen to stay open it won't load. AT. ALL.":

Tabela 13 – Exemplo que ilustra a sensibilidade de Jaccard à quantidade de palavras nos comentários sobre o Facebook.

| **Comentário** | **Distância do *medoid*** |
|---|---|
| Can't open the app since the new update. | 0.83 |
| Facebook fan!!! Stay in touch! | 0.91 |
| Way to stay connected | 0.91 |
| Horrible, Horrible ....updated it jus now and can't even open it at all...uninstalled and installed 5 times....that's why I don't like to update... Smh..well guess Im not a fb user anymore smfh... | 0.91 |

Fonte – elaborado pelo autor.

Isso se dá pois, devido à forma como o cálculo de Jaccard é feito, a distância é afetada não apenas a presença de palavras iguais entre os textos, mas também pela ausência de

palavras diferentes. Tal fenômeno se intensifica em comentários mais extensos, que tendem a conter mais palavras distintas.

Após a análise dos resultados obtidos para Jaccard e Levenshtein, concluiu-se que, de modo geral, ambos possuem sensibilidade à estrutura dos textos submetidos à clusterização, mas são insensíveis ao contexto ou á noção de tema, atentando-se somente às e sequências de caracteres presentes neles. Contudo, é possível afirmar que, para comentários pouco extensos, a distância de Jaccard produz *clusters* com maior coesão entre os temas abordados; em contrapartida, Levenshtein parece apresentar grupos mais coesos, quando são considerados textos longos.

Em relação ao que os usuários comentam, foi possível constatar que vários tipos de problemas são citados, desde os mais gerais e relativos às características de qualidade de *software* de modo mais abrangente, como problemas específicos da plataforma ou da aplicação em questão. De modo geral, é possível afirmar que comentários com notas mais baixas comumente contém relatos de problemas ou solicitações de novas funções e mudanças no aplicativo, indicando que os problemas possuem de fato impacto negativo na avaliação dos usuários.

# 7 CONSIDERAÇÕES FINAIS

O presente trabalho almejou definir uma abordagem para identificação automática dos problemas reportados e mudanças solicitadas, por meio da da clusterização, uma técnica de mineração de dados que busca agrupar objetos de acordo com a sua similaridade. Ao aplicar a abordagem em comentários sobre os aplicativos Facebook e Telegram para o sistema operacional Android, foi confirmado que os usuários de fato relatam a presença de problemas nos aplicativos e sugerem ou requisitam mudanças e novas funcionalidades nas aplicações. Foi visto que problemas podem ser classificados como gerais, referentes ao aplicativo enquanto *software*, ou específicos, que se enfocam em funções específicas à aplicação.

Para realizar a clusterização, foram utilizadas duas medidas de similaridade: a distância de Jaccard e a de Levenshtein. No que diz respeito à distribuição dos comentários pelos clusters gerados por ambas as medidas, foi identificado que houveram poucos grupos compostos por um número grande de comentários, e muitos grupos contendo número moderado ou baixo deles. Para Jaccard, foi observado a concentração de *outliers* no primeiro *cluster*, tornando-o sempre maior que os outros. O mesmo não foi observado para Levenshtein. Além disso, notou-se a geração de como o efeito colateral da seleção aleatória de *medoids* muito semelhantes ou idênticos, notou-se que houveram *clusters* formados apenas pelo seu *medoid*. Tal fenômeno foi mais frequente nos grupos obtidos a partir de Jaccard, o que permite concluir a sensibilidade do algoritmo a textos curtos, que contém poucas palavras.

A fim de definir o melhor valor para o número inicial de *clusters* a ser gerados, foi empregada uma heurística que realizou a para cada aplicação e levando em conta a medida utilizada. Os resultados mostraram que o número de *clusters* mais adequado para Jaccard é maior que o de Levenshtein, indicando que os resultados do último se estabilizam mais cedo que os do primeiro e que, portanto, é possível considerar um intervalo menor de valores de $K$ para Levenshtein.

Outro objetivo do estudo foi analisar a relação entre o que os usuários comentam e as notas que a aplicação recebe. Os resultados evidenciaram que comentários que relatam problemas comumente possuem notas mais baixas. No caso dos comentários que sugerem novas funcionalidades, é possível perceber que a nota deles é variada. Foi visto que há ainda comentários que elogiam a aplicação e suas funcionalidades, e que, como é de esperar, eles apresentam notas médias bastante altas. Ao comparar as médias das notas recebidas pelas duas aplicações, percebeu-se que o Telegram apresenta médias bastante altas, em detrimento do

Facebook, que demonstrou possuir notas moderadas e baixas.

