UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

LUCAS AUGUSTO FERREIRA DE OLIVEIRA

ANÁLISE DE TEXTO NÃO SUPERVISIONADA. APLICAÇÕES: SETORES QUÍMICO E ELÉTRICO

BELO HORIZONTE - MG
2020

LUCAS AUGUSTO FERREIRA DE OLIVEIRA

ANÁLISE DE TEXTO NÃO SUPERVISIONADA. APLICAÇÕES: SETORES QUÍMICO E ELÉTRICO

Versão Final

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Química da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito parcial para obtenção do Grau de Mestre em Engenharia Química.
Linha de Pesquisa: Simulação e Otimização de Processos.
Orientador: Prof. Gustavo Matheus de Almeida.

BELO HORIZONTE – MG
2020

O48a

Oliveira, Lucas Augusto Ferreira de.
Análise de texto não supervisionada. Aplicações [recurso eletrônico] : setores químico e elétrico / Lucas Augusto Ferreira de Oliveira. - 2020.
1 recurso online (xi, 82 f. : il., color.) : pdf.

Orientador: Gustavo Matheus de Almeida.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Bibliografia: f. 78-82.

Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia química - Teses. 2. Aprendizado do computador - Teses. I. Almeida, Gustavo Matheus de. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 66.0(043)

Ficha catalográfica: Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

"Análise de Texto Não Supervisionada. Aplicações: Setores Químico e Elétrico"

Lucas Augusto Ferreira de Oliveira

Dissertação submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Química da Escola de Engenharia da Universidade Federal de Minas Gerais, como parte dos requisitos à obtenção do título de **MESTRE EM ENGENHARIA QUÍMICA**.

270ª DISSERTAÇÃO APROVADA EM 21 DE FEVEREIRO DE 2020 POR:

.....
Prof. Dr. Roberto da Costa Quinino
Depto. de Estatística/UFMG

.....
Prof. Dr. Edgar Campos Furtado
UFSJ

.....
Prof. Dr. Gustavo Matheus de Almeida
Orientador - DEQ/UFMG

AGRADECIMENTOS

Gostaria de agradecer a todos que participaram e participam da minha vida de alguma forma. Apesar de perceber que ainda preciso melhorar muito como ser humano, me vejo feliz em perceber quem estou me tornando. Se estou me tornando quem sou, isso se deve às pessoas com quem convivi e convivo e às vivências que tive e tenho. Me vejo nas pessoas que me auxiliaram e auxiliam em minha caminhada e por esse motivo sou eternamente grato.

Meu agradecimento à Deus e todos os guias que tive nessa jornada e me ajudaram a tomar decisões e refletir o que era bom pra mim. Agradeço à minha família por sempre estar presente e me apoiar em tudo que fiz e faço, minha mãe Teresinha, meu pai Orides e minhas irmãs Júlia, Paula e Letícia.

Agradeço a todos da Radix, que puderam me ajudar a equilibrar o fardo de levar com qualidade os projetos que fazemos e o mestrado. A todos do escritório de BH e aos meus parceiros do escritório do Rio, César Medina, Adonis Carvalho, Pedro Meneses, Lucas Valle e Hugo Rebelo, que me auxiliou tanto na parte técnica, meu muito obrigado.

Muito obrigado também aos companheiros de projeto na CEMIG que proporcionaram e proporcionam um ambiente tão fértil para meu crescimento pessoal e profissional, em especial, obrigado Willian Evans e time por me auxiliar diariamente nesta jornada.

Agradeço imensamente a todos da UFMG e do Departamento de Engenharia Química que fizeram com que meus dias se tornassem mais leves. Em especial agradeço à Ana pelas ótimas conversas e compainha e ao meu orientador Gustavo, que com toda certeza, se não fosse por seu auxílio e orientação em vários aspectos, eu não conseguiria chegar onde cheguei.

Muito obrigado aos membros da banca Roberto Quinino e Edgar Furtado que se disponibilizaram a estar presente neste momento tão importante pra mim.

Por fim, este trabalho foi conduzido como parte do projeto de P&D ANEEL D0650 “Inteligência Artificial aplicada ao relacionamento com clientes”, adotado pela CEMIG Distribuição S.A. O projeto é executado pela empresa Radix Engenharia e Software, Rio de Janeiro, Brasil.

Gostaria de dedicar esse trabalho a todos que lutam contra todo o caos do mundo, buscando serem pessoas melhores.

