UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
PROGRAMA DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

Sadi Júnior Domingos Jacinto

Natasha - Um Sistema de Agrupamento de Histórias de Usuários Por Personas e Desejos

Florianópolis
2022

Sadi Júnior Domingos Jacinto

Natasha - Um Sistema de Agrupamento de Histórias de Usuários Por Personas e Desejos

Dissertação submetida ao Programa de Graduação em Sistemas de Informação da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Sistemas de Informação.
Orientadora: Profa. Dra. Carina Friedrich Dorneles

Florianópolis
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Jacinto, Sadi Júnior Domingos
atasha - Um Sistema de Agrupamento de Histórias de Usuários Por Personas e Desejos / Sadi Júnior Domingos Jacinto ; orientador, Carina Friedrich Dorneles, 2022.
98 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Sistema de Informação, Florianópolis, 2022.

Inclui referências.

1. Sistema de Informação. 2. Agrupamento de Dados. 3. História de Usuários. 4. K-Means, Agglomerative Clustering, DBSCAN, Gaussian Mixture Models. I. Dorneles, Carina Friedrich. II. Universidade Federal de Santa Catarina. Graduação em Sistema de Informação. III. Título.

Sadi Júnior Domingos Jacinto

Natasha - Um Sistema de Agrupamento de Histórias de Usuários Por Personas e Desejos

Trabalho de conclusão de curso apresentado como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação.

Profa. Dra. Carina Friedrich Dorneles
Orientadora
Universidade Federal de Santa Catarina

Banca Examinadora:

Prof. Dr. Renato Fileto
Universidade Federal de Santa Catarina

Prof. Dr. Jônata Tyska Carvalho
Universidade Federal de Santa Catarina

Florianópolis, 2022.

Aos meus pais

The art of programming is the art of organizing complexity. (Edsger W. Dijkstra)

RESUMO

O uso de metodologias ágeis no processo de desenvolvimento de *software* tem se popularizado nas últimas décadas e, com isso, o uso de histórias de usuários para representar os requisitos do ponto de vista dos usuários também se popularizou. Porém, uma vez que histórias de usuários são escritas por seres humanos e em linguagem natural, as mesmas estão propensas a diversos erros, como incompletude e inconsistência, além da provável existência de histórias que representam o mesmo requisito, mas que estão descritas de formas diferentes. Detectar tais inconsistências, apesar de ser uma tarefa fácil para seres humanos, é algo maçante e, em grandes conjuntos de histórias de usuários, acaba exigindo muito tempo e esforço. Assim, este projeto tem como objetivo o desenvolvimento de uma ferramenta *web* que permita a detecção e exibição de histórias de usuários similares, visando facilitar e agilizar o processo de desenvolvimento de *software*. Para tal, foram aplicados os algoritmos *K-Means*, Hierárquico Aglomerativo, DBSCAN e GMM, mostrando-se úteis para uma análise e teste de hipótese exploratórios.

Palavras-Chave: Agrupamento de Dados, Histórias de Usuários, *K-Means*, *Agglomerative Clustering*, DBSCAN, *Gaussian Mixture Models*

ABSTRACT

The use of agile methodologies in the software development process has become popular in recent decades and, with that, the use of user stories to represent requirements from the users' point of view has also become popular. However, since user stories are written by humans and in natural language, they are prone to several errors, such as incompleteness and inconsistency, in addition to the probable existence of stories that represent the same requirement, but are described in different ways. Detecting such inconsistencies, despite being an easy task for humans, is tedious and, in large sets of user stories, ends up demanding a lot of time and effort. Thus, this project aims to develop a web tool that allows the detection and display of similar user stories in order to facilitate and streamline the software development process. To this end, the K-Means, Agglomerative Clustering, DBSCAN and GMM algorithms were applied, proving to be useful for exploratory analysis and hypothesis testing.

Keywords: Data Clustering, User Stories, *K-Means*, *Agglomerative Clustering*, DBSCAN, *Gaussian Mixture Models*

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DBSCAN	*Density-Based Spatial Clustering of Applications with Noise*
GMM	*Gaussian Mixture Models*
PCA	*Principal Component Analysis*
SUS	*System Usability Scale*
TF-IDF	*Term Frequency - Inverse Document Frequency*
t-SNE	*t-Distributed Stochastic Neighbor Embedding*

SUMÁRIO

1 INTRODUÇÃO

Desenvolver *software* é uma tarefa onerosa. Os desenvolvedores não apenas precisam atentar para atender os requisitos e prazos do projeto, como, geralmente, precisam dar suporte e melhorias contínuas ao mesmo, visando atender novos requisitos que surgem dos usuários (KONRAD; GALL, 2008). Isso tudo tendo em mente que o entendimento de tais requisitos é essencial para o sucesso de um projeto (BOURQUE *et al.*, 1999; AMBREEN *et al.*, 2018).

Segundo (LINDVALL, M. *et al.*, 2002), visando melhorar o desempenho do processo de desenvolvimento de *software*, muitas empresas passaram a adotar metodologias ágeis, como o *Scrum* (SCHWABER, 1997). Por terem como características uma gestão mais leve, foco em entregas rápidas e incrementais e fortalecimento do relacionamento entre usuários e desenvolvedores, estas metodologias permitem o desenvolvimento de *software* de forma mais rápida, com melhor qualidade e mais ágil (SCHWABER; BEEDLE, 2002).

Com a popularização do uso de tais metodologias, *vide* Figura 1, popularizou-se também o uso de histórias de usuários para representar os requisitos dos usuários (LINDVALL, T., 2002; LUCASSEN *et al.*, 2016b), que consistem em pequenos textos, com um formato semiestruturado: *As [persona*[1]*], I want/want to/need/can/would like [o quê*[2]*], so [por que*[3]*].* Porém, com tal popularização, um problema surgiu: conforme a complexidade do *software* aumenta, a quantidade de histórias de usuários aumentam também, conforme apontado em (KASSAB, 2015), *vide* Figura 2.

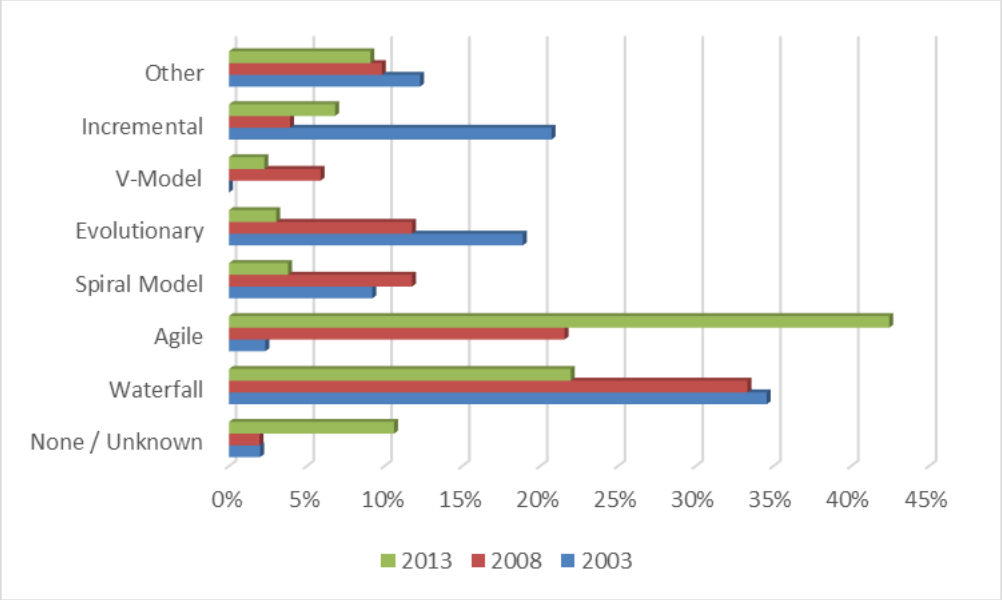

Figura 1 – Metodologias adotadas reportadas em *surveys* de 2003, 2008 e 2013
Fonte: (KASSAB, 2015)

[1] Indica para que tipo de usuário o requisito é desejado.
[2] Indica o desejo do usuário.
[3] Indica o por quê do usuário desejar algo.

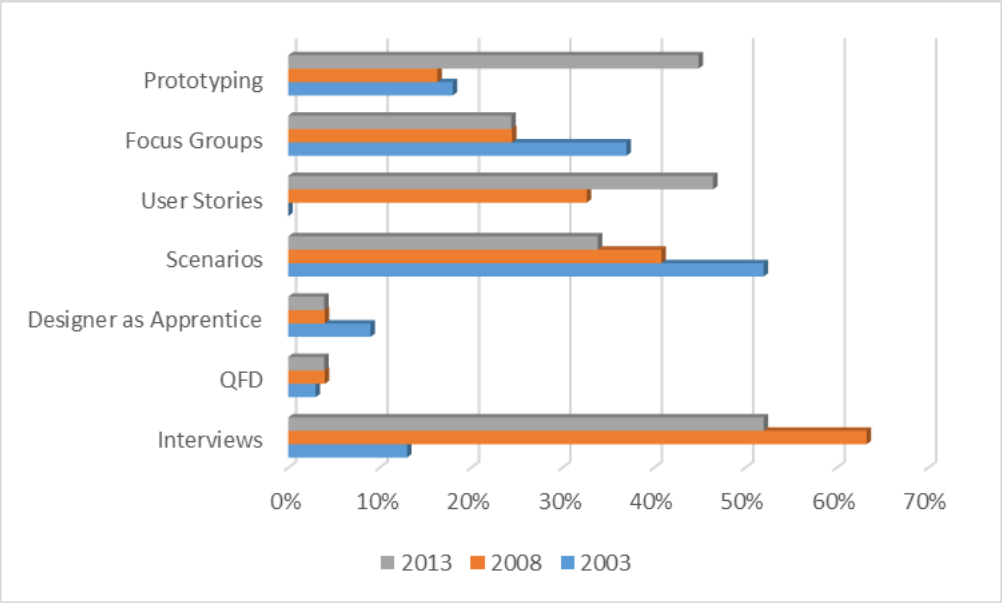

Figura 2 – Uso de técnicas para levantamento de requisitos reportadas em *surveys* de 2003, 2008 e 2013

Fonte: (KASSAB, 2015)

Apesar de tais histórias serem textos curtos e simples de serem entendidos, em grande número se tornam difíceis, do ponto de vista humano, de serem análisadas e gerenciadas. Considerando que tais histórias podem apresentar problemas como incompletude, ambiguidade, duplicidade e ocorrência de histórias similares, mas expressas de formas diferentes, o problema se torna mais complexo. Alguns exemplos desses casos podem ser vistos na Tabela 1.

História de Usuário	**Objetivo Final**
As a Publisher, I want to be able to get access to a previous version I tagged, so that I can return to it and review it. *As a Publisher, I want to be able to overwrite the previously tagged datapackage, so that I can fix it if I mess up.* *As a Publisher, I want to be warned that a tag exists when I try to overwrite it, so that I do not accidentally overwrite stable tagged data which is relied on by consumers.*	Revisar e editar, de forma segura, *tags* de versões antigas.

Tabela 1 – Exemplos de Histórias de Usuários que representam o mesmo requisito

No contexto desse problema, o uso de técnicas de agrupamento se tornam uma solução possível, visto que, por fazerem parte do aprendizado não supervisionado, não necessitam que os dados sejam previamente rotulados. Com base nisso, o presente trabalho objetiva propor uma ferramenta, *Natasha*, para identificar e apresentar, através do uso de técnicas de agrupamento, histórias de usuários que expressam requisitos similares, consequentemente auxiliando a tomada de decisão no processo de

desenvolvimento de *software*, além de também permitir unir ou criar novas histórias de usuários, facilitar definir o que precisa ser realizado primeiro, além de exibir uma visão geral e sistêmica das histórias, para que se possa quebra-lás em entregáveis durante o processo de desenvolvimento de *software*.

1.1 OBJETIVO GERAL

O objetivo geral deste trabalho consiste em oferecer uma ferramenta *web* que, dado um conjunto de requisitos de usuários, expressos na forma de histórias de usuários, agrupe requisitos similares e os apresente em uma *interface* amigável.

1.2 OBJETIVOS ESPECÍFICOS

Foram definidos os seguintes objetivos específicos para o desenvolvimento deste projeto:

- Aplicar algoritmos de agrupamento em um conjunto de dados;
- Elaborar uma análise exploratória com gráficos para visualizar o comportamento dos dados;
- Permitir agrupar os dados por *persona*;
- Verificar a possibilidade de agrupar os dados baseados na *persona* e desejo e
- Desenvolver uma ferramenta *web* que, dado um conjunto de dados, disponibilize os resultados do agrupamento de forma visual, permitindo ao usuário a escolha do algoritmo de agrupamento e seus hiperparâmetros, assim como os hiperparâmetros da representação vetorial dos dados e o algoritmo de redução de dimensionalidade.

1.3 ORGANIZAÇÃO DO TRABALHO

O presente trabalho está dividido da seguinte forma: no Capítulo 2 estão descritos os fundamentos teóricos utilizados no desenvolvimento deste projeto. O Capítulo 3 apresenta os principais trabalhos relacionados. No Capítulo 4 está descrito o processo de desenvolvimento do sistema, incluindo a análise exploratória dos dados, os experimentos realizados, suas principais conclusões, e a *interface web* desenvolvida. Por fim, no Capítulo 5 estão apresentadas as conclusões obtidas neste projeto e sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo objetiva apresentar os principais conceitos utilizados para o desenvolvimento do projeto, sendo dividido em três seções. Na Seção 2.1 está apresentado o processo de agrupamento, dividida em duas subseções, onde a primeira aborda as métricas usadas durante o processo de agrupamento, e a segunda aborda alguns dos métodos existentes para agrupar dados. A Seção 2.2 aborda o tema de redução de dimensionalidade e, por fim, a Seção 2.3 apresenta o conceito de histórias de usuários.

2.1 AGRUPAMENTO

Agrupamento consiste na implementação de técnicas para encontrar grupos similares em um conjunto de dados, sendo que tal descoberta de grupos é realizada através do uso de alguma função de similaridade. Segundo (BAEZA-YATES; RIBEIRO-NETO *et al.*, 1999), tal técnica faz parte do aprendizado de máquina não supervisionado, visto que os dados utilizados não possuem rótulos prévios e o algoritmo não passa por nenhuma fase de treinamento.

Técnicas de agrupamento são muito utilizadas em aplicações de mineração de dados, além de ter diversos estudos que a utilizam também na mineração de textos, visando extrair informações úteis e desconhecidas de textos (FELDMAN; DAGAN, 1995). São exemplos disso os trabalhos de (BAKER; MCCALLUM, 1998; BEKKERMAN *et al.*, 2001) sobre classificação de textos, assim como em (CUTTING *et al.*, 2017) para organização de documentos, entre outros.

Tal diversidade no uso de algoritmos de agrupamento em textos se dá, em parte, pelo fato que é possível utilizar técnicas de agrupamento em textos de diferentes granularidades, como documentos, parágrafos, frases e palavras. O processo de agrupamento, segundo (JAIN; MURTY; FLYNN, 1999; KING, 2015), se divide nas etapas:

1. **Representação dos Padrões ou Preparação:**
   Envolve o pré-processamento dos dados, podendo envolver normalização, conversão de tipos e *encodings*, redução de atributos, entre outros processos, para adequar os dados ao algoritmo de agrupamento a ser utilizado.

   No contexto de dados textuais, segundo (UYSAL; GUNAL, 2014), essa etapa possuí uma influência extremamente perceptível no resultado do agrupamento, sendo algumas das tarefas e técnicas que podem ser executadas nessa etapa:

   - **Tokenização:**
     Segundo (WEBSTER; KIT, 1992), esse processo consiste em dividir o texto em pedaços chamados *tokens*, que podem ser uma ou

mais palavras, ao mesmo tempo em que, preferencialmente, caractéres especiais, como pontuações e números, são removidas. O objetivo principal dessa etapa é simplificar o texto para as próximas etapas.

- **Filtragem:**
  Trata, comumente, de remover palavras extremamente comuns no texto, mas que pouco acrescentam no conteúdo, geralmente pertencendo à classes gramaticais das preposições, artigos, pronomes ou advérbios (SILVA; RIBEIRO, 2003; SAIF *et al.*, 2014).
  Outros processos podem ser usados nessa etapa, como conversão do texto para *lowercase*, remoção de números e caractéres especiais e até mesmo remoção e/ou conversão de *links* e *emojis*, caso o texto advenha de uma rede social, por exemplo.
- **Lemmatização e Stemmização:**
  Ambas buscam reduzir as palavras à sua forma raiz, para agrupar formas derivadas de uma mesma palavra e, com isso, diminuir o ruído no texto. Porém, essas duas abordagens possuem diferenças.
  A *stemmização*, desenvolvida em (LOVINS, 1968), funciona cortando o final ou início da palavra, de acordo com uma lista de prefixos e sufixos.
  Já a *lemmatização* considera a análise morfológica das palavras, tentando mapear e agrupar diferentes formas verbais para o infinitivo, assim como substantivos para uma única forma. Geralmente envolve o uso de vocabulário (SCHÜTZE; MANNING; RAGHAVAN, 2008).
- **Representação dos Dados:**
  É a última etapa do pré-processamento, que consiste em transformar os dados textuais em valores numéricos, sejam inteiros ou reais, para serem utilizados pelos algoritmos de agrupamento.
  A forma mais comum para realizar essa transformação, segundo (HOTHO; NÜRNBERGER; PAASS, 2005), é o uso de um matriz de espaço vetorial, onde cada texto é representado por um vetor de termos em uma matriz, e cada termo está associado a uma palavra do conjunto de dados, sendo um indicativo do valor semântico daquela palavra no conjunto de dados total.
  Uma das implementações mais simples desse conceito é a chamada *Bag of Words*, onde cada termo é um valor númerico inteiro, indicando sua existência no texto. Outra implementação muito conhecida é a chamada TF-IDF, que se trata do uso de medidas

estatísticas que demonstram o quão importante uma palavra é em um texto. Nesta abordagem, palavras muito frequentes tem seu valor diminuido, enquanto palavras pouco frequentes tem seu valor incrementado (HOTHO; NÜRNBERGER; PAASS, 2005; LUHN, 1957; JONES, 1972).

2. **Padrão de Proximidade:**
   Envolve a definição de uma função de proximidade, podendo ser de similaridade ou de dissimilaridade, para determinar se dois objetos fazem parte do mesmo grupo.
3. **Agrupamento:**
   Envolve a aplicação de um algoritmo de agrupamento sobre os dados.
4. **Validação:**
   Envolve validar se o algoritmo de agrupamento escolhido é adequado aos dados.
5. **Interpretação:**
   Envolve avaliar, manualmente, cada grupo gerado no processo de agrupamento, buscando obter informações novas dos dados, além de também permitir avaliações subjetivas dos grupos gerados.

Apesar de, na literatura científica, existirem diversos algoritmos de agrupamento, é de consenso que não existe um algoritmo de agrupamento universal. Assim, cada problema e cada conjunto de dados específicos requerem um ou outro algoritmo.

Além disso, muitos dos algoritmos apresentam restrições, sendo, segundo (KING, 2015; JAIN; MURTY; FLYNN, 1999), as mais comuns:

- Adequação a domínios e/ou conjuntos de dados restritos;
- Restrição dos formatos da estrutura que pode ser encontrada;
- Necessidade de conhecimento prévio do número de grupos presentes nos dados ou o difícil ajuste de parâmetros;
- Instabilidade dos resultados obtidos.

2.1.1 Métricas de Distância

São funções de proximidade, podendo ser de similaridade ou dissimilaridade, usadas para determinar se dois objetos distintos fazem parte do mesmo grupo. São exemplos de métricas de distância a euclidiana, a manhattan, a cosseno e a mahalanobis (JAIN; MURTY; FLYNN, 1999; MAO; JAIN, 1996; XU, R.; WUNSCH, 2005; MAHALANOBIS, 1936; MANLY; ALBERTO, 2016).

2.1.2 Métricas de Avaliação

Segundo (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2001) e (JAIN; DUBES, 1988), é possível avaliar agrupamentos através de três índices ou critérios:

1. Externos:
   Quando existem valores de referência externos, que podem ser utilizados como resultado esperado, é possível avaliar os grupos gerados com métricas semelhantes às usadas no aprendizado supervisionado.

   São exemplos de métricas dessa categoria a homogeneidade, completude, *V-Measure*, Índice de Rand Ajustado e Informação Mútua Ajustada (PALACIO-NIÑO; BERZAL, 2019; SHIRKHORSHIDI; AGHABOZORGI; WAH, 2015).

2. Internos:
   Quando não existem referências externas para avaliação, é possível usar métricas que avaliam os agrupamentos sobre eles próprios, aplicando algum índice que avalia a compatibilidade da estrututura dos grupos ao medir, por exemplo, a densidade dos grupos gerados, ou a distância mínima entre esses grupos, entre outros.

   São exemplos de métricas desse tipo o coeficiente de *Silhouette* (ROUSSE-EUW, 1987), o índice de Calinski-Harabasz (CALIŃSKI; HARABASZ, 1974) e o índice de Davies-Bouldin (DAVIES; BOULDIN, 1979).

3. Relativos:
   Através da comparação entre vários agrupamentos, decide-se qual é o melhor com base em algum critério pré-definido (estabilidade, adequação aos dados, tempo de processamento, entre outros). Utilizado, geralmente, para comparar diferentes algoritmos de agrupamento, assim como determinar o valor mais apropriado de algum parâmetro do algoritmo aplicado.

2.1.3 Métodos

Existem diversos algoritmos de agrupamento, que podem ser categoriazado de acordo com suas características técnicas (FAHAD *et al.*, 2014), sendo alguns deles:

- **Particionais:**
  Os dados são separados em *k* grupos, onde *k* é informado pelo usuário, sendo esse, segundo (ESTER *et al.*, 1996), uma das desvantagens dessa abordagem, uma vez que nem sempre essa informação será de conhecimento do usuário. Outras desvantagens desse método são a sensibilidade à ruídos, uma vez que todos os dados do conjunto, obrigatoriamente, farão parte de algum grupo, e sua ineficiência em encontrar grupos com formatos arbitrários.

Porém, essa abordagem tem como vantagens ser muito eficiente em agrupar grandes conjuntos de dados (GAN; MA; WU, 2020), além de ser muito simples e fácil de implementar. De forma simplificada, esse método funciona ao criar *k* partições, e alocar cada dado em uma dessas partições para, então, usar uma técnica de realocação iterativa que tenta melhorar o particionamento. Segundo (JAIN; DUBES, 1988), dos algoritmos de agrupamento por partição, o mais famoso, simples e utilizado é o *K-Means*.

- **Hierárquicos:**
  Agrupa os dados em níveis hierárquicos sucessivos, em um formato parecido com uma árvore, na qual cada folha é um grupo em sí, no topo está um grupo que agrega todos os demais, e os nodos intermediários representam a combinação ou divisão de dois grupos. O resultado final pode ser visualizado graficamente como uma árvore, chamada de dendograma (HAN; PEI; KAMBER, 2011).