Todos os dados dos comentários coletados e utilizados no trabalho estão disponíveis na internet, para acesso público. Também é disponibilizada no GitHub a aplicação construída para realizar as etapas de pré-processamento, clusterização e visualização dos resultados, se utilizando de outras bibliotecas e ferramentas *open-source*, como o NLTK e o minetext.

A análise do conteúdo dos comentários foi feito manualmente, e considerou apenas os comentários mais similares ao *medoid*. É possível aplicar técnicas de visualização para facilitar tal processo, tais como o uso de nuvens de palavras geradas automaticamente.

Dentre os diversos atributos dos comentários coletados, apenas o conteúdo e a nota foram utilizados neste trabalho. Todavia, é possível realizar investigações com enfoque em outros aspectos, por exemplo, estudar como os tópicos contidos nos comentários se comportam ao longo do tempo.

Também seria interessante aplicar outras técnicas de mineração de dados sobre os comentários, sobretudo a classificação, para averiguar se ela é mais eficaz ou adequada para identificação de problemas relatados. Em alguns grupos resultantes da clusterização, foi possível observar a predominância de certos assuntos, problemas e melhorias. A partir deles, é possível derivar classes de comentários que podem ser utiliadas como base para o processo de classificação.

# REFERÊNCIAS

AGGARWAL, C. C.; ZHAI, C. **Mining Text Data**. 1. ed. 233 Spring Street, New York, NY 10013-1578, USA: Springer Science+Business Media LLC, 2012.

ARANHA, C. N. **Uma abordagem de pré-processamento automático para mineração de textos em português: sob o enfoque da inteligência computacional. 2007. 144 f**. Tese (Doutorado) — Tese (Doutorado em Engenharia Elétrica)–Programa de Pós-Graduação em Engenharia Elétrica, Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro, R. Marquês de São Vicente, 225 - Gávea, Rio de Janeiro - RJ, 22430-060, 2007.

CARREÑO, L. V. G.; WINBLADH, K. Analysis of user comments: an approach for software requirements evolution. In: IEEE. **Software Engineering (ICSE), 2013 35th International Conference on**. San Francisco, California, USA 94111, 2013. p. 582–591.

CHEN, N.; LIN, J.; HOI, S. C.; XIAO, X.; ZHANG, B. Ar-miner: mining informative reviews for developers from mobile app marketplace. In: ACM. **Proceedings of the 36th International Conference on Software Engineering**. [S.l.], 2014. p. 767–778.

CISCO. **Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper**. 2017. Disponível em: <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html>. Acesso em: 24 mai. 2017.

CIURUMELEA, A.; SCHAUFELBÜHL, A.; PANICHELLA, S.; GALL, H. C. Analyzing reviews and code of mobile apps for better release planning. In: IEEE. **Software Analysis, Evolution and Reengineering (SANER), 2017 IEEE 24th International Conference on**. Stonemout Castle, Poland, 2017. p. 91–102.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. **AI magazine**, v. 17, n. 3, p. 37, 1996.

GAO, C.; WANG, B.; HE, P.; ZHU, J.; ZHOU, Y.; LYU, M. R. Paid: Prioritizing app issues for developers by tracking user reviews over versions. In: IEEE. **Software Reliability Engineering (ISSRE), 2015 IEEE 26th International Symposium on**. Montgomery Village Avenue, Gaithersburg, Maryland, United States, 2015. p. 35–45.

GAO, C.; XU, H.; HU, J.; ZHOU, Y. Ar-tracker: Track the dynamics of mobile apps via user review mining. In: IEEE. **Service-Oriented System Engineering (SOSE), 2015 IEEE Symposium on**. [S.l.], 2015. p. 284–290.

GUZMAN, E.; MAALEJ, W. How do users like this feature? a fine grained sentiment analysis of app reviews. In: IEEE. **Requirements Engineering Conference (RE), 2014 IEEE 22nd International**. Blekinge Institute of Technology, Campus Gräsvik, Karlskrona, Sweden, 2014. p. 153–162.