“Ninguém caminha sem aprender a caminhar, sem aprender a fazer o caminho caminhando, refazendo e retocando o sonho pelo qual se pôs a caminhar” (Paulo Freire, Pedagogia da Esperança, 1994)

RESUMO

A análise de texto é uma área que já existe há alguns anos; porém, avançou consideravelmente em função do desenvolvimento da capacidade de coleta e armazenamento de informações em formato texto. A análise de texto pode ser dividida em análise de banco de dados, mineração de texto, e extração de informação. Todos esses pontos são explorados neste trabalho, que propõe uma metodologia para a descoberta e a nomeação de agrupamentos (*clusters*). Essa metodologia utiliza processamento de linguagem natural (*Natural Language Processing*; NLP) através de uma abordagem de aprendizado de máquina não supervisionada. São utilizados dois estudos de caso reais. O primeiro diz respeito a CEMIG, uma das principais concessionárias do setor de energia elétrica no Brasil, com o objetivo de agrupar as mesagens de texto de seus clientes, ou, em outras palavras, de descobrir *intents* de seus usuários. O segundo refere-se a uma empresa de venda de máquinas para a construção civil, também no Brasil, com o objetivo de agrupar pareceres técnicos, emitidos em formato texto, de análises de laboratório de fluidos utilizado nas máquinas. Essas análises são escritas por diferentes analistas; por isso, a necessidade de uma padronização dessa informação. Obtiveram-se resultados satisfatórios em ambos os casos. A combinação, tendo-se PCA como método de redução de dimensionalidade e *k*-means como algoritmo de clusterização, mostrou-se, em geral, a de melhor desempenho, segundo a métrica usual de avaliação denominada coeficiente de *silhouette*, em geral superior a 0,95; também tendo como métricas o tamanho do agrupamento de dados denominado “aleatório”, que reune frases pouco expressivas, em torno de 6%; e o tempo de processamento computacional significativamente baixo. A metodologia se mostrou bastante eficiente para estes casos e pode ser empregada em outros contextos.

Palavras-chave: Análise de texto; Aprendizado não supervisionado; Padronização laboratorial; Comportamento de Clientes. Aprendizado de máquina.

ABSTRACT

Text analysis is an area that has been around for a few years; however, it has advanced considerably due to the development of the capacity to collect and store information in text format. Text analysis can be divided into database analysis, text mining, and information extraction. All these points are explored in this work. It proposes a methodology for the discovery and naming of clusters. This methodology uses natural language processing (Natural Language Processing; NLP) through an unsupervised machine learning approach. Two real case studies are used. The first concerns CEMIG, one of the main concessionaires in the electricity sector in Brazil, with the objective of grouping the text messages of its customers, or, in other words, of discovering intents of its users. The second refers to a company that sells machinery for civil construction, also in Brazil, with the objective of gathering technical opinions, issued in text format, of laboratory analysis of fluids used in the machines. These analyzes are written by different analysts; therefore, the need for a standardization of this information. Satisfactory results were obtained in both cases. The combination, using PCA as a method of dimensionality reduction and k-means as a clustering algorithm, proved to be, in general, the one with the best performance, according to the usual evaluation metric called silhouette coefficient, generally higher than 0,95; also having as metrics the size of the grouping of data called “random”, which brings together little expressive phrases, around 6%; and significantly low computational processing time. The methodology proved to be quite efficient for these cases and can be used in other contexts.

Keywords: Text analysis; Unsupervised learning; Laboratory standardization; Customer behavior; Machine learning.

Lista de Figuras

Lista de Tabelas

SUMÁRIO

1. Introdução

Análise de texto, bastante conhecida pelo seu termo em inglês, *text analytics*, pode também ser facilmente encontrada por *text data mining* (mineração de dados textuais) ou *text mining* (mineração de texto). Todos esses são nomes dados ao processo que em termos gerais envolve extração da informação, reconhecimento de padrões e descoberta de informações significativas, que inicialmente não eram percebidas ou conhecidas antes da aplicação do método, tudo isso realizado de forma automática (HOTHO *et al*., 2005).

A descoberta de informações significativas é feita através da aplicação de métodos estatísticos em conjunto com o processamento do texto, onde adiciona-se ou retira-se recursos linguísticos ou reestrutura-se o texto com base em premissas semânticas que têm como objetivo a descoberta de padrões e sentidos que inicialmente não estão associados ao texto em análise (FELDMAN, SANGER, 2007).