  Esse tipo de método pode seguir duas abordagens diferentes:

  - Aglomerativa:
    Também chamada de abordagem *bottom-up*, na qual inicialmente cada dado é considerado um grupo, que vão sendo concatenados com grupos mais próximos, conforme o algoritmo avança para o topo da hierárquia. É a abordagem mais explorada na literatura, ao ponto de possuir diferentes formas de mesclar os grupos, popularmente chamadas de ligações (HAIR, 2009).
  - Divisiva:
    Também chamada de abordagem *top-down*, introduzida originalmente em (KAUFMAN; ROUSSEEUW, 2009), na qual os dados são inicialmente alocados em um único grupo, sendo divididos conforme a distância entre eles aumenta, de acordo com o progresso do algoritmo em direção às folhas.

- **Baseados em Densidade:**
  Algoritmos desse tipo partem do pressuposto que os grupos são formados por regiões de alta densidade. Caso um dado não obedeça critérios de densidade ou critérios dos limites de distância, este não pode ser alocado em um grupo. Um exemplo de algoritmo que utiliza essa abordagem é o DBSCAN (ESTER *et al.*, 1996).

  Esse algoritmo parte do pressuposto que, se um objeto pertence a um grupo, ele também deve estar próximo, de acordo com determinado raio de distância, a outros objetos deste grupo. Tem como principal vantagem a característica de conseguir encontrar grupos de formas arbitrárias e tamanhos

diferentes nos dados, além de também conseguir identificar e separar os ruídos dos dados, sem a necessidade prévia do número de grupos.

- **Baseados em Distribuição:**
  Algoritmos desse tipo são baseados em modelos estatísticos, uma vez que utilizam a noção de que é provável que os dados de um mesmo grupo pertençam a mesma distribuição, ou seja, as diferentes distribuições encontradas no conjunto de dados são os grupos existentes no conjunto (BILMES *et al.*, 1998). Apresenta bom funcionamento com dados sintéticos e grupos de diferentes tamanhos, mas apresenta problemas com *overfitting*, caso o ajuste de seus parâmetros não seja limitado.

  Além disso, algoritmos que utilizam essa abordagem tendem a ser ineficientes nos casos em não se saiba, *à priori*, qual o tipo de distribuição dos dados, ou caso os dados de um mesmo grupo pertençam a diferentes tipos de distribuições. Um exemplo de algoritmo desse tipo é o *Gaussian Mixture*, que parte do pressuposto que os diferentes grupos tem distribuições gaussianas multidimensionais com parâmetros variados de covariância, média e densidade.

2.2 REDUÇÃO DE DIMENSIONALIDADE

Consiste na redução de atributos em dados com muitas dimensões, mas mantendo o número mínimo necessário de parâmetros para representar as propriedades observadas nos dados originais(FUKUNAGA, 2013).

Em (JIMENEZ; LANDGREBE, 1998) foi mostrado a importância da redução de dimensionalidade em vários domínios, uma vez que, com essa transformação, a maldição da dimensionalidade[1] é mitigada, além de outras propriedades indesejadas de espaços de alta dimensão serem removidos.

As diferentes técnicas para a redução da dimensionalidade podem ser classificadas como lineares e não lineares, sendo exemplos dessas técnicas o PCA (PEARSON, 1901) e o t-SNE (VAN DER MAATEN; HINTON, 2008), respectivamente. A Tabela 2 mostra as diferenças entre esses dois algoritmos.

[1] Termo criado em (HAMMER, 1962), que, de forma breve, mostra que existe um número máximo de atributos que um modelo consegue lidar. Após esse limite, mais atributos apenas irão degradar o modelo, aumentando o custo e tempo de processamento, mas sem adicionar uma quantidade útil ou relevante de conhecimento.

PCA	**t-SNE**
Linear	Não linear
Tenta preservar a estrutura global dos dados	Tenta preservar a estrutura local dos dados
Não envolve hiperparâmetros	Envolve hiperparâmetros como perplexidade, taxa de aprendizado e número de etapas
Altamente afetado por ruídos	Consegue lidar bem com ruídos
Determinístico	Não determinístico ou aleatório
Funciona girando os vetores para preservar a variância	Funciona minimizando a distância entre o ponto em um plano guassiano
O usuário pode decidir a quantidade de variância a ser preservada	O usuário não pode preservar a variância, apenas a a distância usando hiperparâmetros

Tabela 2 – PCA *vs* t-SNE
Adaptado de (DIFFERENCE. . . , 2022)

2.3 HISTÓRIAS DE USUÁRIOS

Segundo (VERNER *et al.*, 2005), a análise de requisitos é essencial para o sucesso de um projeto, já que é necessário, primeiramente, que os desenvolvedores entendam os requisitos dos usuários antes de partir para a fase de implementação.

Nesse contexto, com o crescimento e popularização de técnicas de desenvolvimento ágil (LINDVALL, M. *et al.*, 2002), como o *Scrum* (SCHWABER, 1997), popularizou-se o uso de histórias de usuários, visando facilitar e agiliar o processo de análise de requisitos.

Histórias de usuários são artefatos consistindo em pequenos textos, com um formato semiestruturado, que representam requisitos de usuários (SCHÖN; THOMASCHEWSKI; ESCALONA, 2017; NOEL *et al.*, 2018).

De acordo com (WAUTELET *et al.*, 2014), tais artefatos possuem a seguinte estrutura:

As [persona], I want/want to/need/can/would like [o quê], so [por que]., onde:

- *persona*: Indica qual o tipo de usuário ao qual a história se refere;
- *o quê*: Indica qual o desejo desse usuário/*persona* e
- *por que*: Indica o motivo do desejo de tal usuário/*persona*.

Sendo exemplos dessa estrutura:

- *As a camp counselor, I want to be able to take attendance of my assigned kids, so that I can make ensure everybody is accounted for.*;
- *As a Developer, I want to get a Data Package into Node, so that I can start using the data for doing analysis and visualizations.*.

Apesar de tais artefatos serem facilmente compreendidos por seres humanos, por serem escritos em linguagem natural, tais artefatos não são tão facilmente compreendidos por computados. Um dos desafios que surge, especialmente em grandes organizações, é a ocorrência de histórias de usuários que expressam o mesmo requisito, mas que estão escritas de formas diferentes.

Esse tipo de similaridade, por mais que seja facilmente detectável por seres humanos, se torna inviável, do ponto de vista humano, de ser corrigida em enormes conjuntos de histórias de usuários. Uma das soluções possíveis para esse problema consiste no uso de técnicas de agrupamento de textos, melhor descritas na Seção 2.1, visando agrupar histórias de usuários semelhantes e, com isso, facilitar a análise dos dados.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta os trabalhos relacionados encontrados durante o levantamento bibliográfico. Para tal, as *search engines Google Scholar* e *IEEE Explorer*, foram utilizadas para buscar por projetos, pesquisas e ferramentas sobre agrupamento não supervisionado e visualização de histórias de usuários. A seguinte expressão lógica expressa os termos pesquisados:

- (“user story” OR “user stories” OR “user requirements” OR “software requirements”) AND (“clustering” OR “unsupervised clustering”) AND (“tool” OR “framework” OR “visual” OR “visualization”)

Com os dados coletados, alguns critérios e verificações funcionais foram adotados para filtrar os resultados, a saber:

- Disponibilidade pública, ou através do sistema de Periódicos CAPES[1], do artigo, sendo que, no mesmo, deve haver a abordagem prática de um algoritmo de agrupamento em conjuntos de dados reais. Com base nesse critério, revisões da literatura e propostas de novas abordagens em dados sintéticos foram descartadas;
- O artigo deve, obrigatoriamente, estar ou na língua inglesa ou na língua portuguesa;
- O artigo deve abordar o tópico de agrupamento de requisitos, sendo, com isso, descartados os trabalhos que abordavam outras tarefas usando histórias de usuários, como análise da qualidade dos requisitos (LUCASSEN *et al.*, 2016a);
- Quantidade de citações e
- Ano de criação e/ou publicação.

Com base nesses critérios, abaixo estão descritos os trabalhos que melhor se relacionam com o atual.

3.1 *AN APPROACH TO CLUSTERING AND SEQUENCING OF TEXTUAL REQUIREMENTS*

(BARBOSA *et al.*, 2015) buscou agrupar, e sequenciar, histórias de usuários para auxiliar engenheiros de *software* na fase de implementação, sendo que o sequenciamento dos requisitos foi baseado em um dicionário de dados que identificava as dependências funcionais das etapas semânticas em um determinado domínio.

Neste trabalho, foi utilizada a similaridade cosseno, juntamente com o algoritmo *K-Means* e o cálculo da frequência dos termos para a representação vetorial das

[1] `https://www.periodicos.capes.gov.br`

histórias de usuários, além de ser utilizada a métrica *Silhouette* para encontrar o melhor número de grupos.

As Figuras 3 e 4 ilustram, respectivamente, a abordagem adotada e os resultados obtidos. Ambas foram obtidas da publicação original.

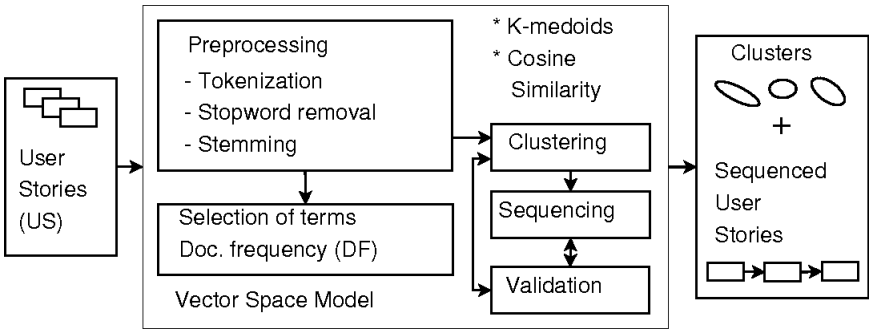

Figura 3 – Trabalho Relacionado 1 - Abordagem Utilizada
Fonte: (BARBOSA *et al.*, 2015)

Run	Max # clusters	# K-med. runs	Global Silhou--ette (S)	Generated Clusters
1	10	9	0,4278	8
2	15	14	**0.4330**	12
3	20	19	0.4060	19
4	25	24	0.4195	13
Avg	-	-	0.4215	13

Figura 4 – Trabalho Relacionado 1 - Resultados obtidos
Fonte: (BARBOSA *et al.*, 2015)

3.2 *VISUALIZING USER STORY REQUIREMENTS AT MULTIPLE GRANULARITY LEVELS VIA SEMANTIC RELATEDNESS*

Em (LUCASSEN *et al.*, 2016c), a abordagem utilizada consistiu em agrupar conceitos para, então, visualizar requisitos de usuários em diferentes níveis de granularidade, em uma interface gráfica, chamada *Visual Narrator*. Isso foi realizado com o uso da implementação do *Word2Vec* de *skip-grams*, para calcular a similaridade entre os conceitos encontrados em cada história de usuário, e o algoritmo hierárquico aglomerativo *Ward*.

As Figuras 5 e 6 ilustram, respectivamente, a abordagem utilizada e a interface criada.

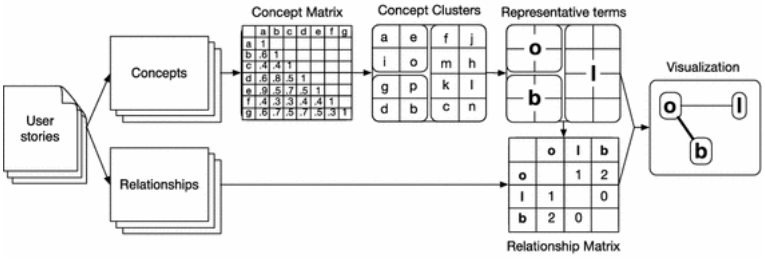

Figura 5 – Trabalho Relacionado 2 - Abordagem Utilizada
Fonte: (LUCASSEN *et al.*, 2016c)

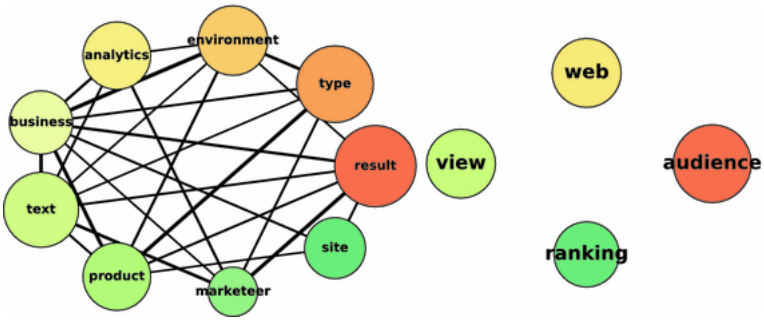

Figura 6 – Trabalho Relacionado 2 - Resultados Gráficos
Fonte: (LUCASSEN *et al.*, 2016c)

3.3 *SEMANTIC CLUSTERING OF FUNCTIONAL REQUIREMENTS USING AGGLOMERATIVE HIERARCHICAL CLUSTERING*

Em (EYAL SALMAN *et al.*, 2018), os autores optaram por uma abordagem diferente, focando em agrupar requisitos de *software* funcionais, que divergem em sua estrutura das histórias de usuários.

Para tal, foi utilizado o algoritmo hierárquico aglomerativo, calculada a similaridade semântica entre os requisitos, e exibido o resultado em um dendograma. As Figuras 7 e 8 ilustram, respectivamente, a abordagem utilizada e o resultado gráfico obtido.

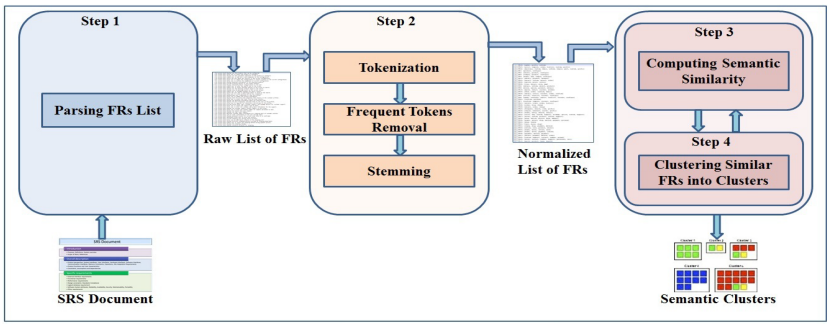

Figura 7 – Trabalho Relacionado 3 - Abordagem Utilizada
Fonte: (EYAL SALMAN *et al.*, 2018)

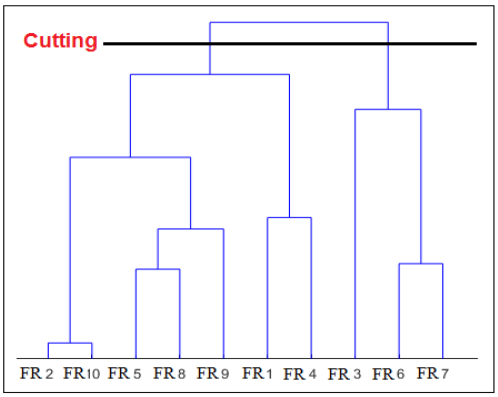

Figura 8 – Trabalho Relacionado 3 - Resultados Gráficos
Fonte: (EYAL SALMAN *et al.*, 2018)

3.4 TABELA COMPARATIVA

A Tabela 3 exibe a comparação entre os trabalhos relacionados com o atual.

Trabalho	**Métrica de Distância**	**Representação Vetorial**	**Algoritmo**	**Exibição Gráfica**	**Agrupamento por *personas***
(BARBOSA *et al.*, 2015)	Cosseno	Frequência dos termos	*K-Means*	Não	Não
(LUCASSEN *et al.*, 2016c)	*skip-grams*	*skip-grams*	Hierárquico Aglomerativo	Modelo Conceitual	Não
(EYAL SALMAN *et al.*, 2018)	Cosseno	Frequência dos termos	Hierárquico Aglomerativo	Dendograma	Não
Natasha	Cosseno, Euclidiana, Manhattan e Mahalenobis	Frequência Inversa dos Termos	*K-Means*, Hierárquico Aglomerativo, DBSCAN e *Gaussian Mixture*	Gráfico de Dispersão Interativo	Sim

Tabela 3 – Comparação entre os trabalhos relacionados

Aqui é importante notar que, comparado aos trabalhos relacionados, o atual testou diferentes métricas de distância. Outra contribuição está no uso de diferentes algoritmos de agrupamento, ao invés de disponibilizar apenas um. Finalmente, a *inter-*

face gráfica disponilizada difere, e muito, dos demais trabalhos, visto que a mesma foi pensada para ser um gráfico de dispersão interativo.

4 *NATASHA*

O presente capítulo apresenta a proposta de agrupamento, sendo dividido nas seguintes seções: a Seção 4.1 apresenta a visão geral da proposta, as premissas e ferramentas adotadas, assim como a descrição de *hardware* e *software* utilizados. A Seção 4.2 descreve o conjunto de dados utilizado. Na Seção 4.3 foram verificados e removidos dados inconsistentes. A Seção 4.4 aborda o uso de visualizações gráficas para realizar uma análise prévia dos dados. Os experimentos e avaliação dos resultados obtidos são abordados na Seção 4.5 e, por fim, a Seção 4.6 exibe a interface *web* desenvolvida.

4.1 VISÃO GERAL

Natasha é uma ferramenta *web*, cujo objetivo é o de realizar o agrupamento de histórias de usuários e apresentar os resultados em uma interface amigável e interativa. Para tal, a proposta de agrupamento utilizada nesta ferramenta foi dividida em 5 etapas:

1. **Obtenção dos Dados:** descreve os dados utilizados, assim como a forma de obtê-los;
2. **Limpeza e Pré-Processamento:** consiste em remover ruídos e dados inconsistentes;
3. **Análise Exploratória:** aplicação de técnicas estatísticas para entender melhor os dados;
4. **Experimentos e Avaliação dos Resultados**: descreve os experimentos realizados e os resultados obtidos e
5. **Desenvolvimento da Interface *web*:** detalha a interface *web* construída, assim como a avaliação da mesma.

Essas etapas estão exibidas na Figura 9.

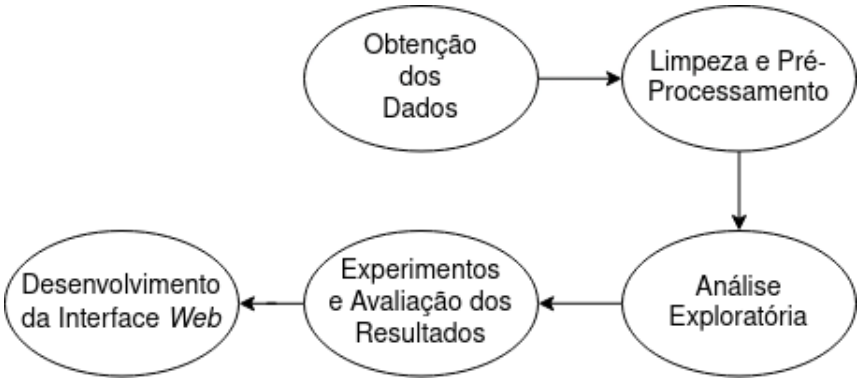

Figura 9 – Etapas do Desenvolvimento

4.1.1 Abordagem e Premissas

Apesar de existirem diversos métodos para agrupar os dados, o presente trabalho focou em explorar os métodos particionais, hierárquicos, baseados em densidade e baseados em distribuição, conforme descrito na Subseção 2.1.3. Destes, foi escolhido um algoritmo de cada tipo, a saber:

- Método Particional: Algoritmo *K-Means*.
- Método Hierárquico: Algoritmo Hierárquico Aglomerativo.
- Método Baseado em Densidade: Algoritmo DBSCAN.
- Método Baseado em Distribuição: Algoritmo *Gaussian Mixture*.

Com relação à representação vetorial adotada, foi utilizado o TF-IDF. Apesar de existirem técnicas mais avançadas para realizar a transformação vetorial de textos, como o *Word2Vec* ou o *Elmo*, optou-se por usar o TF-IDF por sua simplicidade, o que inclui a quantidade baixa de recursos computacionais necessários para o executar, e por, diferente das abordagens citadas anteriormente, não necessitar de uma etapa prévia de treinamento. Finalmente foram utilizados os métodos de redução de dimensionalidade PCA e t-SNE, tanto para verificar se, ao reduzir a dimensionalidade dos dados, isso melhoraria o desempenho dos algoritmos testados, quanto para visualizar graficamente os agrupamentos. Por fim, foram utilizadas as métricas euclidiana, manhattan, coseno e mahalanobis para o cálculo de similaridade.

A Figura 10 representa, de forma simplificada, a abordagem utilizada neste projeto. Além disso, durante os experimentos da Seção 4.5, foram testados, em um *loop*, diferentes valores de hiperparâmetros em cada um dos algoritmos de agrupamento testados e no *TfidfVectorizer*, vide Apêndice C, assim como testado diferentes formas de pré-processar os textos, vide Apêndice D.