HAN, J.; PEI, J.; KAMBER, M. **Data mining: concepts and techniques**. 3. ed. 225 Wyman Street, Waltham, MA 02451, USA: Elsevier, 2012.

HAND, D. J.; MANNILA, H.; SMYTH, P. **Principles of data mining**. Suite 2, 1 Duchess Street, London, W1W 6AN, UK: MIT press, 2001.

HEARST, M. A. Untangling text data mining. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. **Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics**. College Park, Maryland, USA, 1999. p. 3–10.

IACOB, C.; HARRISON, R. Retrieving and analyzing mobile apps feature requests from online reviews. In: IEEE. **Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on**. Shanghai, China, 2013. p. 41–44.

INUKOLLU, V. N.; KESHAMONI, D. D.; KANG, T.; INUKOLLU, M. Factors influencing quality of mobile apps: Role of mobile app development life cycle. **arXiv preprint arXiv:1410.4537**, 2014.

KAO, A.; POTEET, S. R. **Natural language processing and text mining**. 1. ed. Springer-Verlag London Ltd, 236 Gray's Inn Road, Floor 6, London WC1X 8HB, United Kingdom: Springer-Verlag London Ltd., 2007.

MAN, Y.; GAO, C.; LYU, M. R.; JIANG, J. Experience report: Understanding cross-platform app issues from user reviews. In: IEEE. **Software Reliability Engineering (ISSRE), 2016 IEEE 27th International Symposium on**. Ottawa, Ontario, K1R 5T9, Canada, 2016. p. 138–149.

MATHIAK, B.; ECKSTEIN, S. Five steps to text mining in biomedical literature. In: **Proceedings of the second European workshop on data mining and text mining in bioinformatics**. Pisa, Italy: [s.n.], 2004. p. 43–46.

NGUYEN, T.-S.; LAUW, H. W.; TSAPARAS, P. Review synthesis for micro-review summarization. In: ACM. **Proceedings of the eighth ACM international conference on web search and data mining**. Shanghai, China, 2015. p. 169–178.

NICKERSON, R.; MUNTERMANN, J.; VARSHNEY, U.; ISAAC, H. Taxonomy development in information systems: Developing a taxonomy of mobile applications. In: **European Conference in Information Systems**. Verona, Italy: [s.n.], 2009.

OBILE, W. **Ericsson Mobility Report**. 2016. Disponível em: <https://www.ericsson.com/en/mobility-report>. Acesso em: 24 mai. 2017.

PAGANO, D.; MAALEJ, W. User feedback in the appstore: An empirical study. In: IEEE. **Requirements Engineering Conference (RE), 2013 21st IEEE International**. R. Marquês de São Vicente, 225 - Gávea, Rio de Janeiro - RJ, 22430-060, 2013. p. 125–134.

PIETERSE, V.; BLACK, P. E. **Levenshtein Distance**. 2015. Disponível em: <https://xlinux.nist.gov/dads/HTML/Levenshtein.htm>. Acesso em: 11 dez. 2017.

STATISTA.COM. **Number of apps available in leading app stores as of March 2017**. 2017. Disponível em: <https: //www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/>. Acesso em: 24 mai. 2017.

STATISTA.COM. **Number of available applications in the Google Play Store from December 2009 to March 2017**. 2017. Disponível em: <https://www.statista.com/statistics/ 266210/number-of-available-applications-in-the-google-play-store/>. Acesso em: 24 mai. 2017.

STATISTA.COM. **Number of available apps in the Apple App Store from July 2008 to January 2017**. 2017. Disponível em: <https://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/>. Acesso em: 24 mai. 2017.

TAN, A.-H. et al. Text mining: The state of the art and the challenges. In: SN. **Proceedings of the PAKDD 1999 Workshop on Knowledge Discovery from Advanced Databases**. Beijing, 1999. v. 8, p. 65–70.

TAN, P.-N.; STEINBACH, M.; KUMAR, V. **Introdução ao DATAMINING Mineração de Dados**. 1. ed. Rio de Janeiro: Editora Ciência Moderna, 2009.

VASA, R.; HOON, L.; MOUZAKIS, K.; NOGUCHI, A. A preliminary analysis of mobile app user reviews. In: ACM. **Proceedings of the 24th Australian Computer-Human Interaction Conference**. Melbourne, VIC, Australia, 2012. p. 241–244.