Esse é um campo que vem crescendo de forma bem rápida principalmente pela expansão globalizada da internet e meios de comunicação. A quantidade de dados gerados na internet em 2019 é maior do que a quantidade de dados gerados na internet de 1984 a 2013 (TIMES, 2019). O aumento massivo de conteúdo traz consigo o aumento de fontes de dados a serem analisados. Teoricamente, qualquer fonte que gere conteúdo de comunicação linguística pode ser usada como fonte, desde escrita manual, redes sociais, interação entre fornecedor e consumidor de um produto, até conversas verbais geradas em um filme, em uma chamada telefônica ou em uma entrevista na TV (considerando que existem diversas formas de converter o áudio em texto).

A grande maioria dos textos contidos nessas diferentes fontes não estão estruturados em um formato pré-definido, podendo conter ambiguidades que dificultam a compreensão do uso de programas tradicionais de processamento de dados textuais (GRIMES, 2008). Por esse motivo, além do modelo escolhido a ser aplicado, o pré-processamento (estruturação) dos dados é extremamente importante para o resultado gerado após a análise do texto. Existem diversos métodos que podem ser aplicados antes dos métodos de análise de texto, tokenização, remoção de *stop words*, stemização, são todas etapas que levam

a uma melhor estruturação do texto a ser analisado, que trazem a análise (MARSLAND, 2014).

De forma geral, a etapa de estruturação do texto em um formato válido para análise é feita a partir da aplicação de métodos de Processamento de Linguagem Natural (*NLP – Natural Language Processing*). NLP é uma área da linguística que estuda a criação e compreensão da linguagem humana. Interações entre máquinas e humanos são feitas com o intuito de realizar, por exemplo, reconhecimento de fala (CLARK *et al.*, 2013).

A partir da aplicação do NLP é possível associar métodos estatísticos e modelos de aprendizado de máquina para gerar o entendimento e a criação de estruturas linguísticas que gerem significado ao sistema de estudo. A combinação de métodos estatísticos e de aprendizado de máquina com abordagens de NPL é algo que está sendo bastante aplicado nos últimos tempos com uma crescente enorme. Aplicações em análise de sentimentos, detecção de *spam* de e-mails, monitoramento de redes sociais, segurança de dados e criações de *bots* de conversa estão sendo cada vez mais feitas por diferentes setores na indústria (ZHAO *et al*, 2019).

Isso abriu portas para a criação de serviços customizados de linguagem natural que são usados em aplicativos, dispositivos de IoT (*internet of things*) e *bots*, como o LUIS da Microsoft, a Alexa da Amazon ou o Watson da IBM. Estes serviços têm como base o uso de redes neurais que são treinadas usando metodologias de aprendizado de máquina supervisionadas. A criação de *bots* para interação com humanos é uma forma de aplicação direta desses serviços (CANONICO, DE RUSSIS, 2018). Esse tipo de aplicação vem crescendo nos últimos anos. Em 2019, o investimento econômico mundial nessa área foi de 5 bilhões de dólares, com expectativa de mais de 6 bilhões de dólares em 2023, como mostra a Figura 1 (TOADER *et al*, 2020).

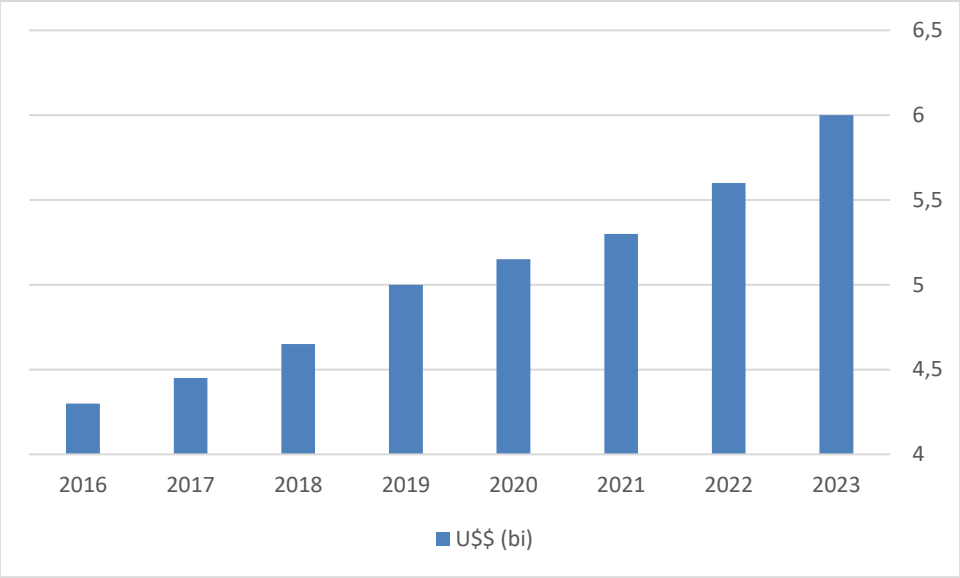


Figura 1: Gasto mundial em bilhões de dólares em desenvolvimento de *chatbots*.