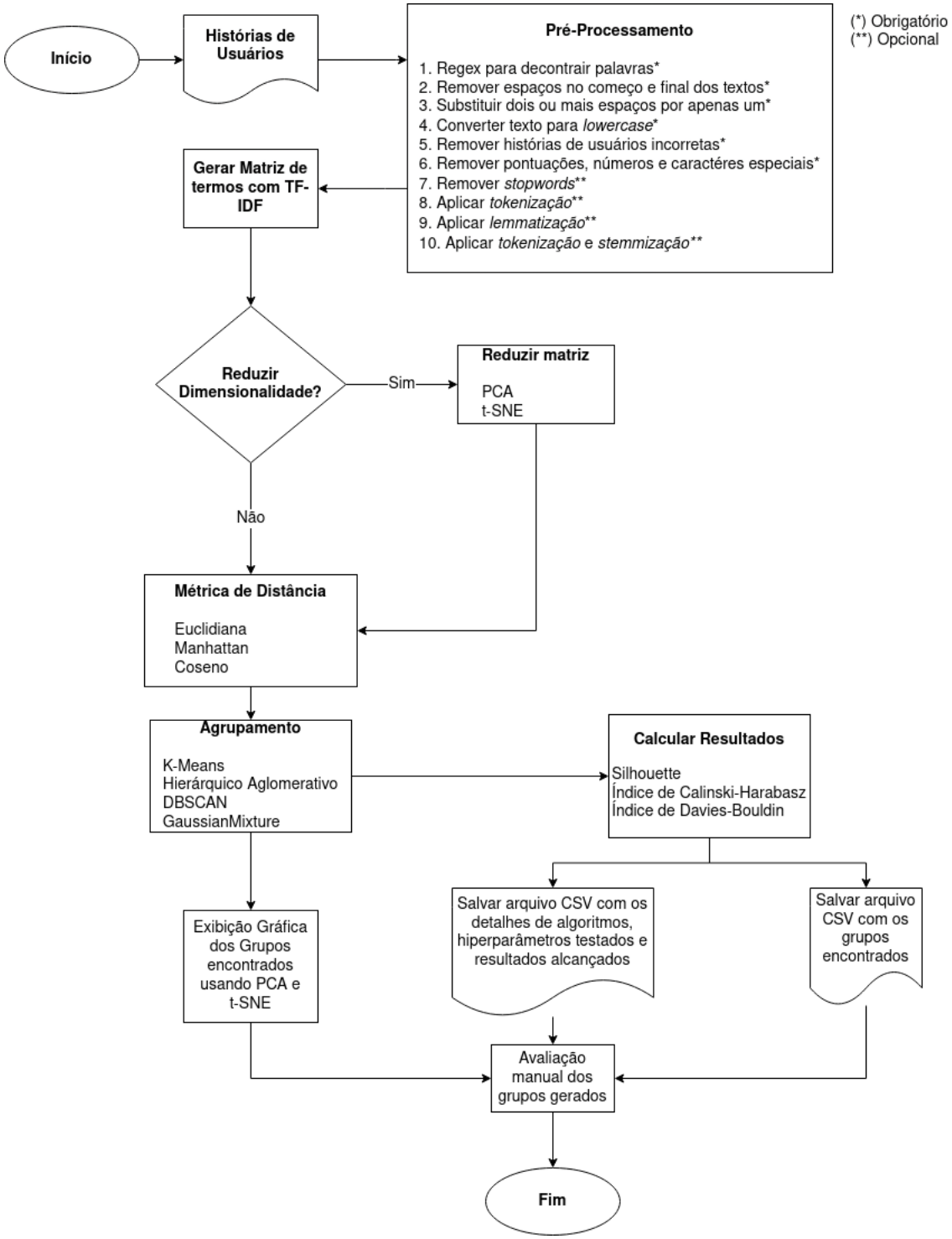

Figura 10 – Visão geral da abordagem utilizada

4.1.2 Ferramentas e Tecnologias

A linguagem utilizada para o desenvolvimento do projeto foi *Python*[1], por, além de ser de fácil entendimento e utilização, possuir diversas bibliotecas sobre ciência e análise de dados, as quais facilitam o desenvolvimento. Das diversas bibliotecas dispo-

[1] `https://www.python.org/`

níveis, foram utilizadas, majoritariamente, o *scikit_learn*[2] e o *Pandas*[3]. Com relação ao desenvolvimento da interface *web*, foi utilizado o *Dash*[4]. Maiores detalhes sobre essas, e outras bibliotecas, usadas neste projeto, assim como configurações de *hardware* e *software*, podem ser encontradas no Apêndice E.

4.2 DADOS

O conjunto de dados utilizado foi obtido na plataforma *Mendeley Data*[5], podendo ser acessado através do *link* `https://data.mendeley.com/datasets/7zbk8zsd8y/1`. Tal conjunto foi usado, originalmente, em (DALPIAZ *et al.*, 2019), para experimentos de detecção de ambiguidades em histórias de usuários, consistindo em um conjunto de 22 arquivos de histórias de usuários na língua inglesa, tendo ao total 1721 linhas. Porém, uma simples análise humana dos dados permitiu descobrir que alguns desses arquivos possuiam linhas vazias, linhas com mais de uma história de usuário, entre outras inconsistências, o que nos leva até a segunda etapa.

4.3 LIMPEZA E PRÉ-PROCESSAMENTO

Visando remover ruídos e dados inconsistentes que possam atrapalhar o resultado final, foi realizada uma limpeza e pré-processamento dos dados, a saber:

- Converter a codificação de todos os arquivos para UTF-8, concatenar todos os arquivos em um único, remover linhas vazias e converter o arquivo final para CSV:
  Para facilitar o uso dos dados, os mesmos foram convertidos para UTF-8, concatenados em um único arquivo, o qual teve suas linhas vazias removidas e, finalmente, o arquivo resultante foi convertido para CSV, contendo este um total de 1665 histórias de usuários.
- Verificar histórias de usuários com *encoding* errado:
  Após a concatenação dos arquivos, foi verificado, manualmente, a existência de 17 histórias de usuários que apresentavam erros no *encoding* das mesmas. Por se tratar de um número pequeno, tais histórias foram descartadas.
- Verificar a ocorrência de contrações:
  Por se tratar de um conjunto de dados em inglês, foi verificado a existência

[2] Biblioteca *open-source* para aprendizado de máquina, acessível em `https://scikit-learn.org/stable/getting_started.html`

[3] Biblioteca *open-source* para análise e manipulação de dados, acessível em `https://pandas.pydata.org/`

[4] *Framework open-source* para visualização de dados em interfaces *web*, acessível em `https://dash.plotly.com/`.

[5] `https://data.mendeley.com`

de contrações nas histórias de usuários, como *don't* e *I'm*. Ao todo, foram encontradas 149 contrações, sendo exemplos de algumas delas:

  - *As an Archivist, I don't want to inadvertently overwrite someone else's changes to a record that I'm editing.*
  - *As a camp administrator, I want to store camper's immediate parent/guardian's information, so that I can easily call to notify them in case a grossly unacceptable behavior.*

- Verificar a existência de valores duplicados:
  Para realizar essa verificação, primeiro foi necessário remover todos as pontuações, converter todos os textos para *lowercase* e remover espaços vazios no começo e final dos textos, assim como substituir a ocorrência, no texto, de dois ou mais espaços por apenas um. Felizmente, nenhum valor duplicado foi encontrado.
- Verificar a existência de histórias de usuários erradas:
  Foi verificado, com o uso de um *regex*, se existiam histórias de usuários que não seguiam o padrão descrito na seção 2.3. A partir dessa análise, foi descoberto que existiam 859 histórias de usuários incorretas e/ou incompletas no conjunto de dados. Alguns exemplos disso são:

  - *As a Developer, I want to list all DataPackages requirements for my project in the file and pin the exact versions of any DataPackage that my project depends on, so that that the project can be deter*
  - *As a library staff member, I want to quickly correct errors in uploaded metadata, and even uploaded documents, while leaving a record of my revisions (and possibly the reasons behind them), so that I*

  Isso é, de certa forma, algo já esperado, tendo em vista que o conjunto de dados foi originalmente usado para detecção de ambiguidades e incompletudes em histórias de usuários. Assim, por se tratarem de histórias de usuários incorretas, as mesmas foram descartadas.
- Renomear manualmente certas ocorrências de *personas* iguais com nomes diferentes como, por exemplo, as *personas admin* e *administrator*, que representam o mesmo tipo de usuário.

Após essa etapa de análise e pré-processamento, o conjunto de dados foi reduzido de 1721 histórias de usuários para apenas 789, existindo 135 tipos de personas nos dados, vide Apêndice F.

4.4 ANÁLISE EXPLORATÓRIA

Análise exploratória envolve o uso de técnicas estatísticas, gráficas e quantitativas para que seja possível entender melhor a natureza dos dados(MEDRI, 2011). Para realizar essa análise, foi verificada a distribuição das histórias de usuários em várias formas, assim como as palavras mais frequentes no conjunto de dados, e a matriz de distâncias entre as histórias.

Assim, inicialmente foi verificada a distribuição do tamanho total das histórias de usuários, descartando pontuações e caracteres especiais, vide Figura 11, buscando verificar a existência de histórias de usuários com tamanhos muito pequenos[6], de forma a encontrar possíveis histórias de usuários incompletas, sendo que essa análise não encontrou nenhuma história de usuário com menos de 10 palavras.

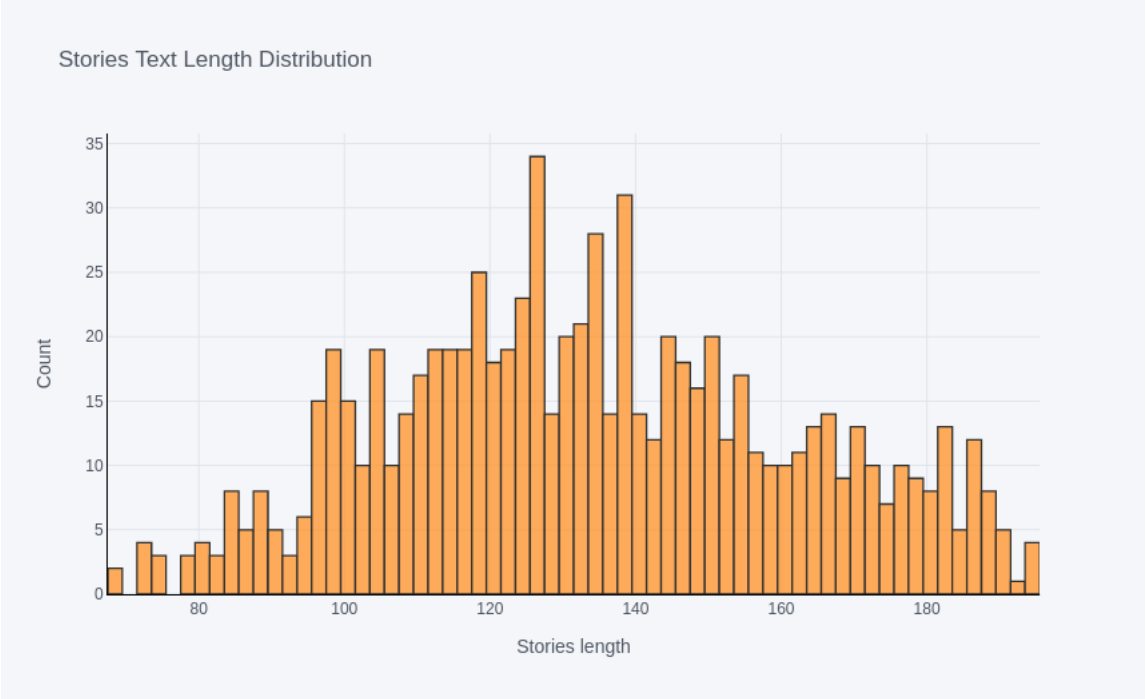

Figura 11 – Distribuição dos tamanhos das histórias de usuários

Foi averiguada, na Figura 12 a distribuição do total de palavras das histórias de usuários, visando ter uma maior ideia do tamanho das histórias, assim como verificar os tamanhos mínimo e máximo delas. Nessa verificação, foi possível encontrar três histórias de usuários que possuíam apenas 15 palavras. Ao analisá-las manualmente, foi possível perceber que, mesmo com esse número reduzido de palavras, eram histórias de usuários válidas e corretas.

[6] Menos de 10 palavras.

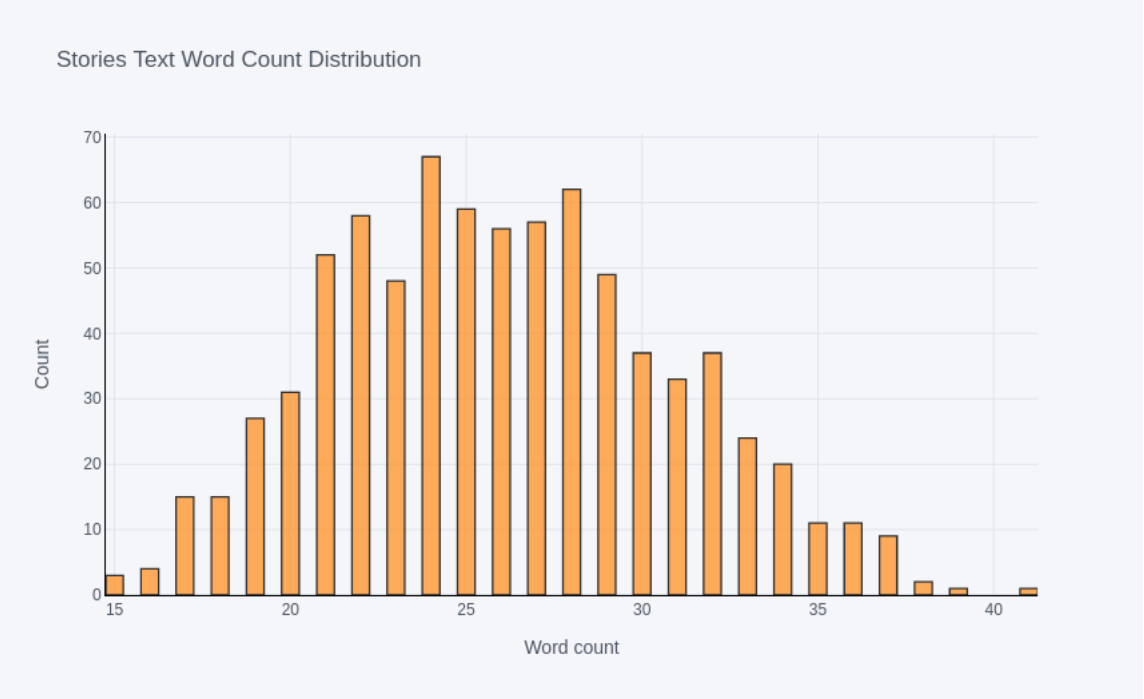

Figura 12 – Distribuição de palavras por histórias de usuários

Para obter uma comprensão maior dos dados, foi verificado, nas Figuras 13 e 14, as vinte palavras mais frequentes, sendo que na primeira não foi removida as *stop words*, enquanto que, na segunda sim. Com essa análise, foi possível perceber que a lista de *stop words*, da biblioteca *nltk*, não contém a palavra *want*. Como essa palavra faz parte da estrutura de uma história de usuário, a mesma foi adicionada como uma *stop word*, por ser muito frequente.

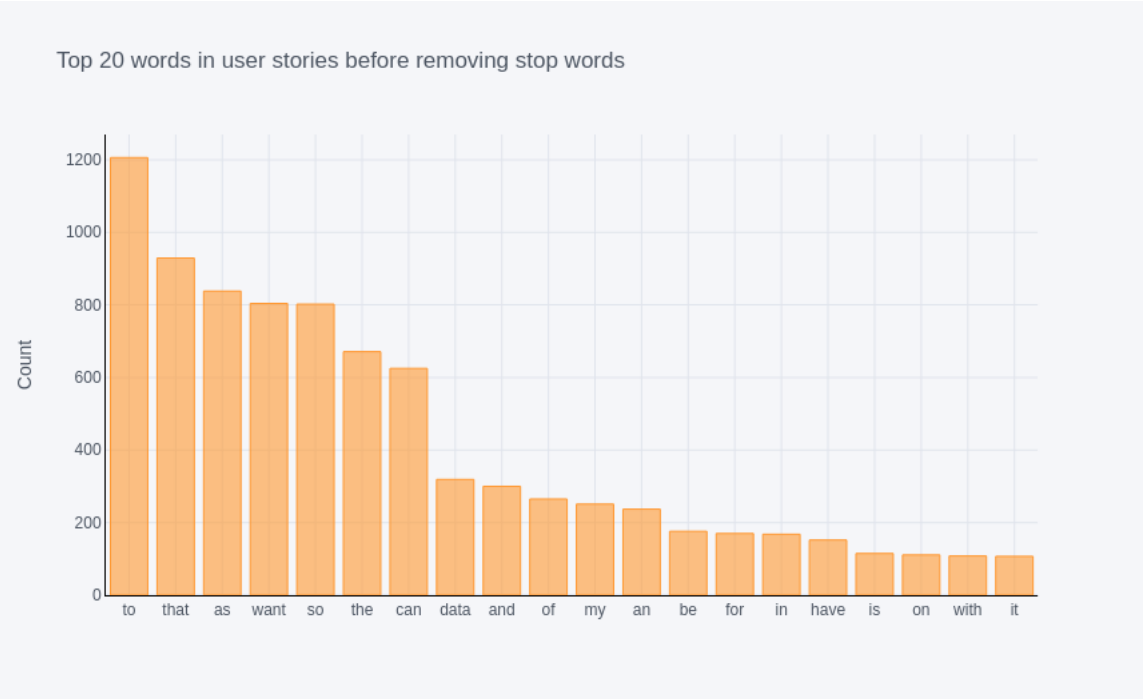

Figura 13 – 20 palavras mais frequentes

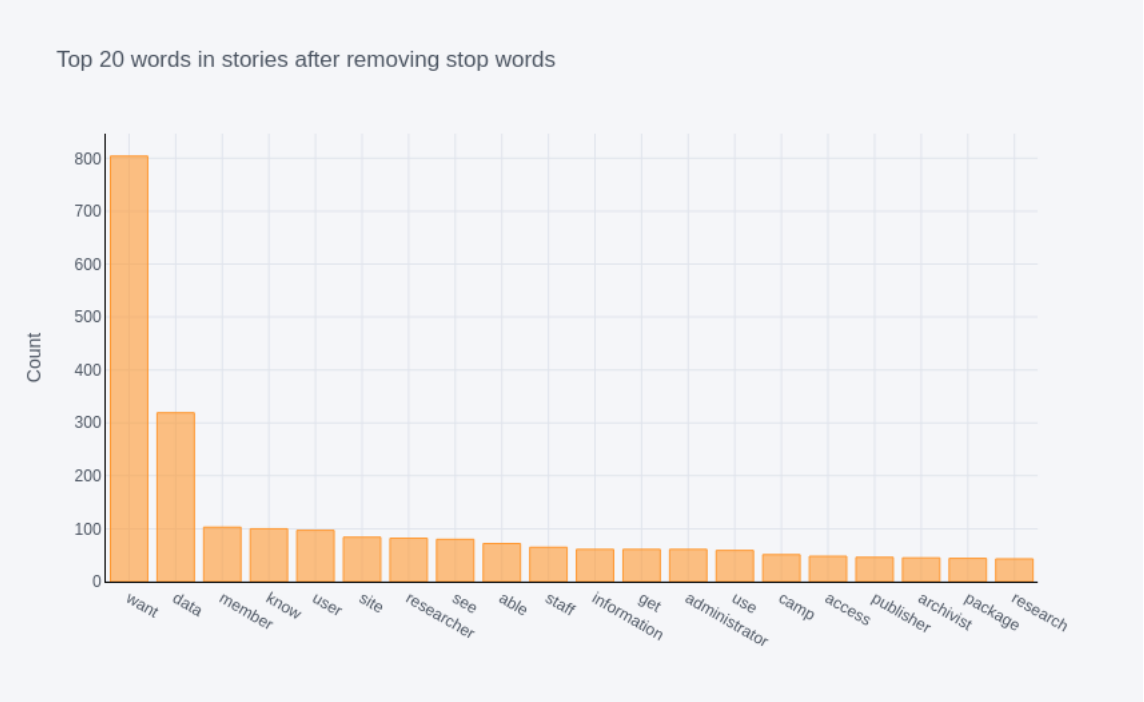

Figura 14 – Distribuição de palavras por histórias de usuários após remover *stop words*

Finalmente, foi utilizada a função de *heatmap*, da biblioteca *seaborn*, para verificar a distância entre as todas as histórias de usuários, usando as métricas de distância euclidiana (Figura 15), manhattan (Figura 16) e coseno (Figura 17), buscando verificar, graficamente, se alguma dessas distâncias melhor se adequava aos dados. Nessas figuras, quanto mais próximo da cor amarela, mas próximas são as histórias enquanto que, quanto mais próximo do azul escuro, mais distantes são as histórias. Baseando-se nesse análise, foi possível perceber que todas as métricas de distância utilizadas se mostraram, infelizmente, baixas.

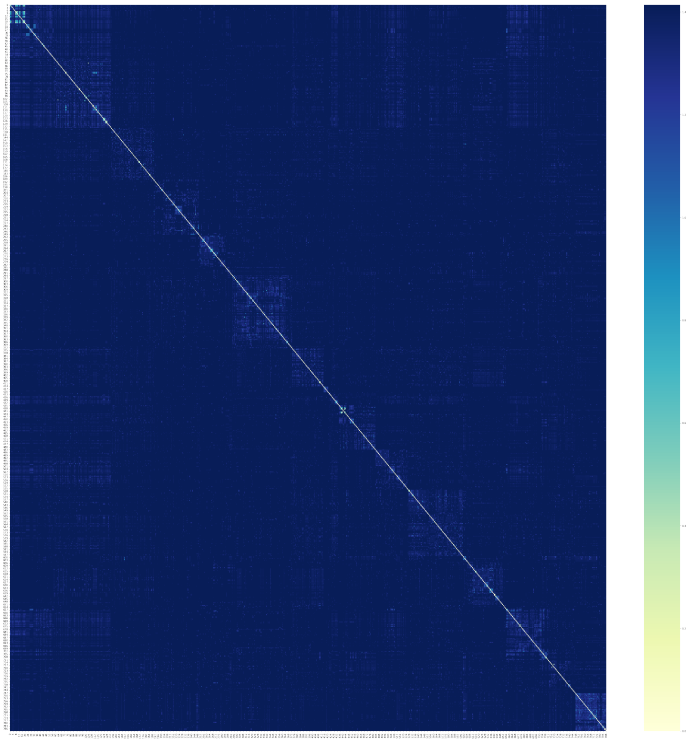

Figura 15 – Distância euclidiana entre todas as histórias

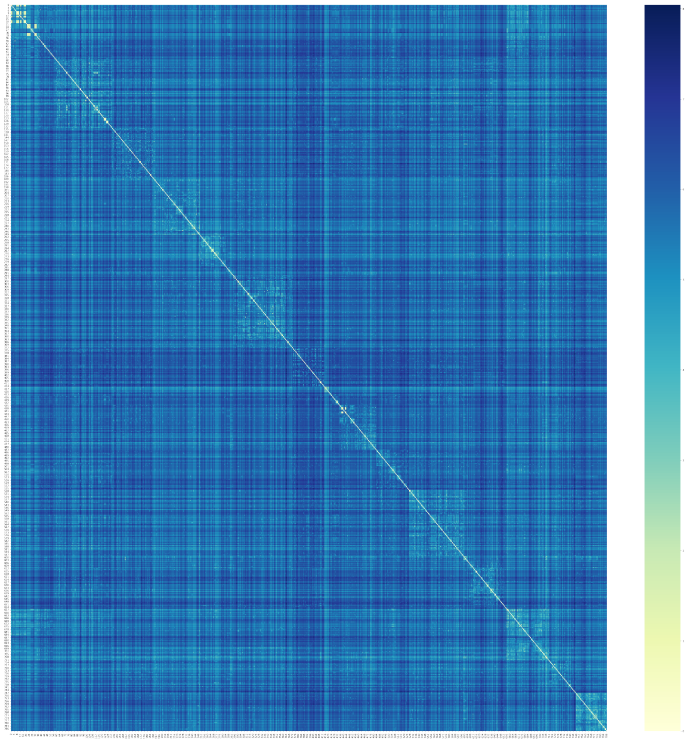

Figura 16 – Distância manhattan entre todas as histórias

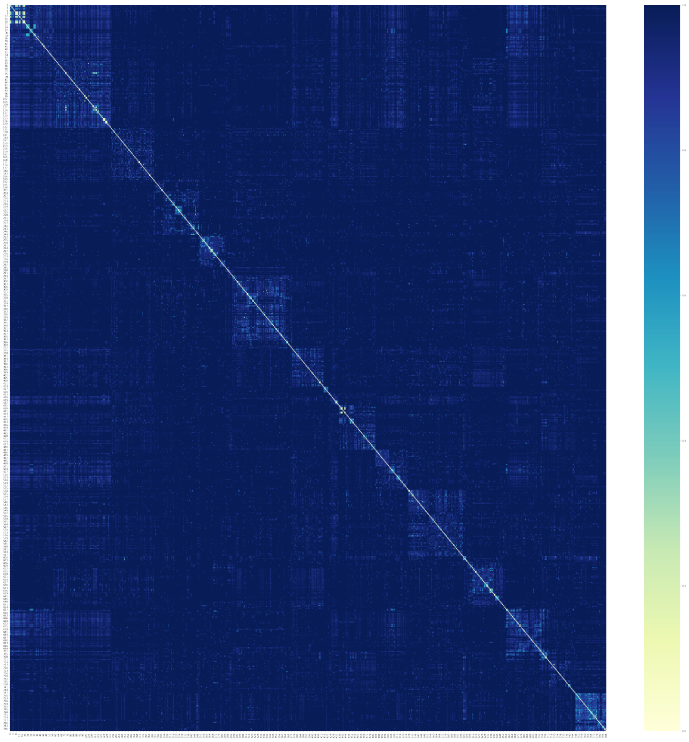

Figura 17 – Distância coseno entre todas as histórias

4.5 EXPERIMENTOS

De forma geral, os experimentos podem ser divididos em duas grandes fases:

1. Verificar agrupamento de todo o conjunto de dados:
   A primeira fase buscou descobrir se os algoritmos de agrupamento selecionados possuíam a capacidade de agrupar, corretamente, as histórias de usuários em grupos baseados na *persona* e no desejo[7], de forma que desejos semelhantes, mas de *personas* diferentes, sejam classificados separadamente, enquanto desejos semelhantes de *personas* iguais sejam agrupados.
2. Verificar agrupamento a partir das *personas*:
   A segunda fase derivou dos resultados da primeira, buscando, primeiramente, criar, através do uso de *regex*, grupos de *personas* e, a partir desses grupos, aplicar novamente os algoritmos de agrupamento, visando, agora, agrupar os dados apenas pelos desejos expressos nas histórias de cada grupo de *persona*.