2. OBJETIVOS

2.1 Objetivo geral

Identificar, de modo automático, com o uso de processamento de linguagem natural e aprendizado de máquina não supervisionado, padrões textuais, através da busca automática de intents em diferentes textos.

Verifica-se o desempenho da metodologia proposta em dois estudos de casos reais, de empresas localizadas no Brasil. O primeiro diz respeito ao setor elétrico: a CEMIG (Compainha Energética de Minas Gerais) uma concessionária de energia elétrica, com o foco no comportamento de seus clientes, através da análise textual de mensagens. O segundo refere-se ao setor químico: uma empresa de venda de maquinário de construção, com o foco em padronização laboratorial, através da análise textual das análises de laboratório de fluidos usados nas máquinas.

Em ambos os casos, busca-se uma análise de texto capaz de gerar valor de forma rápida e que seja robusta, ou seja, que consiga bons resultados independentemente da fonte dos textos.

2.2 Objetivos específicos

Os objetivos específicos do atual trabalho podem ser divididos em:

Do ponto de vista de aprendizado de máquina:

- Pré-processamento (estruturação) dos dados brutos, ou seja, dos dados não estruturados, através de processamento de linguagem natural.
- Busca de *intents* e análise de agrupamentos dos dados estruturados a partir de uma abordagem não supervisionada, que usualmente é feita por abordagens supervisionadas.
- Nomeação dos grupos (padrões, *clusters*) encontrados, ou seja, verificação de desempenho da metodologia proposta.

Do ponto de vista das aplicações:

- Encontrar padrões nos pareceres técnicos de análises físico-químicas de um óleo lubrificante, escritos por diferentes analistas, na empresa de venda de máquinas.
- Criar uma base com os grupos gerados na análise textual, com o objetivo futuro de padronizar textos diferentes; porém, com o mesmo significado (esse processo de padronização não é atividade do presente trabalho).
- Encontrar intenções e padrões nas mensagens de clientes enviadas à concessionária de energia elétrica, que não eram percebidas no atendimento humano tradicional.
- Criar uma base com os perfis das diferentes intenções nas mensagens escritas pelos clientes, com o objetivo futuro de utilizar essa informação como ponto de partida para a criação de um *chatbot* (a criação desse robô não é atividade do presente trabalho).

3. REVISÃO BIBLIOGRÁFICA

3.1 Análise de texto

Análise de texto é um termo da língua portuguesa que referencia um campo bastante amplo que vai além do analisar um conjunto de palavras. Também conhecida por *text data mining* (mineração de dados textuais) ou *text mining* (mineração de texto), pode ser tecnicamente definida, segundo Nord, como:

> "A Análise de texto é o processo de examinar grandes coleções de recursos escritos para gerar novas informações e transformar o texto não estruturado em dados estruturados, para uso em análises adicionais. É o processo de obter informações de alta qualidade do texto. O objetivo principal é, essencialmente, transformar texto em dados para análise, através da aplicação do processamento de linguagem natural (PNL) e métodos analíticos (NORD, 2009).

Existem diversas maneiras de aplicabilidade, desde análises mais simples até aquelas mais complexas. De forma genérica, a análise de texto pode ser resumida como mostra a Figura 2 (MULLER *et al.*, 2016).