4.5.1 Agrupamento por *personas* e desejos

Inicialmente, foi verificado se os algoritmos de agrupamento possuíam a capacidade de criar grupos que separassem corretamente os dados por categoria de *persona* e desejo. Como a análise prévia do conjunto de dados permitiu descobrir existirem

[7] O *want*, conforme descrito na seção 2.3.

135 diferentes categorias de *personas*[8], foi utilizado esse valor para os algoritmos que exigiam como parâmetro um número *k* de grupos. Também foi verificado se o DBSCAN possuía alguma combinação de hiperparâmetros que permitisse dividir os dados nesse número de grupos, porém, sem sucesso, assim, optou-se por filtrar os resultados do DBSCAN por um valor de grupos gerados igual ou superior à 100.

A Tabela 4 apresenta os melhores resultados obtidos por algoritmo, sendo que o Apêndice G detalha e analisa os resultados individuais de cada um. Nessa etapa de testes, as conclusões gerais foram:

- A métrica de avaliação *silhouette*, com algumas combinações de algoritmos e hiperparâmetros, atingia seu valor máximo (1), porém, os grupos gerados não faziam o menor sentido do ponto de vista humano. Assim, uma das filtragens dos resultados obtidos consistiu em filtrar os resultados por essa métrica usando um valor abaixo de 1;
- Aplicar a redução de dimensionalidade nos dados, antes de agrupá-los, aumentou consideravelmente o valor das métricas de avaliação dos grupos, porém, por uma análise humana dos grupos gerados, foi constatado que os mesmos, do ponto de vista humano, não tinham significado, assim, apesar de terem sido realizados testes usando os dados reduzidos, os mesmos, após passarem pela etapa de avaliação humana, foram descartados.
- Mesmo testando diferentes valores de hiperparâmetros, algoritmos, reduzindo a dimensionalidade dos dados, entre outros, os grupos gerados nesse primeiro experimento mostraram-se incapazes de agrupar, corretamente, os dados por categoria de *persona* e desejo. Porém, isso não significa que essa é uma tarefa impossível, já que pode existir algum conjunto específico de hiperparâmetros e/ou algoritmos que resolvam essa tarefa.
- Para o conjunto de dados, algoritmos e hiperparâmetros utilizados, a métrica de distância euclidiana mostrou, de forma geral, um melhor desempenho, conforme as métricas de avaliação utilizadas, do que a cosseno. Foram buscados artigos que comparacem as métricas de distância utilizadas nesse projeto, porém, apenas um foi encontrado: em (QIAN *et al.*, 2004) foi mostrado por análise teórica e resultados experimentais que, em dados com alta dimensionalidade, as distâncias cosseno e euclidiana apresentam resultados similares e, em dados normalizados e agrupados, essas duas métricas se tornam mais similares. Esse mesmo resultado foi alcançado em (VADIVEL; MAJUMDAR; SURAL, 2003). Em (SINWAR; KAUSHIK, 2014), os resultados experimentais dos autores mostraram que a distância euclidiana obteve um melhor resultado quando comparado com a distância manhattan.

[8] O Apêndice D detalha quais são as *personas* existentes no conjunto de dados usado.

- De forma geral, foi percebido que o algoritmo *K-Means*, segundo as métricas testadas, tanto de distância quanto de avaliação, sobre o conjunto de dados utilizado, obteve resultados mais safisfatórios. Dessa forma, uma estratégia interessante é a de utilizá-lo primeiro para obter maior conhecimento sobre os dados para, só então, partir para o uso de outros algoritmos.
- Apenas remover as *stop words*, sem aplicar sobre o conjunto de dados nenhuma técnica de *lemmatização* ou *stemmização*, apresentou resultados bastante satisfatórios em alguns algoritmos, sendo algo muito apreciado, visto que remove complexidade do processo de agrupamento.
- Por fim, as métricas de avaliação, de forma geral, não são diretamente proporcionais entre si, dado que, quando uma métrica de avaliação atinge um alto valor, as demais não necessariamente apresentam o mesmo comportamento.

Algoritmo	**Métrica de Distância**	**Coeficiente de *Silhouette***	**Índice de Calinski-Harabasz**	**Índice de Davies-Bouldin**	**Tempo de Execução**
DBSCAN	Euclidiana	0.760848	36.264652	1.005990	0.025091 seg
GMM	Mahalanobis	0.073631	2.036996	1.695388	0.422684 seg
Hierárquico Aglomerativo	Euclidiana	0.564175	97.825598	0.836707	0.020472 seg
K-Means	Cosseno	0.863832	824.547571	0.215420	13.783269 seg

Tabela 4 – Resultados do Primeiro Experimento

4.5.2 Agrupamento por desejos

Visto que, com o experimento inicial, não foi possível agrupar os dados por *personas* e desejos, foi utilizada uma nova abordagem, que consistiu em, primeiramente, aplicar uma função de *regex* para separar os dados conforme a categoria de *persona*, para então, em cada categoria de *persona* encontrada, reaplicar os testes realizados no primeiro experimento.

Como nessa segunda etapa de testes os dados a serem agrupados possuiam tamanhos diferentes, muitos sendo bastante pequenos[9], foi possível realizar a categorização manual de alguns grupos, permitindo uma verificação mais precisa de alguns agrupamentos. Há de ser mencionado, entretanto, que essa categorização foi realizado pelo próprio autor e, como a mesma não foi revisada por outros, pode existir um viés nos dados.

A Tabela 5 apresenta os melhores resultados obtidos por algoritmo, sendo que o Apêndice H detalha e analisa os resultados individuais de cada um. Nessa etapa de testes, as conclusões gerais foram:

[9] Menos de 40 histórias.

- As métricas de avaliação internas nos grupos de *personas* manualmente rotulados, se mostraram insuficientes para a correta avaliação dos grupos gerados, visto que, mesmo quando o modelo conseguia agrupar com elevada precisão os dados, essas métricas apresentavam valores insatisfatórios. Porém, é importante reforçar novamente que, como os dados foram categorizados pelo autor, e não foram revisados por outros, esse pode ser um erro de metodologia e não um erro no uso das métricas.
- Ao contrário dos resultados do primeiro experimento, neste a métrica cosseno apresentou resultados mais próximos aos da euclidiana e, em alguns casos, melhores. Isso provavelmente ocorreu devido ao número reduzido de histórias de usuários processadas, consequentemente reduzindo a dimensionalidade dos dados.
- Os algoritmos Hierárquico Aglomerativo e GMM apresentaram os melhores resultados nas métricas de validação externas, seguidos pelo algoritmo DBSCAN e, por último, o *K-Means*, que obteve os piores resultados.
- Apesar de o DBSCAN não ter apresentado os melhores resultados nas métricas de avaliação externas, o mesmo não é verdade no que tange as métricas de avaliação internas, o que leva o autor a crer que, de modo geral, esse algoritmo se adequa melhor ao conjunto de dados testado.
- De forma geral, apesar de terem sido testadas diferentes formas de pré-processar os textos, as que dominaram essa bateria de testes foram:
  - apenas remover *stop words*;
  - remover *stop words* e aplicar *stemmização* e
  - remover *stop words*, aplicar *lemmatização* e *stemmização*.

Algoritmo	**Métrica de Distância**	**Coeficiente de *Silhouette***	**Índice de Calinski-Harabasz**	**Índice de Davies-Bouldin**	**Homogeneidade**	**Completude**	***V-Measure***	**Índice de Rand Ajustado**	**Informação Mútua Ajustada**	**Tempo de Execução**
DBSCAN	Cosseno	0.43873	4.856284	1.56226	0.752982	0.760254	0.756601	0.557313	0.578222	0.006761 seg
GMM	Mahalanobis	0.339904	4.894808	1.624545	0.817083	0.832559	0.824748	0.644831	0.702807	0.003535 seg
Hierárquico Aglomerativo	Manhattan	0.407233	6.062793	0.738213	0.949886	0.759364	0.844007	0.646569	0.656034	0.000322 seg
K-Means	Cosseno	0.094928	3.749833	2.840545	0.299127	1.0	0.460504	0.200745	0.395783	0.121665 seg

Tabela 5 – Resultados do Segundo Experimento

4.6 INTERFACE *WEB*

Uma vez que os experimentos foram concluídos, iniciou-se o desenvolvimento da *interface web*. Para tal, foi utilizado o *framework Dash*, partindo das premisas iniciais:

- Buscando criar uma ferramenta que possa ser utilizada de forma genérica, e tendo em mente os resultados obtidos nas fases de análise exploratória e experimentos, foram adicionados três principais módulos na ferramenta:
    1. Um breve módulo para realizar uma análise exploratória dos dados;
    2. Um módulo para agrupar todos os dados, sem separá-los por *personas*;
    3. Um módulo para exibir os dados agrupados por categoria de *persona*.
- O segundo módulo permite ao usuário escolhar diferentes formas de pré-processar os textos, os hiperparâmetros do TF-IDF, o algoritmo de agrupamento e seus hiperparâmetros e a forma de visualização, tanto em 2D quanto em 3D, sendo que, utilizando o *Dash*, foi possível disponibilizar uma *interface* interativa com o gráfico de dispersão gerado e
- Por fim, o arquivo de entrada a ser processado em todos os módulos deve ser CSV válido, com cada linha contendo uma história de usuário. O nome do *header* deve ser *text*, e histórias de usuários incorretas serão automáticamente descartadas.

A Figura 18 exibe o protótipo inicial da *interface*, que, apesar de simplista, se mostrou útil para visualizar os agrupamentos e obter *insights* dos dados. Ainda sobre essa interface, a Figura 19 exibe o módulo de escolha das histórias de usuários, a Figura 20 exibe a análise exploratória dos dados, a Figura 21 exibe as formas de pré-processar os dados, a Figura 22 exibe a opção de vetorização dos textos. A Figura 23 exibe as histórias em forma de gráfico, porém sem aplicar qualquer tipo de agrupamento. A Figura 24 exibe o resultado do agrupamento dos dados e, por fim, a Figura 25 exibe o agrupamento por *personas*.