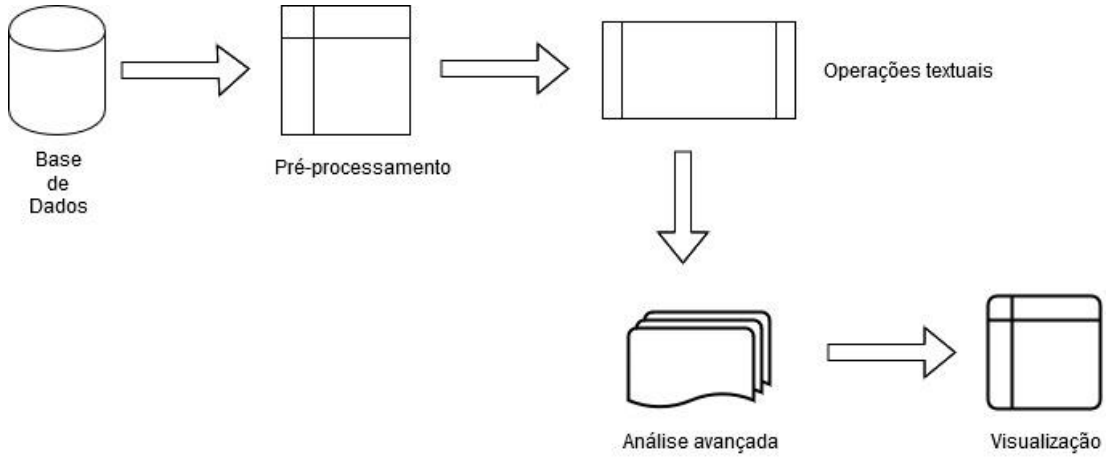


Figura 2: Fluxo genérico de análise de texto.

A base de dados pode ter fontes variadas, desde relações textuais provenientes de diferentes locais da internet até sistemas específicos empresariais ou pessoais que tenham interações textuais. A forma de obtenção destes textos também pode variar bastante.

O pré-processamento envolve a transformação dos dados, geralmente em formatos não ideais para leitura e entendimento, ou seja, dados não estruturados, em dados estruturados, de forma que seja simples e rápido a leitura e o entendimento dos mesmos de forma automatizada.

As operações textuais são os métodos aplicados nos textos com a finalidade de melhorar e otimizar o entendimento e a aplicação das metodologias estatísticas e de aprendizado de máquina. São diversas, e para cada caso existem operações que trazem melhores ou piores resultados.

A análise avançada é a aplicação direta das metodologias estatísticas e de aprendizado de máquina que podem ter diversos objetivos, desde buscar tendências de comportamento e padrões até análises de sentimento ou monitoramento de redes sociais.

A etapa de vizualização também pode ser bastante ampla, sendo gerada de inúmeras formas diferentes, dependendo da finalidade da aplicação. Muitas vezes, essa etapa não é aplicada por ser utilizada uma outra abordagem de aplicação dos resultados gerados.

De forma geral, a analise de texto é uma área extremamente ampla e que apesar de existir alguns padrões de aplicação do método, é necessário que cada caso seja analisado com base no objetivo desejado, a partir dos dados. O presente trabalho aplica uma metodologia que tem como base as etapas na Figura 2*,* mas traz modificações específicas necessárias para a obtenção de um melhor desempenho.

3.2 Descoberta de Conhecimento em Bancos de Dados (KDD)

Descoberta de Conhecimento em Bancos de Dados, ou *Knowledge-Discovery In Databases* (KDD), representa o processo de extração e pré-processamento dos textos, de maneira automatizada, que possa ser o mais fácil possível para a aplicação dos passos subsequentes.

Diretamente ligado ao ramo de mineração de dados, o KDD surgiu a partir da necessidade de se extrair dados a partir de uma fonte que tem um crescimento extremamente rápido em volume, e que se apresenta cada vez mais, de maneiras diferenciadas (HU *et al*., 2007). Ferramentas de extração mais simples buscam por dados diretamente em bancos de dados relacionais

ou em fontes textuais primárias, como arquivos de linguagem de marcação como XML (*Extensible Markup Language*) ou HTML (*HyperText Markup Language*), que são formatos para a criação de documentos com dados organizados de forma hierárquica, enquanto ferramentas mais complexas, onde permissão e acesso são exigidos, buscam por dados em diferentes camadas de bancos de dados e sistemas.

A Figura 3 e a Figura 4 mostram a diferença de uma fonte que contém dados não estruturados e esses mesmos dados apresentados em um formato estruturado. A Figura 3 descreve um *e-mail* enviado por um técnico de uma empresa à outro técnico, onde é descrito textualmente detalhes de um processo (dados não estruturados), e na sequencia, na Figura 4, estes mesmos dados são apresentados em um arquivo com extensão .csv, já com todas as variáveis de interesse filtradas.