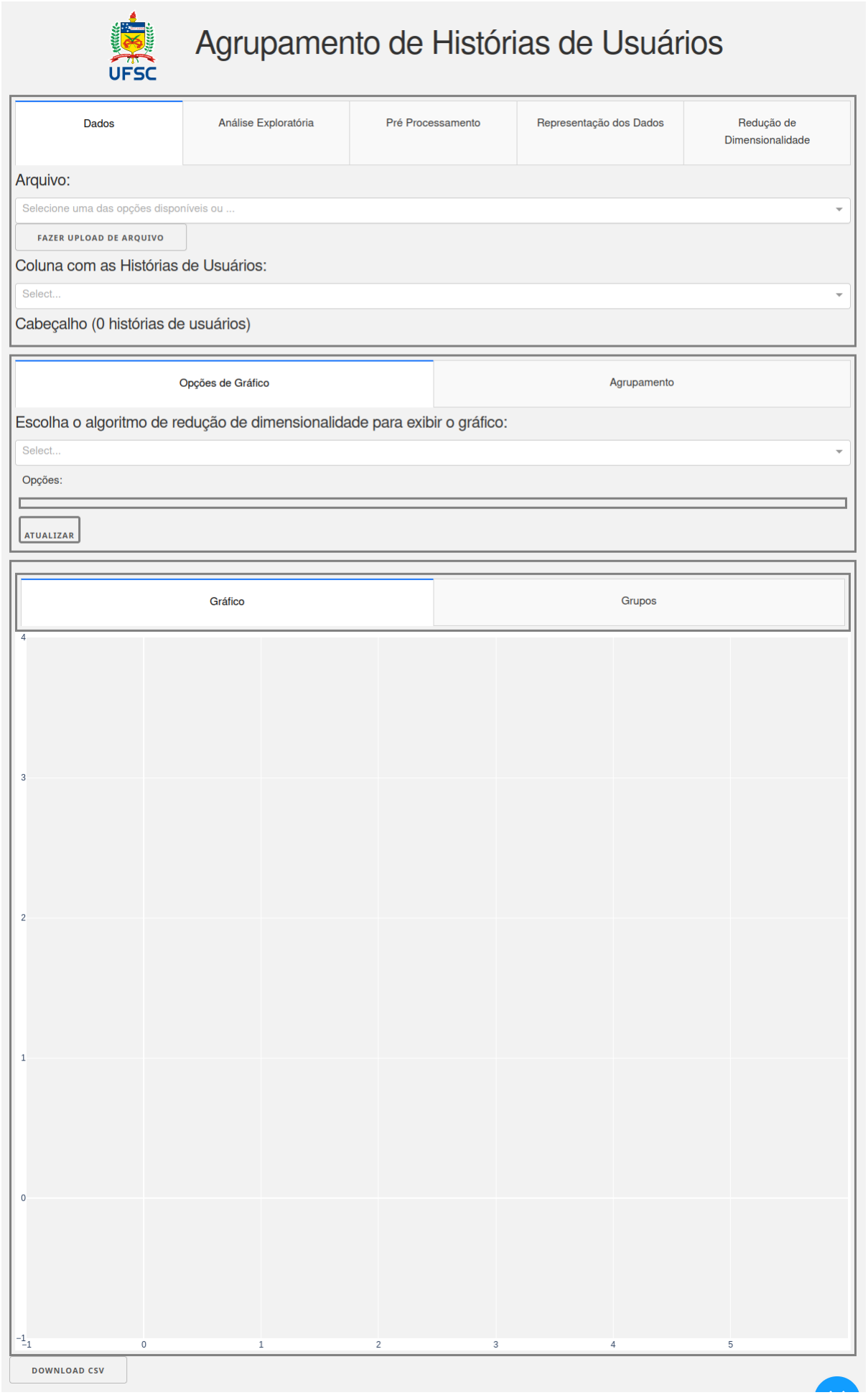

Figura 18 – Visão geral da interface

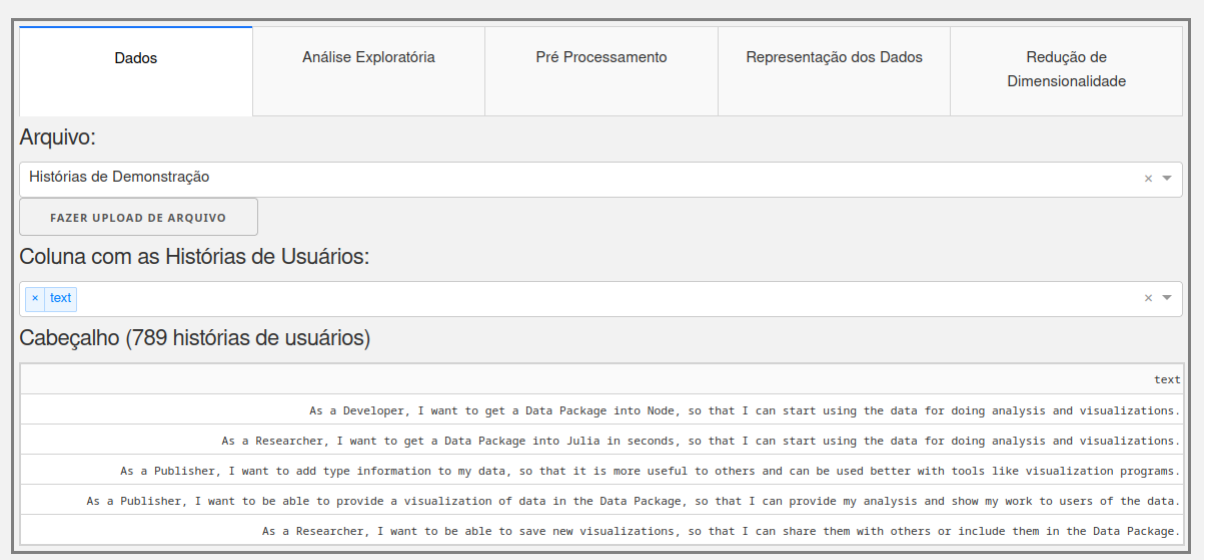

Figura 19 – Escolha dos dados para agrupar

Figura 20 – Análise exploratória

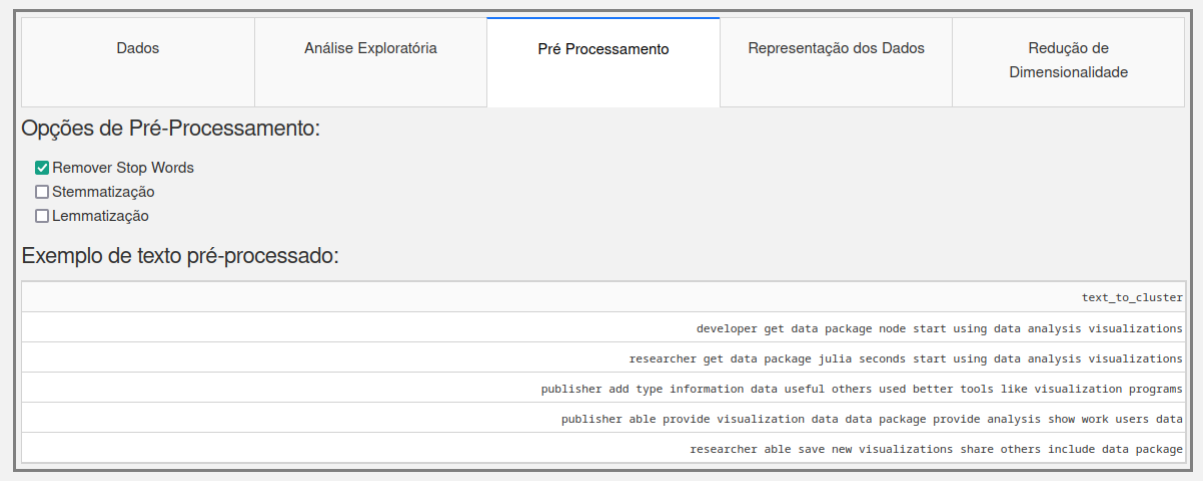

Figura 21 – Opções de pré-processamento

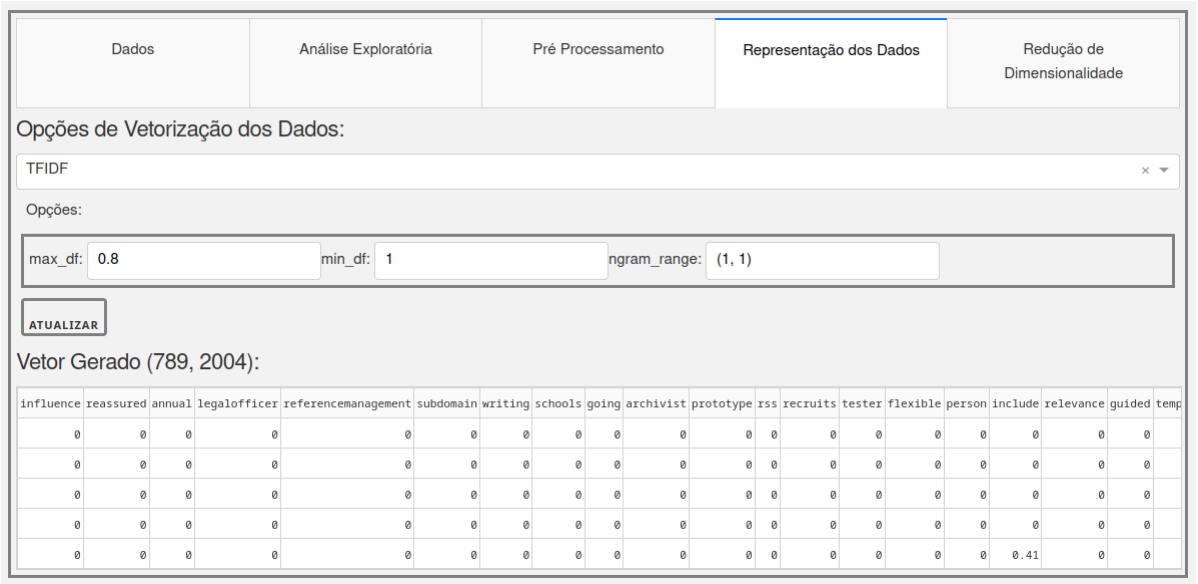

Figura 22 – Representação vetorial dos textos

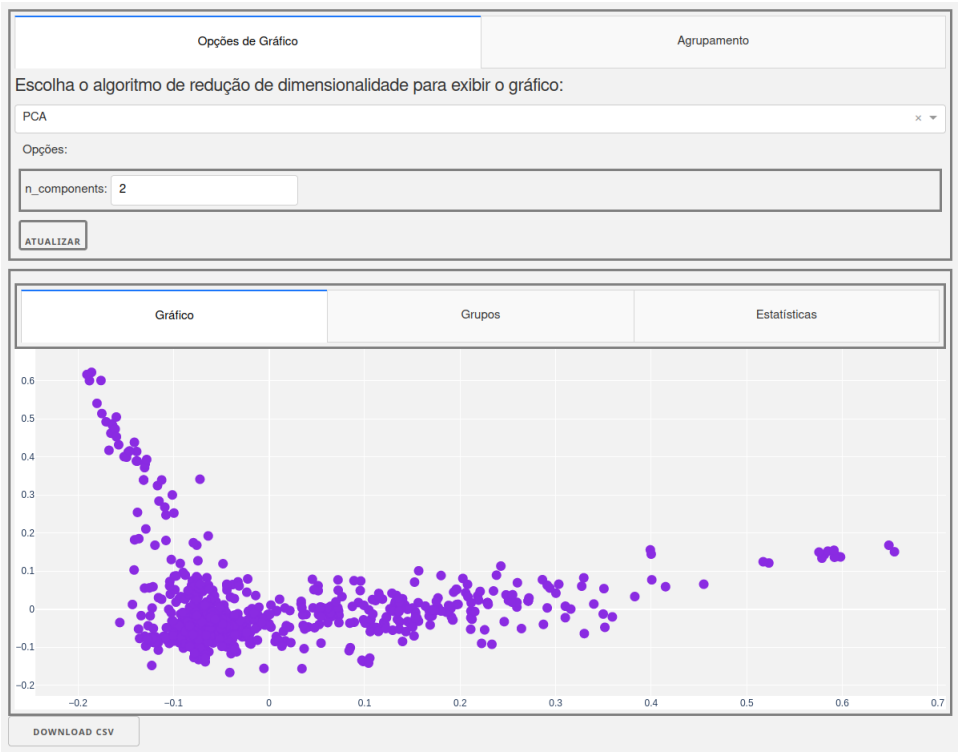

Figura 23 – Apenas visualizar graficamente os dados

Figura 24 – Executar algoritmo de agrupamento

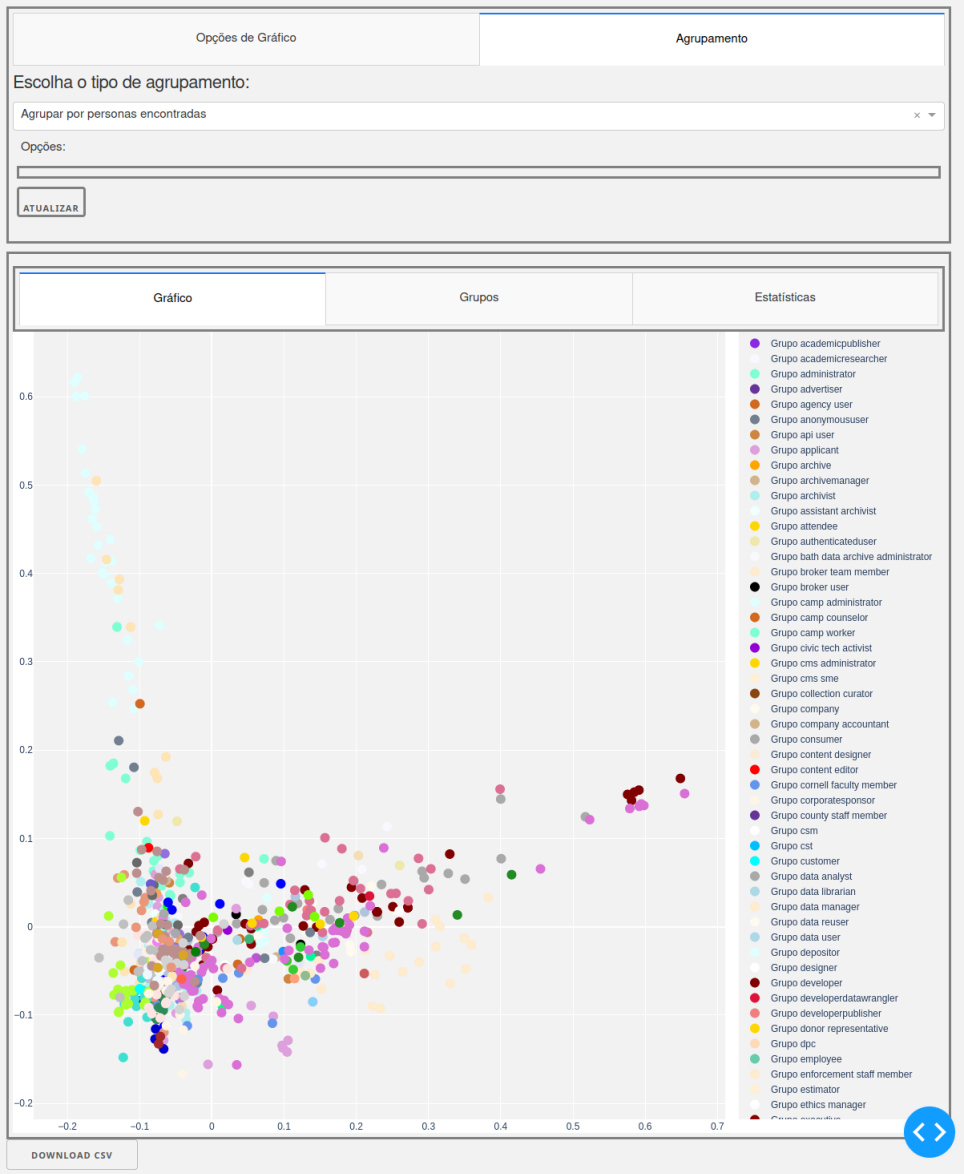

Figura 25 – Agrupar por personas

4.6.1 Avaliação de Usabilidade

Para (NIELSEN *et al.*, 2012) usabilidade é definida como um atributo de qualidade, responsável por determinar a facilidade de uso de determinada interface de usuário. Segundo (RUBIN; CHISNELL, 2008) teste de usabilidade é o processo no qual participantes representativos de determinado grupo alvo avaliam o grau de acerto que determinado produto possuí em relação à critérios específicos de usabilidade, sendo que tal avaliação pode utilizar parâmetros como a eficácia, eficiência e satisfação. De acordo com (NIELSEN, 2012), se o objetivo do teste é o de entender a utilização de uma interface, visando melhorar seu *design*, 2 pessoas é o número ideal para projetos de baixo custo.

Dessa forma, visando avaliar o protótipo desenvolvido neste projeto, foi realizado um teste de usabilidade com pessoas inseridas no contexto de desenvolvimento de

software. Para tal teste, foi utilizada a escala SUS (*System Usability Scale*) (BROOKE *et al.*, 1996), visando avaliar a efetividade[10], eficiência[11] e satisfação[12] no uso do protótipo. O SUS é aplicado através de um questionário com 10 perguntas, cujas respostas podem ser de 1 (discordo totalmente) à 5 (concordo totalmente). Além disso, foi disponibilizada a oportunidade dos participantes de realizarem comentários sobre o protótipo.

Foram criadas 8 atividades para serem avaliadas no teste, sendo que foi definido a porcentagem e tempo esperados de conclusão de cada atividade, detalhadas na Tabela 6. Aqui é importante ressaltar que o tempo mínimo esperado desconsidera o tempo necessário para realizar algum processamento computacional como, por exemplo, reduzir a dimensionalidade dos dados.

Número	**Atividade**	**Eficácia**	**Eficiência**
1	Selecionar um conjunto de histórias de usuários	2/2	10 segundos
2	Selecionar a coluna que contém as histórias de usuários	2/2	5 segundos
3	Selecionar uma ou mais formas de pré-processar os dados e analisar o resultado	2/2	20 segundos
4	Selecionar uma forma de representação vetorial dos textos e analisar o vetor gerado	2/2	10 segundos
5	Selecionar uma forma de representação gráfica dos dados e inserir os hiperparâmetros necessários	2/2	20 segundos
6	Selecionar uma forma de agrupamento e inserir os hiperparâmetros necessários	2/2	20 segundos
7	Navegar pelo gráfico gerado	2/2	10 segundos
8	Fazer o *download* do arquivo CSV com os dados agrupados	2/2	10 segundos

Tabela 6 – Atividades realizadas

Para a aplicação do teste, dois participantes foram selecionados, sendo que ambos são graduados do curso de Sistemas de Informação e atuam como desenvolvedores de *software fullstack*, do gênero masculino, e possuem 23 e 24 anos respectivamente. Para a realização do teste, foi realizado um encontro presencial na Biblioteca Universitário, onde o testador descreveu para ambos o contexto e uso do protótipo.

[10] Sucesso no uso do protótipo.
[11] Esforço necessário para utilizar o protótipo.
[12] Valor subjetivo do usuário sobre o quão agradável foi a utilização do protótipo.

Além disso, foram disponibilizados dois conjuntos de dados de histórias de usuários para o teste sendo:

- Conjunto de dados já embarcado no próprio protótipo, descrito em 4.2.
- Conjunto de dados derivado do primeiro, mas com um numéro reduzido de histórias de usuários.

Antes de iniciar os testes, foi recomendado que os participantes se sentissem livres para realizar demais atividades, caso quisessem, assim como, a qualquer momento, expressar críticas/sugestões sobre o protótipo. A Tabela 7 apresenta os resultados obtidos nesse teste, onde os valores em vermelho indicam caso o resultado esperado não foi alcançado, e os valores em verde o contrário. A Tabela 8 apresenta o resultado do questionário SUS e, por fim, a Tabela 9 apresenta as sugestões/críticas sintetizadas dos usuários.

Atividade	**Participante**	**Concluiu atividade**	**Tempo**
1	1	Sim	8 segundos
	2	Sim	3 segundos
2	1	Sim	1 segundos
	2	Sim	2 segundos
3	1	Sim	9 segundos
	2	Sim	14 segundos
4	1	Sim	1 segundos
	2	Sim	3 segundos
5	1	Sim	11 segundos
	2	Sim	16 segundos
6	1	Sim	13 segundos
	2	Sim	19 segundos
7	1	Sim	2 segundos
	2	Sim	7 segundos
8	1	Sim	3 segundos
	2	Sim	4 segundos

Tabela 7 – Resultados do teste

Participante	**Pontuação SUS**
1	90 pontos
2	70 pontos
Média	80 pontos

Tabela 8 – Resultado do questionário SUS

Participante	**Sugestão/crítica**
1	A interface poderia ter mais cores
2	Os componentes da interface poderiam ter um botão de ajuda, para explicar o que cada um deles é e faz
	Na opção de **Representação dos Dados** só existe uma opção, então não faz sentido ter um *Select box* lá
	Seria interessante ter um *link* no cabeçalho da página para redirecionar para o projeto no *GitHub*
	Alguns botões ficariam melhores se alinhados no centro da página ao invés de no canto esquerdo

Tabela 9 – Sugestões dos participantes

Apesar de o teste em si ter apresentado resultados extremamente satisfatórios, a amostragem usada pode ter sido muito pequena para uma avaliação real da usabilidade do protótipo, assim como o vínculo dos participantes com o testador pode ter influenciado os resultados.

5 CONCLUSÕES

Neste trabalho, foi investigada a efetividade de agrupar, de forma não supervisionada, histórias de usuários geradas como requisitos de *software*, utilizando diferentes abordagens. A partir dessa investigação, a ferramenta *web Natasha* foi proposta, visando automatizar o processo de agrupamento de histórias de usuários. Durante seu desenvolvimento, foi perceptível que a etapa de pré-processamento exigiu um esforço considerável para garantir a qualidade dos dados.

Os resultados preliminares mostraram que, independente da abordagem utilizada no conjunto de dados de teste, não foi possível agrupar, de forma satisfatória, as histórias de usuários por categoria de *persona* e desejo.

Finalmente, com relação à *interface web* desenvolvida, foi percebida uma grande utilidade da mesma na etapa de análise exploratória dos dados, especialmente quando o usuário não possuí conhecimento dos grupos existentes nas histórias de usuários, assim como para visualizar, de forma interativa, os possíveis grupos existentes nos dados, sobre a ótica de diferentes algoritmos de agrupamento.

5.1 SUGESTÕES E TRABALHOS FUTUROS

Durante o desenvolvimento deste projeto, alguns pontos não foram abordados, visto que nem todos os problemas podem ser resolvidos de uma única vez. Alguns dos pontos não abordados, assim como sugestões de trabalhos futuros, são:

- Uso de outros algoritmos de agrupamento, como *SpectralClustering* ou *HDBSCAN*;
- Uso de técnicas modernas de processamento de linguagem natural, e outros modelos de linguagem e grafos de conhecimento, na etapa de pré-processamento dos textos;
- Utilização de outras bibliotecas de *stop words*, como a *Gensim* e *spaCy*;
- Uso de outros conjuntos de dados, incluindo histórias de usuários em outros idiomas[1];
- Analisar, de forma precisa, o desempenho dos algoritmos aqui testados, em questão de uso de recursos computacionais;
- Permitir dados de entrada com outros *encodings* que não o UTF-8;
- Criar outras formas de visualização de grupos, como o uso de dendogramas;
- Adicionar mais informações estatísticas sobre os grupos encontrados, como o grau médio de similaridade entre as histórias de um grupo, distâncias máximas e mínimas entre grupos, entre outros;

[1] Português brasileiro, por exemplo.

- Testar agrupar as *personas* encontradas em grupos mais gerais de categorias de *persona* como, por exemplo, agrupar as *personas developer*, *it manager*, *tester* e *web developer* em um mesmo grupo chamado *it-people*;
- Testar outras métricas de avaliação, como as propostas em (PALACIO-NIÑO; BERZAL, 2019).

REFERÊNCIAS

AMBREEN, Talat; IKRAM, Naveed; USMAN, Muhammad; NIAZI, Mahmood. Empirical research in requirements engineering: trends and opportunities. **Requirements Engineering**, Springer, v. 23, n. 1, p. 63–95, 2018.

AMINE, Abdelmalek; ELBERRICHI, Zakaria; SIMONET, Michel. Evaluation of text clustering methods using wordnet. **Int. Arab J. Inf. Technol.**, v. 7, n. 4, p. 349–357, 2010.

BAEZA-YATES, Ricardo; RIBEIRO-NETO, Berthier *et al.* **Modern information retrieval**. [*S.l.*]: ACM press New York, 1999. v. 463.

BAKER, L Douglas; MCCALLUM, Andrew Kachites. Distributional clustering of words for text classification. *In*: PROCEEDINGS of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. [*S.l.*: *s.n.*], 1998. P. 96–103.

BARBOSA, Ricardo; JANUARIO, Daniele; SILVA, Ana Estela; MORAES, Regina; MARTINS, Paulo. An approach to clustering and sequencing of textual requirements. *In*: IEEE. 2015 IEEE International Conference on Dependable Systems and Networks Workshops. [*S.l.*: *s.n.*], 2015. P. 39–44.

BEKKERMAN, Ron; EL-YANIV, Ran; TISHBY, Naftali; WINTER, Yoad. On feature distributional clustering for text categorization. *In*: PROCEEDINGS of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. [*S.l.*: *s.n.*], 2001. P. 146–153.

BILMES, Jeff A *et al.* A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. **International computer science institute**, Berkeley, CA, v. 4, n. 510, p. 126, 1998.

BOURQUE, Pierre; DUPUIS, Robert; ABRAN, Alain; MOORE, James W; TRIPP, Leonard. The guide to the software engineering body of knowledge. **IEEE software**, IEEE, v. 16, n. 6, p. 35–44, 1999.

BROOKE, John *et al.* SUS-A quick and dirty usability scale. **Usability evaluation in industry**, London–, v. 189, n. 194, p. 4–7, 1996.

CALIŃSKI, Tadeusz; HARABASZ, Jerzy. A dendrite method for cluster analysis. **Communications in Statistics-theory and Methods**, Taylor & Francis, v. 3, n. 1, p. 1–27, 1974.

CUTTING, Douglass R; KARGER, David R; PEDERSEN, Jan O; TUKEY, John W. Scatter/gather: A cluster-based approach to browsing large document collections. *In*: ACM NEW YORK, NY, USA, 2. ACM SIGIR Forum. [*S.l.*: *s.n.*], 2017. v. 51, p. 148–159.

DALPIAZ, Fabiano; VAN DER SCHALK, Ivor; BRINKKEMPER, Sjaak; AYDEMIR, Fatma Başak; LUCASSEN, Garm. Detecting terminological ambiguity in user stories: Tool and experimentation. **Information and Software Technology**, Elsevier, v. 110, p. 3–16, 2019.

DAVIES, David L; BOULDIN, Donald W. A cluster separation measure. **IEEE transactions on pattern analysis and machine intelligence**, IEEE, n. 2, p. 224–227, 1979.

DIFFERENCE between PCA vs T-Sne. [*S.l.*: *s.n.*], jan. 2022. Disponível em: `https://www.geeksforgeeks.org/difference-between-pca-vs-t-sne/`. Acesso em: 1 jul. 2022.

ESTER, Martin; KRIEGEL, Hans-Peter; SANDER, Jörg; XU, Xiaowei *et al.* A density-based algorithm for discovering clusters in large spatial databases with noise. *In*: 34. KDD. [*S.l.*: *s.n.*], 1996. v. 96, p. 226–231.

EYAL SALMAN, Hamzeh; HAMMAD, Mustafa; SERIAI, Abdelhak-Djamel; AL-SBOU, Ahed. Semantic clustering of functional requirements using agglomerative hierarchical clustering. **Information**, MDPI, v. 9, n. 9, p. 222, 2018.

FAHAD, Adil; ALSHATRI, Najlaa; TARI, Zahir; ALAMRI, Abdullah; KHALIL, Ibrahim; ZOMAYA, Albert Y; FOUFOU, Sebti; BOURAS, Abdelaziz. A survey of clustering algorithms for big data: Taxonomy and empirical analysis. **IEEE transactions on emerging topics in computing**, IEEE, v. 2, n. 3, p. 267–279, 2014.

FELDMAN, Ronen; DAGAN, Ido. Knowledge Discovery in Textual Databases (KDT). *In*: KDD. [*S.l.*: *s.n.*], 1995. v. 95, p. 112–117.

FUKUNAGA, Keinosuke. **Introduction to statistical pattern recognition**. [*S.l.*]: Elsevier, 2013.

GAN, Guojun; MA, Chaoqun; WU, Jianhong. **Data clustering: theory, algorithms, and applications**. [*S.l.*]: SIAM, 2020.

HAIR, Joseph F. Multivariate data analysis, 2009.

HALKIDI, Maria; BATISTAKIS, Yannis; VAZIRGIANNIS, Michalis. On clustering validation techniques. **Journal of intelligent information systems**, Springer, v. 17, n. 2, p. 107–145, 2001.

HAMMER, PC. **Adaptive control processes: a guided tour (R. Bellman)**. [*S.l.*]: Society for Industrial e Applied Mathematics, 1962.

HAN, Jiawei; PEI, Jian; KAMBER, Micheline. **Data mining: concepts and techniques**. [*S.l.*]: Elsevier, 2011.

HOTHO, Andreas; NÜRNBERGER, Andreas; PAASS, Gerhard. A brief survey of text mining. *In*: CITESEER, 1. LDV Forum. [*S.l.*: *s.n.*], 2005. v. 20, p. 19–62.

JAIN, Anil K; DUBES, Richard C. **Algorithms for clustering data**. [*S.l.*]: Prentice-Hall, Inc., 1988.

JAIN, Anil K; MURTY, M Narasimha; FLYNN, Patrick J. Data clustering: a review. **ACM computing surveys (CSUR)**, Acm New York, NY, USA, v. 31, n. 3, p. 264–323, 1999.

JIMENEZ, Luis O; LANDGREBE, David A. Supervised classification in high-dimensional space: geometrical, statistical, and asymptotical properties of multivariate data. **IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)**, IEEE, v. 28, n. 1, p. 39–54, 1998.

JONES, Karen Sparck. A statistical interpretation of term specificity and its application in retrieval. **Journal of documentation**, MCB UP Ltd, 1972.

KASSAB, Mohamad. The Changing Landscape of Requirements Engineering Practices Over The Past Decade. *In*.

KAUFMAN, Leonard; ROUSSEEUW, Peter J. **Finding groups in data: an introduction to cluster analysis**. [*S.l.*]: John Wiley & Sons, 2009.

KING, Ronald S. **Cluster analysis and data mining: An introduction**. [*S.l.*]: Mercury Learning e Information, 2015.

KONRAD, Sascha; GALL, Michael. Requirements engineering in the development of large-scale systems. *In*: IEEE. 2008 16th IEEE International Requirements Engineering Conference. [*S.l.*: *s.n.*], 2008. P. 217–222.

LINDVALL, Mikael; BASILI, Vic; BOEHM, Barry; COSTA, Patricia; DANGLE, Kathleen; SHULL, Forrest; TESORIERO, Roseanne; WILLIAMS, Laurie; ZELKOWITZ, Marvin. Empirical findings in agile methods. *In*: SPRINGER. CONFERENCE on extreme programming and agile methods. [*S.l.*: *s.n.*], 2002. P. 197–207.

LINDVALL, Torgny. **Lectures on the coupling method**. [*S.l.*]: Courier Corporation, 2002.

LOVINS, Julie Beth. Development of a stemming algorithm. **Mech. Transl. Comput. Linguistics**, v. 11, n. 1-2, p. 22–31, 1968.

LUCASSEN, Garm; DALPIAZ, Fabiano; WERF, Jan Martijn EM van der; BRINKKEMPER, Sjaak. Improving agile requirements: the quality user story framework and tool. **Requirements engineering**, Springer, v. 21, n. 3, p. 383–403, 2016.

LUCASSEN, Garm; DALPIAZ, Fabiano; WERF, Jan Martijn EM van der; BRINKKEMPER, Sjaak. The use and effectiveness of user stories in practice. *In*: SPRINGER. INTERNATIONAL working conference on requirements engineering: Foundation for software quality. [*S.l.*: *s.n.*], 2016. P. 205–222.

LUCASSEN, Garm; DALPIAZ, Fabiano; WERF, Jan Martijn EM van der; BRINKKEMPER, Sjaak. Visualizing user story requirements at multiple granularity levels via semantic relatedness. *In*: SPRINGER. INTERNATIONAL Conference on Conceptual Modeling. [*S.l.*: *s.n.*], 2016. P. 463–478.

LUHN, H. P. A Statistical Approach to Mechanized Encoding and Searching of Literary Information. **IBM Journal of Research and Development**, v. 1, n. 4, p. 309–317, 1957.

MAHALANOBIS, Prasanta Chandra. On the generalized distance in statistics. *In*: NATIONAL INSTITUTE OF SCIENCE OF INDIA.

MANLY, Bryan FJ; ALBERTO, Jorge A Navarro. **Multivariate statistical methods: a primer**. [*S.l.*]: Chapman e Hall/CRC, 2016.

MAO, Jianchang; JAIN, Anil K. A self-organizing network for hyperellipsoidal clustering (HEC). **Ieee transactions on neural networks**, IEEE, v. 7, n. 1, p. 16–29, 1996.

MEDRI, Waldir. Análise exploratória de dados. **Londrina: Universidade Estadual de Londrina**, 2011.

NIELSEN, Jakob. How many test users in a usability study. **Nielsen Norman Group**, v. 4, n. 06, 2012.

NIELSEN, Jakob *et al.* Usability 101: Introduction to usability. Fremont, CA, 2012.

NOEL, Rene; RIQUELME, Fabián; MAC LEAN, Roberto; MERINO, Erick; CECHINEL, Cristian; BARCELOS, Thiago S; VILLARROEL, Rodolfo; MUNOZ, Roberto. Exploring collaborative writing of user stories with multimodal learning analytics: A case study on a software engineering course. **IEEE Access**, IEEE, v. 6, p. 67783–67798, 2018.

PALACIO-NIÑO, Julio-Omar; BERZAL, Fernando. Evaluation metrics for unsupervised learning algorithms. **arXiv preprint arXiv:1905.05667**, 2019.

PEARSON, Karl. LIII. On lines and planes of closest fit to systems of points in space. **The London, Edinburgh, and Dublin philosophical magazine and journal of science**, Taylor & Francis, v. 2, n. 11, p. 559–572, 1901.

QIAN, Gang; SURAL, Shamik; GU, Yuelong; PRAMANIK, Sakti. Similarity between Euclidean and cosine angle distance for nearest neighbor queries. *In*: PROCEEDINGS of the 2004 ACM symposium on Applied computing. [*S.l.*: *s.n.*], 2004. P. 1232–1237.

ROUSSEEUW, Peter J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, Elsevier, v. 20, p. 53–65, 1987.

RUBIN, Jeffrey; CHISNELL, Dana. **Handbook of usability testing: how to plan, design and conduct effective tests**. [*S.l.*]: John Wiley & Sons, 2008.

SAIF, Hassan; FERNÁNDEZ, Miriam; HE, Yulan; ALANI, Harith. On stopwords, filtering and data sparsity for sentiment analysis of twitter, 2014.

SCHÖN, Eva-Maria; THOMASCHEWSKI, Jörg; ESCALONA, Marıéa José. Agile Requirements Engineering: A systematic literature review. **Computer standards & interfaces**, Elsevier, v. 49, p. 79–91, 2017.

SCHÜTZE, Hinrich; MANNING, Christopher D; RAGHAVAN, Prabhakar. **Introduction to information retrieval**. [*S.l.*]: Cambridge University Press Cambridge, 2008. v. 39, p. 32–33.

SCHWABER, Ken. Scrum development process. *In*: BUSINESS object design and implementation. [*S.l.*]: Springer, 1997. P. 117–134.

SCHWABER, Ken; BEEDLE, Mike. **Agile software development with scrum. Series in agile software development**. [*S.l.*]: Prentice Hall Upper Saddle River, 2002. v. 1.

SHIRKHORSHIDI, Ali Seyed; AGHABOZORGI, Sr; WAH, Teh. A Comparison Study on Similarity and Dissimilarity Measures in Clustering Continuous Data. **PLOS ONE**, v. 10, e0144059, dez. 2015.

SILVA, Catarina; RIBEIRO, Bernardete. The importance of stop word removal on recall values in text categorization. *In*: IEEE. PROCEEDINGS of the International Joint Conference on Neural Networks, 2003. [*S.l.*: *s.n.*], 2003. v. 3, p. 1661–1666.

SINWAR, Deepak; KAUSHIK, Rahul. Study of Euclidean and Manhattan distance metrics using simple k-means clustering. **Int. J. Res. Appl. Sci. Eng. Technol**, v. 2, n. 5, p. 270–274, 2014.

UYSAL, Alper Kursat; GUNAL, Serkan. The impact of preprocessing on text classification. **Information processing & management**, Elsevier, v. 50, n. 1, p. 104–112, 2014.

VADIVEL, AKMSSA; MAJUMDAR, AK; SURAL, Shamik. Performance comparison of distance metrics in content-based image retrieval applications. *In*: INTERNATIONAL

Conference on Information Technology (CIT), Bhubaneswar, India. [*S.l.*: *s.n.*], 2003. P. 159–164.

VAN DER MAATEN, Laurens; HINTON, Geoffrey. Visualizing data using t-SNE. **Journal of machine learning research**, v. 9, n. 11, 2008.

VERNER, June; COX, Karl; BLEISTEIN, Steven; CERPA, Narciso *et al.* Requirements engineering and software project success: an industrial survey in Australia and the US. **Australasian Journal of information systems**, Australian Computer Society, v. 13, n. 1, 2005.

WAUTELET, Yves; HENG, Samedi; KOLP, Manuel; MIRBEL, Isabelle. Unifying and Extending User Story Models. *In*: JARKE, Matthias; MYLOPOULOS, John; QUIX, Christoph; ROLLAND, Colette; MANOLOPOULOS, Yannis; MOURATIDIS, Haralambos; HORKOFF, Jennifer (Ed.). **Advanced Information Systems Engineering**. Cham: Springer International Publishing, 2014. P. 211–225.

WEBSTER, Jonathan J; KIT, Chunyu. Tokenization as the initial phase in NLP. *In*: COLING 1992 volume 4: The 14th international conference on computational linguistics. [*S.l.*: *s.n.*], 1992.

XU, Rui; WUNSCH, Donald. Survey of clustering algorithms. **IEEE Transactions on neural networks**, Ieee, v. 16, n. 3, p. 645–678, 2005.

APÊNDICE A – CÓDIGO-FONTE

O código-fonte do protótipo desenvolvido se encontra no repositório: `https://github.com/SadiJr/natasha`.

APÊNDICE B – ARTIGO

Natasha - Um Sistema de Agrupamento de Histórias de Usuários Por Personas e Desejos

Sadi Jr. D. Jacinto[1]

[1] Departamento de Informática e Estatística

Universidade Federal de Santa Catarina (UFSC) – Florianópolis – SC – Brasil

sadijacinto@gmail.com

Abstract. *The use of agile methodologies in the software development process has become popular in recent decades and, with that, the use of user stories to represent requirements from the users' point of view has also become popular. However, since user stories are written by humans and in natural language, they are prone to several errors, such as incompleteness and inconsistency, in addition to the probable existence of stories that represent the same requirement, but are described in different ways. Detecting such inconsistencies, despite being an easy task for humans, is tedious and, in large sets of user stories, ends up demanding a lot of time and effort. Thus, this project aims to develop a web tool that allows the detection and display of similar user stories in order to facilitate and streamline the software development process. To this end, the K-Means, Agglomerative Clustering, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Gaussian Mixture Models (GMM) algorithms were applied, proving to be useful for exploratory analysis and hypothesis testing.*

Resumo. *O uso de metodologias ágeis no processo de desenvolvimento de software tem se popularizado nas últimas décadas e, com isso, o uso de histórias de usuários para representar os requisitos do ponto de vista dos usuários também se popularizou. Porém, uma vez que histórias de usuários são escritas por seres humanos e em linguagem natural, as mesmas estão propensas a diversos erros, como incompletude e inconsistência, além da provável existência de histórias que representam o mesmo requisito, mas que estão descritas de formas diferentes. Detectar tais inconsistências, apesar de ser uma tarefa fácil para seres humanos, é algo maçante e, em grandes conjuntos de histórias de usuários, acaba exigindo muito tempo e esforço. Assim, este projeto tem como objetivo o desenvolvimento de uma ferramenta web que permita a detecção e exibição de histórias de usuários similares, visando facilitar e agilizar o processo de desenvolvimento de software. Para tal, foram aplicados os algoritmos K-Means, Agglomerative Clustering, DBSCAN e GMM, mostrando-se úteis para uma análise e teste de hipótese exploratórios.*

1. Introdução

Desenvolver *software* é uma tarefa onerosa. Os desenvolvedores não apenas precisam atentar para atender os requisitos e prazos do projeto, como, geralmente, precisam dar suporte e melhorias contínuas ao mesmo, visando atender novos requisitos que surgem dos usuários [Konrad and Gall 2008]. Isso tudo tendo em mente que o entendimento de tais requisitos é essencial para o sucesso de um projeto [Bourque et al. 1999, Ambreen et al. 2018].

Segundo [Lindvall et al. 2002], visando melhorar o desempenho do processo de desenvolvimento de *software*, muitas empresas passaram a adotar metodologias ágeis, como o *Scrum* [Schwaber 1997]. Com a popularização do uso de tais metodologias [Kassab 2015], popularizou-se também o uso de histórias de usuários para representar os requisitos dos usuários [Lindvall 2002, Lucassen et al. 2016].

Porém, com tal popularização, um problema surgiu: conforme a complexidade do *software* aumenta, a quantidade de histórias de usuários aumentam também[Kassab 2015]. Apesar de tais histórias serem textos curtos e simples de serem entendidos, em grande número se tornam difíceis, do ponto de vista humano, de serem análisadas e gerenciadas. Considerando que tais histórias podem apresentar problemas como incompletude, ambiguidade, duplicidade e ocorrência de histórias similares, mas expressas de formas diferentes, o problema se torna mais complexo.

No contexto desse problema, o uso de técnicas de aprendizado não supervisionado se tornam uma possível solução, visto que as mesmas não necessitam que os dados sejam préviamente rotulados. Com base nisso, o presente trabalho objetiva propor uma ferramenta *web* para identificar e apresentar, através do uso de técnicas de agrupamento, histórias de usuários que expressam requisitos similares, consequentemente auxiliando a tomada de decisão no processo de desenvolvimento de *software*.

2. Fundamentação Teórica

2.1. Agrupamento

Agrupamento consiste na implementação de técnicas para encontrar grupos similares em um conjunto de dados, sendo que tal descoberta de grupos é realizada através do uso de alguma função de similaridade. Segundo [Baeza-Yates et al. 1999], tal técnica faz parte do aprendizado de máquina não supervisionado, visto que os dados utilizados não possuem rótulos prévios e o algoritmo não passa por nenhuma fase de treinamento.

Técnicas de agrupamento são muito utilizadas em aplicações de mineração de dados, além de ter diversos estudos que a utilizam também na mineração de textos, visando extrair informações úteis e desconhecidas de textos [Feldman and Dagan 1995]. São exemplos disso os trabalhos de [Baker and McCallum 1998, Bekkerman et al. 2001] sobre classificação de textos, assim como em [Cutting et al. 2017] para organização de documentos, entre outros.

Tal diversidade no uso de algoritmos de agrupamento em textos se dá, em parte, pelo fato que é possível utilizar técnicas de agrupamento em textos de diferentes granularidades, como documentos, parágrafos, frases e palavras. O processo de agrupamento, segundo [Jain et al. 1999, King 2015], se divide nas etapas:

1. **Representação dos Padrões ou Preparação:**
   Envolve o pré-processamento dos dados, podendo envolver normalização, conversão de tipos e *encodings*, redução de atributos, entre outros processos, para adequar os dados ao algoritmo de agrupamento a ser utilizado.
2. **Padrão de Proximidade:**
   Envolve a definição de uma função de proximidade, podendo ser de similaridade ou de dissimilaridade, para determinar se dois objetos fazem parte do mesmo grupo.
3. **Agrupamento:**
   Envolve a aplicação de um algoritmo de agrupamento sobre os dados.
4. **Validação:**
   Envolve validar se o algoritmo de agrupamento escolhido é adequado aos dados.

5. **Interpretação:**
   Envolve avaliar, manualmente, cada grupo gerado no processo de agrupamento, buscando obter informações novas dos dados, além de também permitir avaliações subjetivas dos grupos gerados.

Apesar de, na literatura científica, existirem diversos algoritmos de agrupamento, como o *K-Means*, o Algoritmo Hierárquico Aglomerativo, o DBSCAN e o GMM, é de consenso que não existe um algoritmo de agrupamento universal. Assim, cada problema e cada conjunto de dados específicos requerem um ou outro algoritmo.

2.2. Métricas de Distância

São funções de proximidade, podendo ser de similaridade ou dissimilaridade, usadas para determinar se dois objetos distintos fazem parte do mesmo grupo. São exemplos de métricas de distância a euclidiana, a manhattan e a coseno.

2.3. Métricas de Avaliação

Segundo [Halkidi et al. 2001] e [Jain and Dubes 1988], é possível avaliar agrupamentos através de três índices ou critérios:

1. Externos:
   Quando existem valores de referência externos, que podem ser utilizados como resultado esperado, é possível avaliar os grupos gerados com métricas semelhantes às usadas no aprendizado supervisionado. São exemplos de métricas dessa categoria a homogeneidade, completude, *V-Measure*, Índice de Rand Ajustado e Informação Mútua Ajustada [Shirkhorshidi et al. 2015].
2. Internos:
   Quando não existem referências externas para avaliação, é possível usar métricas que avaliam os agrupamentos sobre eles próprios, aplicando algum índice que avalia a compatibilidade da estrututura dos grupos ao medir, por exemplo, a densidade dos grupos gerados, ou a distância mínima entre esses grupos, entre outros. São exemplos de métricas desse tipo o Coeficiente de *Silhouette*, o Índice de Calinski-Harabasz e o Índice de Davies-Bouldin [Rousseeuw 1987, Caliński and Harabasz 1974, Davies and Bouldin 1979].
3. Relativos:
   Através da comparação entre vários agrupamentos, decide-se qual é o melhor com base em algum critério pré-definido (estabilidade, adequação aos dados, tempo de processamento, entre outros). Utilizado, geralmente, para comparar diferentes algoritmos de agrupamento, assim como determinar o valor mais apropriado de algum parâmetro do algoritmo aplicado.

2.4. Redução de Dimensionalidade

Consiste na redução de atributos em dados com muitas dimensões, mas mantendo o número mínimo necessário de parâmetros para representar as propriedades observadas nos dados originais [Fukunaga 2013]. As diferentes técnicas para a redução da dimensionalidade podem ser classificadas como lineares e não lineares, sendo o *Principal Component Analysis* (PCA) [Pearson 1901] e o *t-Distributed Stochastic Neighbor Embedding* (t-SNE) [Van der Maaten and Hinton 2008], exemplos dos mesmos, respectivamente.

2.5. Histórias de Usuários

Histórias de usuários são artefatos consistindo em pequenos textos, com um formato semi-estruturado, que representam requisitos de usuários [Schön et al. 2017, Noel et al. 2018]. De acordo com [Wautelet et al. 2014], tais artefatos possuem a seguinte estrutura:

As [persona], I want/want to/need/can/would like [o quê], so [por que]., onde:

- *persona*: Indica qual o tipo de usuário ao qual a história se refere;
- *o quê*: Indica qual o desejo desse usuário/*persona* e
- *por que*: Indica o motivo do desejo de tal usuário/*persona*.

Apesar de tais artefatos serem facilmente compreendidos por seres humanos, por serem escritos em linguagem natural, tais artefatos não são tão facilmente compreendidos por computados. Um dos desafios que surge, especialmente em grandes organizações, é a ocorrência de histórias de usuários que expressam o mesmo requisito, mas que estão escritas de forma diferentes que, em grandes quantidades, se tornam inviáveis, do ponto de vista humano, de serem corrigidas. Sendo uma das soluções possíveis para esse problema o uso de técnicas de agrupamento de textos, visando agrupar automaticamente histórias de usuários semelhantes e, com isso, facilitar a análise das mesmas.

3. *Natasha*

O desenvolvimento da proposta de agrupamento desse projeto foi dividido em 5 etapas, *vide* Figura 1, sendo as mesmas:

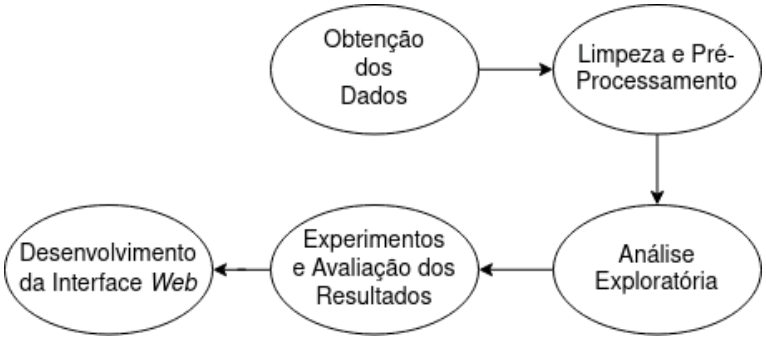

Figura 1. Etapas do Desenvolvimento

1. **Obtenção dos Dados:**
   O conjunto de dados utilizado foi obtido na plataforma *Mendeley Data*, podendo ser acessado através do *link* https://data.mendeley.com/datasets/7zbk8zsd8y/1, e usado originalmente em [Dalpiaz et al. 2019] para experimentos de detecção de ambiguidades em histórias de usuários. Consiste em um conjunto de 22 arquivos, com cada arquivo possuindo mais de 50 histórias de usuários, sendo todas elas na língua inglesa, existindo, ao total 1721 linhas.
2. **Limpeza e Pré-Processamento:**
   Visando remover ruídos e dados inconsistentes que possam atrapalhar o resultado final, foi realizada uma limpeza e pré-processamento dos dados:
   - Converter a codificação de todos os arquivos para UTF-8, concatenar todos os arquivos em um único, remover linhas vazias e converter o arquivo final para CSV;
   - Verificar e remover histórias de usuários com *encoding* errado;
   - Verificar a ocorrência de contrações e as descontrair;
   - Verificar e remover a existência de valores duplicados;
   - Verificar a existência de histórias de usuários erradas, que não seguem o padrão descrito na Seção 2.5 e
   - Renomear manualmente certas ocorrências de *personas* iguais com nomes diferentes.

Após essa etapa de análise e pré-processamento, o conjunto de dados foi reduzido de 1721 linhas para 789 histórias de usuários, existindo 135 categorias de *personas* nos dados.

3. **Análise Exploratória:**
   Análise exploratória envolve o uso de técnicas estatísticas, gráficas e quantitativas para ser possível entender melhor a natureza dos dados [MEDRI 2011]. Para realizar essa análise, foi verificado a distribuição das histórias de usuários em várias formas, assim como as palavras mais frequentes no conjunto de dados, e a matriz de distâncias entre as histórias usando as métricas de distância euclidiana, manhattan e coseno.
4. **Experimentos e Avaliação dos Resultados:**
   Para os experimentos realizados, os algoritmos utilizados foram o *KMeans*, o Hierárquico Aglomerativo, o DBSCAN e o GMM. Para a representação vetorial dos dados, foi utilizado o *Term Frequency - Inverse Document Frequency* (TF-IDF). Para a visualização dos dados, foram utilizados os métodos de redução de dimensionalidade PCA e t-SNE. Foram testados diferentes hiperparâmetros para os algoritmos de agrupamento e para o TF-IDF, assim como formas de pré-processar os dados. De forma geral, os experimentos podem ser divididos em duas grandes fases:

   (a) Verificar agrupamento de todo o conjunto de dados:
   A primeira fase buscou descobrir se os algoritmos de agrupamento selecionados possuíam a capacidade de agrupar, corretamente, as histórias de usuários em grupos baseados na *persona* e no desejo, de forma que desejos semelhantes, mas de *personas* diferentes, sejam classificados em grupos separados, enquanto que desejos semelhantes de *personas* iguais sejam agrupados. Porém, nenhum dessas combinações obteve resultados satisfatórios. Nessa etapa de testes, as principais conclusões foram:
   A Tabela 1 apresenta os melhores resultados obtidos por algoritmo. Nessa etapa de testes, as conclusões gerais foram:

   - A métrica de avaliação *silhouette*, com algumas combinações de algoritmos e hiperparâmetros, atingia seu valor máximo (1), porém, os grupos gerados não faziam o menor sentido do ponto de vista humano. Assim, uma das filtragens dos resultados obtidos consistiu em filtrar os resultados por essa métrica usando um valor abaixo de 1;
   - Aplicar a redução de dimensionalidade nos dados, antes de agrupá-los, aumentou consideravelmente o valor das métricas de avaliação dos grupos, porém, por uma análise humana dos grupos gerados, foi constatado que os mesmos, do ponto de vista humano, não tinham significado, assim, apesar de terem sido realizados testes usando os dados reduzidos, os mesmos, após passarem pela etapa de avaliação humana, foram descartados.
   - Mesmo testando diferentes valores de hiperparâmetros, algoritmos, reduzindo a dimensionalidade dos dados, entre outros, os grupos gerados nesse primeiro experimento mostraram-se incapazes de agrupar, corretamente, os dados por categoria de *persona* e desejo. Porém, isso não significa que essa é uma tarefa impossível, já que pode existir algum conjunto específico de hiperparâmetros e/ou algoritmos que resolvam essa tarefa.
   - Para o conjunto de dados, algoritmos e hiperparâmetros utilizados, a métrica de distância euclidiana mostrou, de forma geral,

um melhor desempenho, conforme as métricas de avaliação utilizadas, do que a cosseno. Foram buscados artigos que comparacem as métricas de distância utilizadas nesse projeto, porém, apenas um foi encontrado: em [Qian et al. 2004] foi mostrado por análise teórica e resultados experimentais que, em dados com alta dimensionalidade, as distâncias cosseno e euclidiana apresentam resultados similares e, em dados normalizados e agrupados, essas duas métricas se tornam mais similares. Esse mesmo resultado foi alcançado em [Vadivel et al. 2003]. Em [Sinwar and Kaushik 2014], os resultados experimentais dos autores mostraram que a distância euclidiana obteve um melhor resultado quando comparado com a distância manhattan.

- De forma geral, foi percebido que o algoritmo *K-Means*, segundo as métricas testadas, tanto de distância quanto de avaliação, sobre o conjunto de dados utilizado, obteve resultados mais safisfatórios. Dessa forma, uma estratégia interessante é a de utilizá-lo primeiro para obter maior conhecimento sobre os dados para, só então, partir para o uso de outros algoritmos.
- Apenas remover as *stop words*, sem aplicar sobre o conjunto de dados nenhuma técnica de *lemmatização* ou *stemmização*, apresentou resultados bastante satisfatórios em alguns algoritmos, sendo algo muito apreciado, visto que remove complexidade do processo de agrupamento.
- Por fim, as métricas de avaliação, de forma geral, não são diretamente proporcionais entre si, dado que, quando uma métrica de avaliação atinge um alto valor, as demais não necessariamente apresentam o mesmo comportamento.

Algoritmo	**Métrica de Distância**	**Coeficiente de *Silhouette***	**Índice de Calinski-Harabasz**	**Índice de Davies-Bouldin**	**Tempo de Execução**
DBSCAN	Euclidiana	0.760848	36.264652	1.005990	0.025091 seg
GMM	Mahalanobis	0.073631	2.036996	1.695388	0.422684 seg
Hierárquico Aglomerativo	Euclidiana	0.564175	97.825598	0.836707	0.020472 seg
K-Means	Cosseno	0.863832	824.547571	0.215420	13.783269 seg

Tabela 1. Resultados do Primeiro Experimento

(b) Verificar agrupamento a partir das *personas*:
A segunda fase derivou dos resultados da primeira, buscando, primeiramente, criar, através do uso de *regex*, grupos de *personas* e, a partir desses grupos, aplicar novamente os algoritmos de agrupamento, visando, agora, agrupar os dados apenas pelos desejos expressos nas histórias de cada grupo de *persona*.
Como nessa segunda etapa de testes os dados a serem agrupados possuiam tamanhos diferentes, muitos sendo bastante pequenos[1], foi possível realizar a categorização manual de alguns grupos, permitindo uma verificação mais precisa de alguns agrupamentos. Há de ser mencionado, entretanto, que essa categorização foi realizado pelo próprio autor e, como a mesma não foi revisada por outros, pode existir um viés nos dados.
A Tabela 2 apresenta os melhores resultados obtidos por algoritmo. Nessa etapa de testes, as conclusões gerais foram:

[1]Menos de 40 histórias.

- As métricas de avaliação internas nos grupos de *personas* manualmente rotulados, se mostraram insuficientes para a correta avaliação dos grupos gerados, visto que, mesmo quando o modelo conseguia agrupar com elevada precisão os dados, essas métricas apresentavam valores insatisfatórios. Porém, é importante reforçar novamente que, como os dados foram categorizados pelo autor, e não foram revisados por outros, esse pode ser um erro de metodologia e não um erro no uso das métricas.
- Ao contrário dos resultados do primeiro experimento, neste a métrica cosseno apresentou resultados mais próximos aos da euclidiana e, em alguns casos, melhores. Isso provavelmente ocorreu devido ao número reduzido de histórias de usuários processadas, consequentemente reduzindo a dimensionalidade dos dados.
- Os algoritmos Hierárquico Aglomerativo e GMM apresentaram os melhores resultados nas métricas de validação externas, seguidos pelo algoritmo DBSCAN e, por último, o *K-Means*, que obteve os piores resultados.
- Apesar de o DBSCAN não ter apresentado os melhores resultados nas métricas de avaliação externas, o mesmo não é verdade no que tange as métricas de avaliação internas, o que leva o autor a crer que, de modo geral, esse algoritmo se adequa melhor ao conjunto de dados testado.
- De forma geral, apesar de terem sido testadas diferentes formas de pré-processar os textos, as que dominaram essa bateria de testes foram:
  - apenas remover *stop words*;
  - remover *stop words* e aplicar *stemmização* e
  - remover *stop words*, aplicar *lemmatização* e *stemmização*.

Algoritmo	**Métrica de Distância**	**Coeficiente de *Silhouette***	**Índice de Calinski-Harabasz**	**Índice de Davies-Bouldin**	**Homogeneidade**	**Completude**	***V-Measure***	**Índice de Rand Ajustado**	**Informação Mútua Ajustada**	**Tempo de Execução**
DBSCAN	Cosseno	0.43873	4.856284	1.56226	0.752982	0.760254	0.756601	0.557313	0.578222	0.006761 seg
GMM	Mahalanobis	0.339904	4.894808	1.624545	0.817083	0.832559	0.824748	0.644831	0.702807	0.003535 seg
Hierárquico Aglomerativo	Manhattan	0.407233	6.062793	0.738213	0.949886	0.759364	0.844007	0.646569	0.656034	0.000322 seg
K-Means	Cosseno	0.094928	3.749833	2.840545	0.299127	1.0	0.460504	0.200745	0.395783	0.121665 seg

Tabela 2. Resultados do Segundo Experimento

5. **Desenvolvimento da Interface *web*:**
   Uma vez que os experimentos foram concluídos, iniciou-se o desenvolvimento da interface *web*. Para tal, foi utilizado o *framework Dash*. Buscando criar uma ferramenta que possa ser utilizada de forma genérica, e tendo em mente os resultados obtidos nas fases de análise exploratória e experimentos, foram adicionados três principais módulos na ferramenta:
   (a) Um breve módulo para realizar uma análise exploratória dos dados;
   (b) Um módulo para agrupar todos os dados, sem separá-los por *personas*;
   (c) Um módulo para exibir os dados agrupados por categoria de *persona*.

   A Figura 2 exibe o protótipo inicial da interface, que, apesar de simplista, se mostrou útil para visualizar os agrupamentos e obter *insights* dos dados. Ainda sobre essa interface, a Figura 3 exibe a análise exploratória dos dados, a Figura 4 exibe

as histórias em forma de gráfico, porém sem aplicar nenhum agrupamento. A Figura 5 exibe o resultado do agrupamento dos dados e, por fim, a Figura 6 exibe o agrupamento por *personas*.

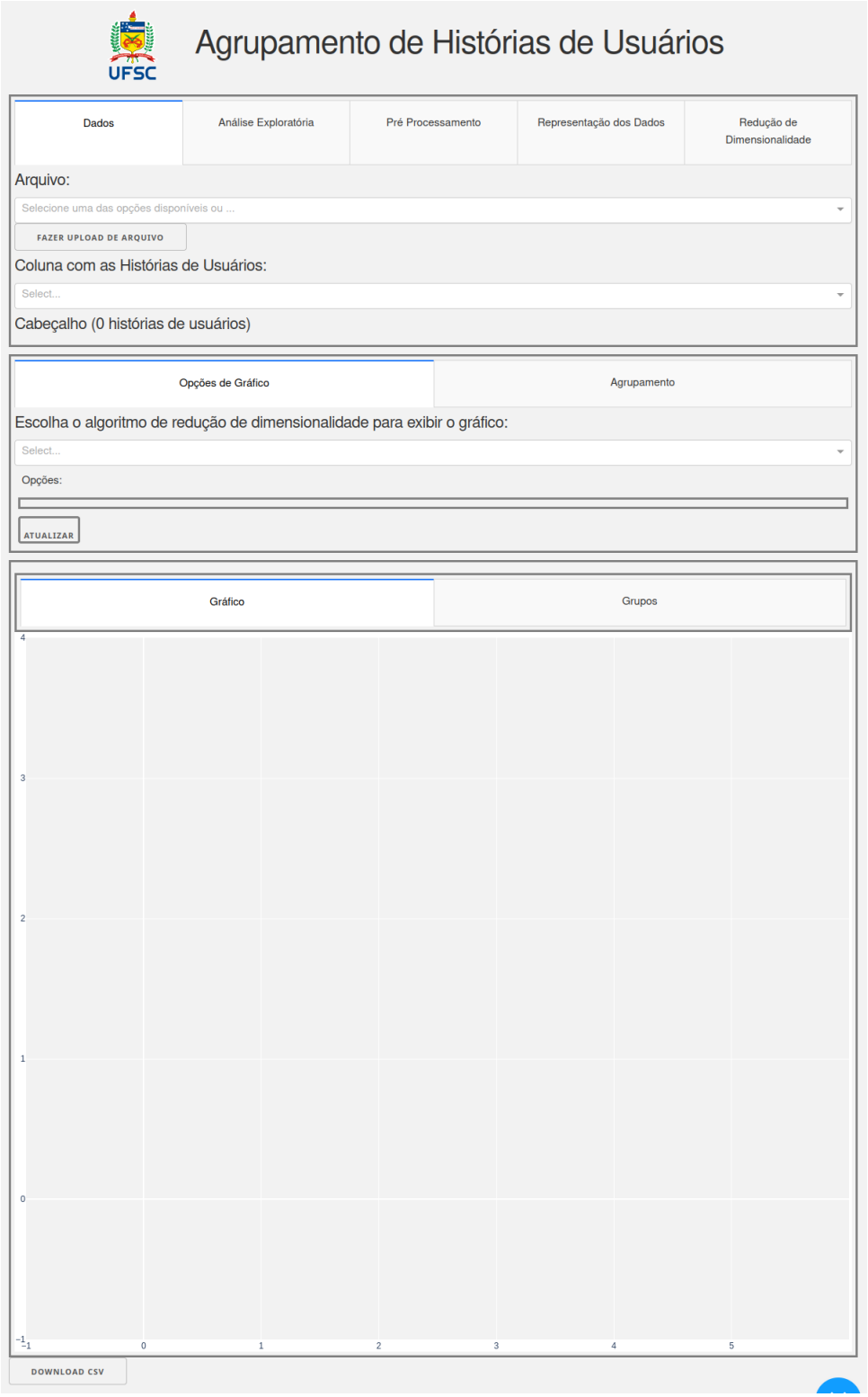

Figura 2. Visão geral da interface

Figura 3. Análise exploratória

Figura 4. Apenas visualizar graficamente os dados

Figura 5. Executar algoritmo de agrupamento

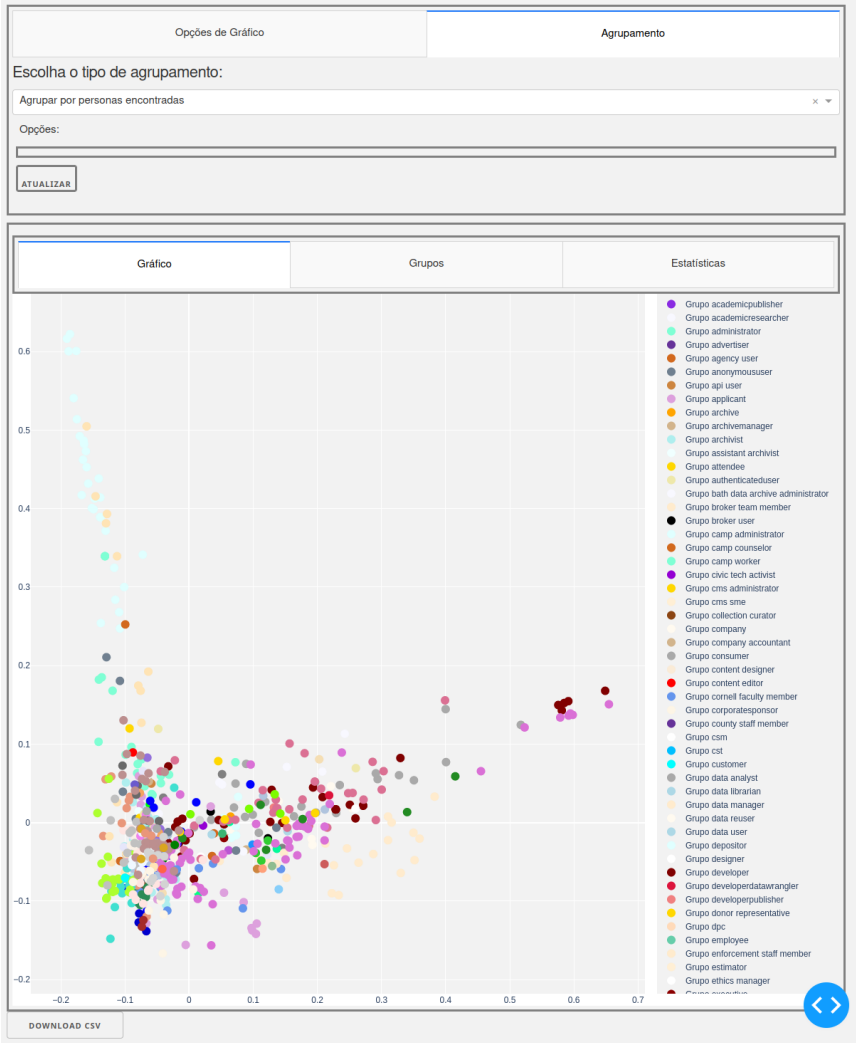

Figura 6. Agrupar por personas

O código desta interface se encontra disponível no *link* https://github.com/SadiJr/natasha.

3.1. Avaliação de Usabilidade

Para [Nielsen et al. 2012] usabilidade é definida como um atributo de qualidade, responsável por determinar a facilidade de uso de determinada interface de usuário. Segundo [Rubin and Chisnell 2008] teste de usabilidade é o processo no qual participantes representativos de determinado grupo alvo avaliam o grau de acerto que determinado produto possuí em relação à critérios específicos de usabilidade, sendo que tal avaliação pode utilizar parâmetros como a eficácia, eficiência e satisfação. De acordo com [Nielsen 2012], se o objetivo do teste é o de entender a utilização de uma interface, visando melhorar seu *design*, 2 pessoas é o número ideal para projetos de baixo custo.

Dessa forma, visando avaliar o protótipo desenvolvido neste projeto, foi realizado um teste de usabilidade com pessoas inseridas no contexto de desenvolvimento de *software*. Para tal teste, foi utilizada a escala *System Usability Scale*

(SUS) (*System Usability Scale*) [Brooke et al. 1996], visando avaliar a efetividade[2], eficiência[3] e satisfação[4] dos mesmos no uso do protótipo. O SUS é aplicado através de um questionário com 10 perguntas, cujas respostas podem ser de 1 (discordo totalmente) à 5 (concordo totalmente). Além disso, foi disponibilizada a oportunidade dos participantes de realizarem comentários sobre o protótipo.
Foram criadas 8 atividades para serem avaliadas no teste, sendo que, para cada uma delas, foi definida a porcentagem e tempo esperados de conclusão da mesma. A Tabela 3 detalha essas atividades, assim como a porcentagem e tempo mínimo esperados de conclusão das mesmas. Aqui é importante ressaltar que o tempo mínimo esperado desconsidera o tempo necessário para realizar algum processamento computacional como, por exemplo, reduzir a dimensionalidade dos dados.

Número	Atividade	Eficácia	Eficiência
1	Selecionar um conjunto de histórias de usuários	2/2	10 segundos
2	Selecionar a coluna que contém as histórias de usuários	2/2	5 segundos
3	Selecionar uma ou mais formas de pré-processar os dados e analisar o resultado	2/2	20 segundos
4	Selecionar uma forma de representação vetorial dos textos e analisar o vetor gerado	2/2	10 segundos
5	Selecionar uma forma de representação gráfica dos dados e inserir os hiperparâmetros necessários	2/2	20 segundos
6	Selecionar uma forma de agrupamento e inserir os hiperparâmetros necessários	2/2	20 segundos
7	Navegar pelo gráfico gerado	2/2	10 segundos
8	Fazer o *download* do arquivo CSV com os dados agrupados	2/2	10 segundos

Tabela 3. Atividades realizadas

Para a aplicação do teste, dois participantes foram selecionados, sendo que ambos são graduados do curso de Sistemas de Informação e atuam como desenvolvedores de *software fullstack*, do gênero masculino, e possuem 23 e 24 anos respectivamente. Para a realização do teste, foi realizado um encontro presencial na Biblioteca Universitário, onde o testador descreveu para ambos o contexto e uso do protótipo.
Antes de iniciar os testes, foi recomendado que os participantes se sentissem livres para realizar demais atividades, caso quisessem, assim como, a qualquer momento, expressar críticas/sugestões sobre o protótipo. A Tabela 4 apresenta os resultados obtidos nesse teste, onde os valores em vermelho indicam caso o resultado esperado não foi alcançado, e os valores em verde o contrário. A Tabela 5 apresenta o resultado do questionário SUS e, por fim, a Tabela 6 apresenta as sugestões/críticas sintetizadas dos usuários.

[2]Sucesso no uso do protótipo.
[3]Esforço necessário para utilizar o protótipo.
[4]Valor subjetivo do usuário sobre o quão agradável foi a utilização do protótipo.

Atividade	Participante	Concluiu atividade	Tempo
1	1	Sim	8 segundos
	2	Sim	3 segundos
2	1	Sim	1 segundos
	2	Sim	2 segundos
3	1	Sim	9 segundos
	2	Sim	14 segundos
4	1	Sim	1 segundos
	2	Sim	3 segundos
5	1	Sim	11 segundos
	2	Sim	16 segundos
6	1	Sim	13 segundos
	2	Sim	19 segundos
7	1	Sim	2 segundos
	2	Sim	7 segundos
8	1	Sim	3 segundos
	2	Sim	4 segundos

Tabela 4. Resultados do teste

Participante	Pontuação SUS
1	90 pontos
2	70 pontos
Média	80 pontos

Tabela 5. Resultado do questionário SUS

Participante	Sugestão/crítica
1	A interface poderia ter mais cores
2	Os componentes da interface poderiam ter um botão de ajuda, para explicar o que cada um deles é e faz
	Na opção de **Representação dos Dados** só existe uma opção, então não faz sentido ter um *Select box* lá
	Seria interessante ter um *link* no cabeçalho da página para redirecionar para o projeto no *GitHub*
	Alguns botões ficariam melhores se alinhados no centro da página ao invés de no canto esquerdo

Tabela 6. Sugestões dos participantes

4. Conclusão

Neste trabalho, foi investigado a efetividade de agrupar, de forma não supervisionada, histórias de usuários geradas como requisitos de *software*, utilizando diferentes abordagens. A partir dessa investigação, uma ferramenta *web* foi proposta, visando automatizar o processo de agrupamento de histórias de usuários. Durante o desenvolvimento da mesma, foi perceptível que a etapa de pré-processamento exigiu um esforço considerável para garantir a qualidade dos dados.

Os resultados preliminares mostraram que, independente da abordagem utilizada no conjunto de dados de teste, não foi possível agrupar, de forma satisfatória, as histórias de usuários por categoria de *persona* e desejo. Com relação à *interface web* desenvolvida, foi percebida uma grande utilidade da mesma na etapa de análise exploratória dos dados, especialmente quando o usuário não possuí conhecimento dos grupos existentes nas histórias de usuários, assim como para visualizar, de forma interativa, os possíveis grupos existentes nos dados, sobre a ótica de diferentes algoritmos de agrupamento.

Referências

[Ambreen et al. 2018] Ambreen, T., Ikram, N., Usman, M., and Niazi, M. (2018). Empirical research in requirements engineering: trends and opportunities. *Requirements Engineering*, 23(1):63–95.

[Baeza-Yates et al. 1999] Baeza-Yates, R., Ribeiro-Neto, B., et al. (1999). *Modern information retrieval*, volume 463. ACM press New York.

[Baker and McCallum 1998] Baker, L. D. and McCallum, A. K. (1998). Distributional clustering of words for text classification. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 96–103.

[Bekkerman et al. 2001] Bekkerman, R., El-Yaniv, R., Tishby, N., and Winter, Y. (2001). On feature distributional clustering for text categorization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 146–153.

[Bourque et al. 1999] Bourque, P., Dupuis, R., Abran, A., Moore, J. W., and Tripp, L. (1999). The guide to the software engineering body of knowledge. *IEEE software*, 16(6):35–44.

[Brooke et al. 1996] Brooke, J. et al. (1996). Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7.

[Caliński and Harabasz 1974] Caliński, T. and Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, 3(1):1–27.

[Cutting et al. 2017] Cutting, D. R., Karger, D. R., Pedersen, J. O., and Tukey, J. W. (2017). Scatter/gather: A cluster-based approach to browsing large document collections. In *ACM SIGIR Forum*, volume 51, pages 148–159. ACM New York, NY, USA.

[Dalpiaz et al. 2019] Dalpiaz, F., Van Der Schalk, I., Brinkkemper, S., Aydemir, F. B., and Lucassen, G. (2019). Detecting terminological ambiguity in user stories: Tool and experimentation. *Information and Software Technology*, 110:3–16.

[Davies and Bouldin 1979] Davies, D. L. and Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, (2):224–227.

[Feldman and Dagan 1995] Feldman, R. and Dagan, I. (1995). Knowledge discovery in textual databases (kdt). In *KDD*, volume 95, pages 112–117.

[Fukunaga 2013] Fukunaga, K. (2013). *Introduction to statistical pattern recognition*. Elsevier.

[Halkidi et al. 2001] Halkidi, M., Batistakis, Y., and Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of intelligent information systems*, 17(2):107–145.

[Jain and Dubes 1988] Jain, A. K. and Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice-Hall, Inc.

[Jain et al. 1999] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323.

[Kassab 2015] Kassab, M. (2015). The changing landscape of requirements engineering practices over the past decade.

[King 2015] King, R. S. (2015). *Cluster analysis and data mining: An introduction*. Mercury Learning and Information.

[Konrad and Gall 2008] Konrad, S. and Gall, M. (2008). Requirements engineering in the development of large-scale systems. In *2008 16th IEEE International Requirements Engineering Conference*, pages 217–222. IEEE.

[Lindvall et al. 2002] Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., Tesoriero, R., Williams, L., and Zelkowitz, M. (2002). Empirical findings in agile methods. In *Conference on extreme programming and agile methods*, pages 197–207. Springer.

[Lindvall 2002] Lindvall, T. (2002). *Lectures on the coupling method*. Courier Corporation.

[Lucassen et al. 2016] Lucassen, G., Dalpiaz, F., van der Werf, J. M. E., and Brinkkemper, S. (2016). The use and effectiveness of user stories in practice. In *International working conference on requirements engineering: Foundation for software quality*, pages 205–222. Springer.

[MEDRI 2011] MEDRI, W. (2011). Análise exploratória de dados. *Londrina: Universidade Estadual de Londrina*.

[Nielsen 2012] Nielsen, J. (2012). How many test users in a usability study. *Nielsen Norman Group*, 4(06).

[Nielsen et al. 2012] Nielsen, J. et al. (2012). Usability 101: Introduction to usability.

[Noel et al. 2018] Noel, R., Riquelme, F., Mac Lean, R., Merino, E., Cechinel, C., Barcelos, T. S., Villarroel, R., and Munoz, R. (2018). Exploring collaborative writing of user stories with multimodal learning analytics: A case study on a software engineering course. *IEEE Access*, 6:67783–67798.

[Pearson 1901] Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

[Qian et al. 2004] Qian, G., Sural, S., Gu, Y., and Pramanik, S. (2004). Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *Proceedings of the 2004 ACM symposium on Applied computing*, pages 1232–1237.

[Rousseeuw 1987] Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

[Rubin and Chisnell 2008] Rubin, J. and Chisnell, D. (2008). *Handbook of usability testing: how to plan, design and conduct effective tests*. John Wiley & Sons.

[Schön et al. 2017] Schön, E.-M., Thomaschewski, J., and Escalona, M. J. (2017). Agile requirements engineering: A systematic literature review. *Computer standards & interfaces*, 49:79–91.

[Schwaber 1997] Schwaber, K. (1997). Scrum development process. In *Business object design and implementation*, pages 117–134. Springer.

[Shirkhorshidi et al. 2015] Shirkhorshidi, A. S., Aghabozorgi, S., and Wah, T. Y. (2015). A comparison study on similarity and dissimilarity measures in clustering continuous data. *PloS one*, 10(12):e0144059.

[Sinwar and Kaushik 2014] Sinwar, D. and Kaushik, R. (2014). Study of euclidean and manhattan distance metrics using simple k-means clustering. *Int. J. Res. Appl. Sci. Eng. Technol*, 2(5):270–274.

[Vadivel et al. 2003] Vadivel, A., Majumdar, A., and Sural, S. (2003). Performance comparison of distance metrics in content-based image retrieval applications. In *International Conference on Information Technology (CIT), Bhubaneswar, India*, pages 159–164.

[Van der Maaten and Hinton 2008] Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

[Wautelet et al. 2014] Wautelet, Y., Heng, S., Kolp, M., and Mirbel, I. (2014). Unifying and extending user story models. In Jarke, M., Mylopoulos, J., Quix, C., Rolland, C., Manolopoulos, Y., Mouratidis, H., and Horkoff, J., editors, *Advanced Information Systems Engineering*, pages 211–225, Cham. Springer International Publishing.

APÊNDICE C – HIPERPARÂMETROS TESTADOS

Esse apêndice foca em detalhar os hiperparâmetros testados na Seção 4.5.

- *K-Means*:
  Foram testados os hiperparâmetros *n_clusters*, *max_iter*, *n_init* e utilizado o valor 42 como *random_state*.

  Nos primeiros testes realizados, vide Seção 4.5.1, o hiperparâmetro *n_clusters* foi testado dentro do intervalo 135 até 400, sendo incrementado em 1. Já nos testes realizados na Seção 4.5.2, esse hiperparâmetro foi testado dentro do intervalo de 2 até a metado do total de dados, sendo incrementado em 1.

  Os demais hiperparâmetros foram testados nos intervalos:

  - *max_iter*: 300 até 500, sendo incrementado em 1;
  - *n_init*: 10 até 50, sendo incrementado em 1.

- *AgglomerativeClustering*:
  Foram testados os hiperparâmetros *n_clusters*, *linkage* e *affinity*.

  O parâmetro *n_clusters* seguiu a mesma lógica do algoritmo *K-Means*, tendo os demais hiperparâmetros testados nos intervalos:

  - *linkage*: *ward*, *complete*, *average* e *single*;
  - *affinity*: *cosine*, *euclidean* e *manhattan*.

- *DBSCAN*:
  Foram testados os hiperparâmetros *metric*, *eps* e *min_samples* nos intervalos:

  - *metric*: *cosine*, *euclidean* e *manhattan*.
  - *eps*: 0.1 até 2.0, sendo incrementado em 0.1;
  - *min_samples*: 2 até 10, sendo incrementado em 1.

- *GaussianMixture*:
  Foram testados os hiperparâmetros *n_components*, *covariance_type* e *tol*, sendo usado 42 como *random_state*.

  - *covariance_type*: *full*, *tied*, *diag* e *spherical*;
  - *tol*: 0.00001 até 0.002, sendo incrementado em 0.0001.

  O hiperparâmetro *n_components* seguiu a mesma lógica do *K-Means*.

- *TfidfVectorizer*:
  Foram testados os parâmetros *max_df* e *min_df*:

  - *max_df*: 1.0, e depois no intervalo de 0.95 até 0.8, sendo decrementado em 0.01.

  - *min_df*: 1, e depois no intervalo de 0.05 até 0.2, sendo incrementado em 0.01.

- PCA e t-SNE:
  Foi utilizado o valor 42 como *random_state* em ambos. Com relação ao PCA, foi utilizado o valor 0.95 como *n_components*.

APÊNDICE D – PRÉ-PROCESSAMENTOS TESTADOS

Durante os testes na seção 4.5, foi testado pré-processar os textos de cinco formas diferentes, para verificar o impacto no pré-processamento nos grupos gerados. Abaixo se encontra um exemplo de frase existente no conjunto de dados, e os diferentes tipos e resultados das formas de pré-processamento:

Frase Original: “As a UI designer, I want to report to the Agencies about user testing, so that they are aware of their contributions to making Broker a better UX.”.

As formas de pré-processamento utilizadas foram:

- Apenas remover pontuações, espaços em branco, números e caractéres especiais.
  Resultado: “as a ui designer i want to report to the agencies about user testing so that they are aware of their contributions to making broker a better ux”
- Remover pontuações, espaços em branco, números, caractéres especiais e *stop words*.
  Resultado: “ui designer want report agencies user testing aware contributions making broker better ux”
- Remover pontuações, espaços em branco, números, caractéres especiais; *stop words* e aplicar função de lematização.
  Resultado: “ui designer want report agency user testing aware contribution making broker better ux”
- Remover pontuações, espaços em branco, números, caractéres especiais; *stop words* e aplicar função de stemmização.
  Resultado: “ui design want report agenc user test awar contribut make broker better ux”
- Remover pontuações, espaços em branco, números, caractéres especiais; *stop words* e aplicar função de stemmização e lemmatização.
  Resultado: “ui design want report agenc user test awar contribut make broker better ux”

APÊNDICE E – CONFIGURAÇÕES DE AMBIENTE

As seguintes configurações de *hardware* e de *software* foram utilizadas neste projeto.

- *Hardware*:

Computador	Lenovo ideapad 320-15IKB
Processador	Intel i5-8250U (8) @ 3.400GHz
Memória Primária	19790MiB DDR4 2133 MT/s
Placa de vídeo	NVIDIA GeForce MX150 - 2048 MB
Memória Secundária	SSD 1TB WDC WDS100T2B0A

Tabela 10 – *Hardware* utilizado

- *Software*:

Sistema Operacional	Manjaro Linux
Kernel	5.17.15-1
Linguagem	Python 3.10.5
cufflinks	0.17.3
dash	2.7.0
flask_caching	2.0.1
matplotlib	3.6.2
nltk	3.7
numpy	1.23.5
pandas	1.5.1
plotly	5.11.0
scikit_learn	1.1.3

Tabela 11 – *Softwares* utilizados

APÊNDICE F – *PERSONAS* ENCONTRADAS NOS DADOS

Esse apêndice fornece quais foram as *personas* que foram encontradas no conjunto de dados utilizado nos experimentos.

- academicpublisher;
- academicresearcher;
- administrator;
- advertiser;
- agency user;
- anonymoususer;
- api user;
- applicant;
- archive;
- archivemanager;
- archivist;
- assistant archivist;
- attendee;
- authenticateduser;
- bath data archive administrator;
- broker team member;
- broker user;
- camp administrator;
- camp counselor;
- camp worker;
- civic tech activist;
- cms administrator;
- cms sme;
- collection curator;
- company;
- company accountant;
- consumer;
- content designer;

- content editor;
- cornell faculty member;
- corporatesponsor;
- county staff member;
- csm;
- cst;
- customer;
- data analyst;
- data librarian;
- data manager;
- data reuser;
- data user;
- depositor;
- designer;
- developer;
- developerdatawrangler;
- developerpublisher;
- donor representative;
- dpc;
- employee;
- enforcement staff member;
- estimator;
- ethics manager;
- executive;
- extension administrator;
- externalcollaborator;
- externalcoordinator;
- fabs user;
- faculty data steward;
- faculty member;
- funder;

- fundingbody;
- government publisher;
- inspection staff member;
- inspection staff supervisor;
- inspection supervisor;
- inspector;
- institutional data manager;
- institutional data steward;
- investigator;
- it manager;
- it officer;
- it staff member;
- juniorresearcher;
- legalofficer;
- library staff member;
- machine learning expert;
- manager;
- member;
- metadata manager;
- moderator;
- nsf administrator;
- nsf employee;
- nsf member;
- olderperson;
- owner;
- parent;
- participant;
- patron;
- pi;
- plan review staff member;
- planning staff member;

- postgraduate convenor;
- practitioner;
- prospectiveapplicant;
- publisher;
- recruiter;
- rector;
- recyclingfacility;
- recyclingfacility representative;
- repository manager;
- repository operator;
- repository support team member;
- repositorymanagerresearcher;
- research centre director;
- research evaluation manager;
- research head;
- research information manager;
- research participant;
- researcher;
- researchergovernment publisher;
- researcherpublisher;
- site administrator;
- site editor;
- site member;
- site visitor;
- sponsor;
- staff member;
- stakeholder;
- student;
- summit coordinator;
- superuser;
- team member;

- tester;
- trainee;
- trainer;
- trainingcoordinator;
- ui designer;
- univitservice;
- user;
- user researcher;
- visitor;
- visualdesigner;
- web developer;
- web recruiter manager;
- website user e
- workshop attendee.

APÊNDICE G – RESULTADOS DO PRIMEIRO EXPERIMENTO POR ALGORITMO

A Tabela 12 apresenta os resultados obtidos com o algoritmo DBSCAN. A Tabela 13 apresenta os resultados obtidos com o algoritmo *Agglomerative Clustering*. Já a Tabela 14 apresenta os resultados do algoritmo GMM. Por fim, a Tabela 15 apresenta os resultados obtidos com o algoritmo *KMeans*. Nessas tabelas, existem as seguintes colunas:

- **MD*:** Indica a métrica de distância usada no experimento, podendo ter os valores:
  - **E*:** Distância euclidiana;
  - **M*:** Distância manhattan;
  - **C*:** Distância coseno e
  - **ML*:** Distância mahalanobis.

  Todas essas distâncias foram detalhadas na Seção 2.1.1.
- **S*:** Indica a métrica de avaliação Coeficiente de *Silhouette*, um valor mais próximo de 1 indica melhor agrupamento;
- **ICH*:** Indica a métrica de avaliação do Índice de Calinski-Harabasz, quanto maior o valor nessa métrica, mais densos e bem separados são os grupos;
- **IDB*:** Indica a métrica de avaliação do Índice de Davies-Bouldin, valores mais próximos de zero indicam grupos mais bem divididos.

  Consulte a Seção 2.1.2 para maiores informações sobre cada uma.
- **TE*:** incida o tempo de execução do algoritmo;
- **min_df:** indica o valor mínimo utilizado pelo algoritmo TF-IDF para realizar a remoção de termos pouco frequentes no texto;
- **max_df:** indica o valor máximo utilizado pelo algoritmo TF-IDF para realizar a remoção de termos muito frequentes no texto;
- **PP*:** Indica as formas de pré-processar os textos utilizadas, podendo ser:
  - *X1*: não remove *stop words*, não aplica *lemmatização* ou *stemmização*;
  - *X2*: apenas remove *stop words*;
  - *X3*: apenas aplica *lemmatização*;
  - *X4*: apenas aplica *stemmização*;
  - *X5*: remove *stop words* e aplicação *lemmatização*;
  - *X6*: remove *stop words* e aplica *stemmização*;
  - *X7*: remove *stop words*, aplica *lemmatização* e *stemmização* e
  - *X8*: apenas aplica *lemmatização* e *stemmização*.
- **HP*:** indica os hiperparâmetros usados pelo algoritmo.
- **Negrito**: valores em negrito indicam os melhores resultados alcançados pela métrica onde o mesmo está inserido.

MD*	S*	ICH*	IDB*	TE*	min_df	max_df	PP*	HP*
E*	**0.760848**	**36.264652**	1.005990	0.025091 seg	0.05	0.95	X2	EPS: 0.3, min_samples: 2
E*	0.445736	13.277871	**0.932255**	0.014922 seg	0.2	0.8	X3	EPS: 0.1, min_samples: 2
M*	**0.735215**	**35.361896**	1.001319	0.032944 seg	0.05	0.95	X2	EPS: 0.4, min_samples: 2
M*	0.410036	13.063780	**0.935525**	0.016622 seg	0.2	0.8	X3	EPS: 0.1, min_samples: 2
C*	**0.046983**	2.401551	1.964561	0.119399 seg	1	1.0	X8	EPS: 0.6, min_samples: 2
C*	-0.085820	**6.176275**	1.253387	0.034209 seg	0.12	0.88	X4	EPS: 0.1, min_samples: 2
C*	-0.153545	4.523059	**1.249243**	0.035990 seg	0.1	0.9	X1	EPS: 0.1, min_samples: 2

Tabela 12 – Resultados do Experimento - DBSCAN

Com relação as conclusões sobre esse algoritmo:

- A métrica de distância euclidiana apresentou melhores resultados em comparação com as demais, embora isso já fosse experado, vide (QIAN *et al.*, 2004; VADIVEL; MAJUMDAR; SURAL, 2003; SINWAR; KAUSHIK, 2014; AMINE; ELBERRICHI; SIMONET, 2010)
- Em ambos os casos, independente da métrica utilizada, o hiperparâmetro *min_samples* se manteve constante;
- A execução desse algoritmo se mostrou extremamente rápida, sequer chegando a ultrapassar 1 minuto.
- De todos os algoritmos testados, foi o que obteve o pior resultado usando a distância Coseno.

MD*	**S***	**ICH***	**IDB***	**TE***	**min_df**	**max_df**	**PP***	**HP***
E*	**0.564175**	**97.825598**	**0.836707**	0.020472 seg	0.2	0.8	X1	K: 135, *linkage*: *ward*
M*	**0.138857**	2.247299	1.160961	0.367498 seg	1	1.0	X7	K: 397, *linkage*: *complete*
M*	0.080066	**2.426792**	2.320409	0.365995 seg	1	1.0	X6	K: 135, *linkage*: *complete*
M*	-0.06879	1.053005	**1.028609**	0.636414 seg	1	1.0	X4	K: 139, *linkage*: *single*
C*	**0.166596**	2.436421	1.109527	0.601723 seg	1	1.0	X6	K: 388, *linkage*: *complete*
C*	0.109507	**2.778201**	2.287262	0.648578 seg	1	1.0	X6	K: 135, *linkage*: *complete*
C*	0.149464	2.351044	**1.02704**	0.647533 seg	1	1.0	X7	K: 399, *linkage*: *average*

Tabela 13 – Resultados do Experimento - *Agglomerative*

Com relação as conclusões sobre esse algoritmo:

- Assim como o DBSCAN, apresentou melhores resultados com a métrica de distância euclidiana, além de também apresentar resultados mais satisfatórios na métrica Coseno quando comparado ao DBSCAN, porém obteve resultados mais insatisfatórios na métrica manhattan.
- Fora com o uso da métrica euclidiana, esse algoritmo apresentou um pior tempo de execução quando comparado ao DBSCAN, embora também não tenha ultrapassado um minuto. Além disso, fora os testes com a métrica euclidiana, os demais testes apresentaram estabilidade no uso do *min_df* e *max_df*.

MD*	**S***	**ICH***	**IDB***	**TE***	**min_df**	**max_df**	**PP***	**HP***
ML*	**0.073631**	2.036996	1.695388	0.422684 seg	1	1.0	X1	K: 285, *covariance*: *spherical*, *tolerance*: 0.00191
ML*	0.05396	**2.392152**	2.562143	0.282018 seg	1	1.0	X2	K: 135, *covariance*: *full*, *tolerance*: 0.00191
ML*	0.07315	2.031404	**1.685178**	07:48.311415 min	1	1.0	X1	K:288, *covariance*: *full*, *tolerance*: 0.00061

Tabela 14 – Resultados do Experimento - GMM

Com relação as conclusões sobre esse algoritmo:

- Foi, de todos os algoritmos, o que obteve os piores resultados, assim como o que mais demorou em uma de suas execuções.
- Além disso, foi o algoritmo que apresentou o maior consumo de RAM, sendo, em muitos testes, interrompido pela ausência de recursos computacionais suficientes.
- Por fim, esse algoritmo não apresentou nenhuma variação nos valores de *min_df* e *max_df* em seus melhores resultados, e mesmo assim tais resultados foram ruins, o que leva o autor a acreditar que esse tipo de algoritmo de agrupamento ou não é adequado ao conjunto de dados utilizado ou não é adequado à tarefa de agrupamento de textos.

MD*	S*	ICH*	IDB*	TE*	min_df	max_df	PP*	HP*
E*	**0.781733**	**195.441215**	**0.514631**	06.398468 seg	0.05	0.95	X2	K: 135, max_iter: 500, n_init: 50
M*	**0.852465**	**2887.866215**	**0.178453**	16.214982 seg	0.05	0.95	X2	K: 207, max_iter: 500, n_init: 50
C*	**0.863832**	**824.547571**	**0.215420**	13.783269 seg	0.06	0.94	X2	K: 136, max_iter: 500, n_init: 50

Tabela 15 – Resultados do Experimento - *KMeans*

Com relação as conclusões sobre esse algoritmo:

- Foi o que obteve os melhores resultados em todas as métricas, embora tenha sido, de modo geral, o mais demorado[1].
- Além disso, alguns hiperparâmetros desse algoritmo se manteram constantes, como o *max_iter* em 500 e o *n_init* em 50, o que, do ponto de vista do funcionamento desse algoritmo faz sentido, uma vez que, quanto mais iterações são realizadas, maiores as chances de os pontos serem realocados para os grupos corretos.
- Parecido com o GMM, esse algoritmo apresentou pouca variação nos valores de *min_df* e *max_df* e nenhuma variação da forma de pré-processar o texto, o que pode ser um indicativo de que, usando esse algoritmo, apenas remover as *stop words* seja suficiente para realizar um agrupamento exploratório para, só então, partir para outras formas de pré-processar os textos.
- Ao contrário dos demais algoritmos, aqui o uso da métrica de distância Coseno apresentou melhores resultados no que tange à métrica de Silhouette, enquanto que a distância manhattan obteve melhores resultados nas métricas de Índice de Calinski-Harabasz e Índice de Davies-Bouldin, sendo a distância euclidiana a que apresentou piores resultados em todas essas métricas de avaliação, embora seu tempo de execução tenha sido o menor entre as três.

[1] Perdendo apenas para uma das execuções do GMM que durou 07:48.311415 min.

APÊNDICE H – RESULTADOS DO SEGUNDO EXPERIMENTO POR ALGORITMO

A Tabela 16 apresenta os resultados obtidos com o algoritmo DBSCAN. A Tabela 17 apresenta os resultados obtidos com o algoritmo *Agglomerative Clustering*. Já a Tabela 18 apresenta os resultados do algoritmo GMM. Por fim, a Tabela 19 apresenta os resultados obtidos com o algoritmo *KMeans*. Nessas tabelas, afora as colunas já anteriormente explicadas na Lista 1, existem as seguintes colunas:

- **P*:** Indica a categoria de *Persona* testada e a quantidade de histórias de usuários que tal *persona* possui;
- **H*:** Indica a métrica de avaliação de homogeneidade, um valor mais próximo de 1 indica um agrupamento mais homogêneo[1];
- **C*:** Indica a métrica de avaliação de completude, um valor mais próximo de 1 indica um agrupamento mais completo[2];
- **VM*:** Indica a métrica de avaliação *V-Measure*, é a média harmônica entre homogeneidade e completudo, sendo um valor igual a 1 uma indicação de um agrupamento perfeito;
- **IRA*:** Indica a métrica de avaliação índice de Rand Ajustado, um valor próximo de 1 indica um agrupamento mais próximo ao esperado e
- **IMA*:** Indica a métrica de avaliação Informação Mútua Ajustada, cujo valor, quanto mais próximo de 1, indica um agrupamento mais próximo ao esperado.

  Maiores detalhes sobre as novas métricas de avaliação aqui apresentadas podem ser encontrados na Seção 2.1.2.

P*	MD*	S*	ICH*	IDB*	H*	C*	VM*	IRA*	IMA*	TE*	min_df	max_df	PP*	HP*
Developer, 39 histórias de usuários	E*	0.364474	4.856284	1.56226	**0.752982**	**0.760254**	**0.756601**	**0.557313**	**0.578222**	0.005084 seg	0.05	0.95	X2	EPS: 1.0, min_samples: 2
Developer, 39 histórias de usuários	M*	0.366714	4.582719	1.553029	**0.688034**	0.765149	0.724545	0.477685	0.546797	0.005128 seg	0.05	0.95	X2	EPS: 1.9, min_samples: 2
Developer, 39 histórias de usuários	M*	0.332379	4.300579	1.488669	0.684495	**0.937004**	**0.791089**	**0.508913**	**0.688623**	0.005437 seg	0.05	0.95	X6	EPS: 2.0, min_samples: 2
Developer, 39 histórias de usuários	C*	0.43873	4.856284	1.56226	**0.752982**	**0.760254**	**0.756601**	**0.557313**	**0.578222**	0.006761 seg	0.05	0.95	X2	EPS: 0.5, min_samples: 2

Tabela 16 – Resultados do Experimento - DBSCAN

Com relação às conclusões sobre esse algoritmo:

- As métricas de distância coseno e euclidiana, baseando-se nas métricas de avaliação, apresentaram os mesmos resultados, com exceção da métrica Silhouette, e no tempo de processamento.

[1] Grupos possuem dados de apenas um tipo.
[2] Os grupos possuem apenas dados da mesma classe.

- Os valores de *min_df* e *max_df* se manteram constantes, assim como a forma de pré-processar os textos, indicando que esses hiperparâmetros se adequam melhor a esse conjunto de dados, o que não significa que o mesmo é verdade para outros conjuntos de dados.
- Assim como na primeira bateria de testes, o valor de *min_samples* se manteve constante em 2;
- A métrica de distância manhattan, apesar de ter apresentado resultados piores nas métricas homogeneidade e índice de Rand Ajustado, apresentou melhores resultados nas métricas completude, *V-Measure* e Informação Mútua Ajustada.

P*	MD*	S*	ICH*	IDB*	H*	C*	VM*	IRA*	IMA*	TE*	min_df	max_df	PP*	HP*
Developer, 39 histórias de usuários	E*	0.113350	2.173865	1.066345	**0.933181**	**0.74601**	**0.829164**	**0.630504**	**0.623306**	0.000936 seg	1	1.0	X6	K: 18, *linkage*: *ward*
Developer, 39 histórias de usuários	M*	0.407233	6.062793	0.738213	**0.949886**	**0.759364**	**0.844007**	**0.646569**	**0.656034**	0.000322 seg	0.05	0.95	X6	K: 18, *linkage*: *complete*
Developer, 39 histórias de usuários	C*	0.459927	6.524824	0.724384	**0.916477**	**0.732656**	**0.814321**	**0.614439**	**0.590578**	0.000353 seg	0.05	0.95	X6	K: 18, *linkage*: *complete*

Tabela 17 – Resultados do Experimento - *Agglomerative*

Com relação às conclusões sobre esse algoritmo:

- De forma geral, foi o que apresentou os melhores resultados nas métricas de avaliação externa, assim como o que apresentou melhor desempenho em tempo de execução.
- Surprendentemente, nesse algoritmo e conjunto de dados utilizado, a métrica de distância manhattan apresentou melhores resultados em comparação com as demais.
- O pré-processamento, assim como os hiperparâmetros do algoritmo se mantiveram constantes, levando o autor a crer que os mesmos se ajustaram muito bem aos dados.

P*	MD*	S*	ICH*	IDB*	H*	C*	VM*	IRA*	IMA*	TE*	min_df	max_df	PP*	HP*
Developer, 39 histórias de usuários	ML*	0.339904	4.894808	1.624545	**0.817083**	**0.832559**	**0.824748**	**0.644831**	**0.702807**	0.003535 seg	0.05	0.95	X7	K: 9, *covariance*: *spherical*, *tolerance*: 0.00191

Tabela 18 – Resultados do Experimento - GMM

Com relação às conclusões sobre esse algoritmo:

- Apresentou resultados muito bons, bastante parecidos com o algoritmo *Agglomerative Clustering*, além de ter atingido o maior valor na métrica de Informação Mútua Ajustada e o segundo maior valor na métrica de Completude, perdendo apenas para o algoritmo *KMeans* nesta.

P*	MD*	S*	ICH*	IDB*	H*	C*	VM*	IRA*	IMA*	TE*	min_df	max_df	PP*	HP*
Developer, 39 histórias de usuários	E*	0.060435	3.510005	3.175572	**0.264422**	**0.821523**	**0.400073**	**0.162638**	**0.328109**	0.937619 seg	1	1.0	X7	K: 2, max_iter: 400, n_init: 40
Developer, 39 histórias de usuários	M*	0.086966	3.749833	2.840545	**0.299127**	**1.0**	**0.460504**	**0.200745**	**0.395783**	0.062516 seg	1	1.0	X7	K: 2, max_iter: 400, n_init: 40
Developer, 39 histórias de usuários	C*	0.094928	3.749833	2.840545	**0.299127**	**1.0**	**0.460504**	**0.200745**	**0.395783**	0.121665 seg	1	1.0	X7	K: 2, max_iter: 400, n_init: 40

Tabela 19 – Resultados do Experimento - *KMeans*

Com relação às conclusões sobre esse algoritmo:

- Apresentou, de forma geral, os piores resultados entre os algoritmos testados, com exceção da métrica de Completude, que atingiu o maior valor possível nas métricas de distância manhattan e coseno;
- Também foi o algoritmo que levou mais tempo para ser processado;
- Por fim, devido ao valor de *K* em todas as métricas de distância ter o valor de 2, isso leva o autor a crer que tal algoritmo apresentou problemas ao lidar com ruídos e formatos arbistrários nos dados, o que é o esperado desse algoritmo, conforme descrito na Seção 2.1.3.