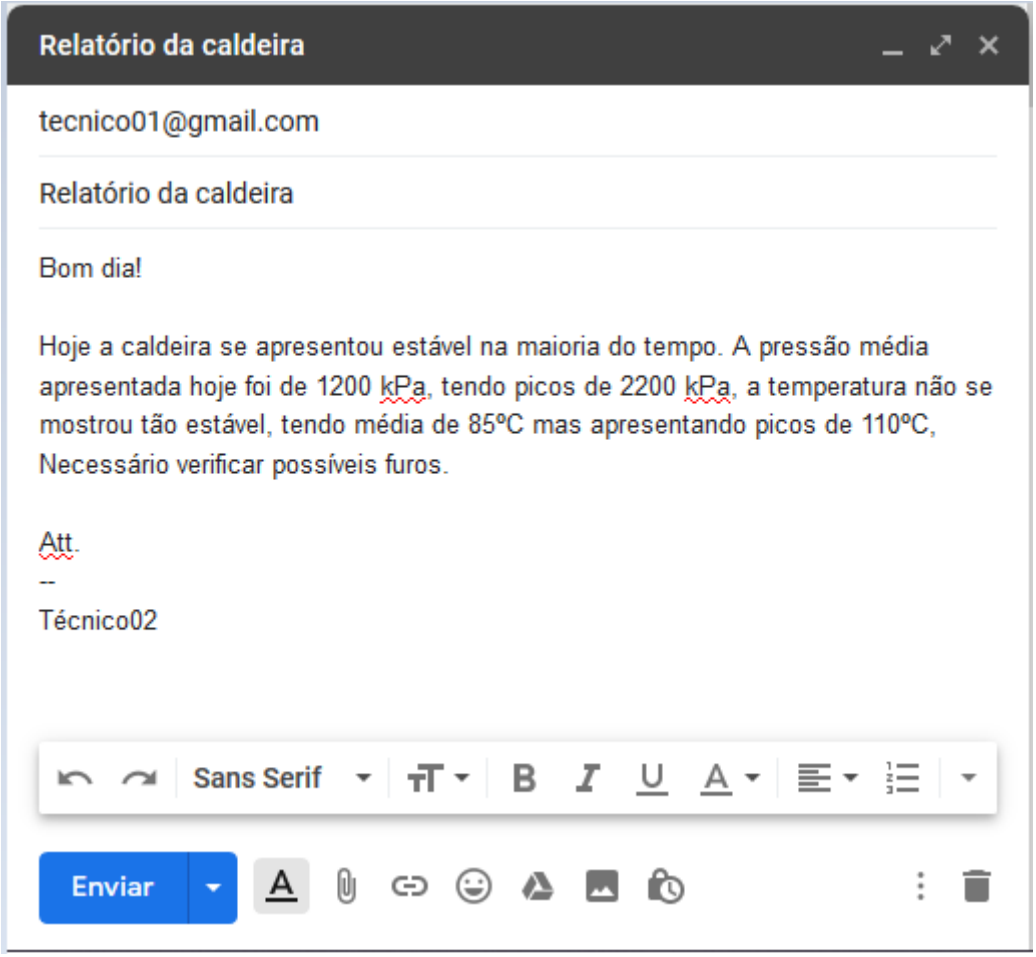


Figura 3: E-mail entre técnicos (dados não estruturados).

	A	B	C	D
1	caldeira estável maioria do tempo			
2	pressão média 1200 kPa			
3	pressão máxima 2200 kPa			
4	temperatura média 85°C			
5	temperatura máxima 110°C			
6				

Figura 4: Dados já transcritos para um arquivo com extensão .csv (dados estruturados).

3.3 Processamento de Linguagem Natural (NLP)

O processo de linguagem natural, ou *Natural Language Processing* (NLP), é um ramo presente em diversas áreas. Em geral, busca tratar questões como reconhecimento de fala, de entendimento e de criação de linguagem natural. Para se alcançar um desses objetivos, existe uma quantidade enorme de métodos e abordagens que podem ser aplicadas com o intuito de se criar esse processamento linguístico. O presente trabalho lida com alguns desses métodos e termos, conforme o objetivo inicialmente proposto, que serão explicados em detalhes nas próximas subseções.

3.3.1 Palavras de parada (*stop words*)

Palavras de parada, ou como são comumente conhecidas, *stop words*, são normalmente as palavras mais comuns existentes em uma determinada linguagem. Não existe uma lista universal ou um vocabulário prévio que defina quais são as *stop words* de uma determinada linguagem, e com base no contexto, palavras que inicialmente eram definidas assim podem deixar de ser.

Principalmente para mecanismos de busca na Internet, considerar ou não *stop words* é uma questão importante, pois isso afeta diretamente o resultado de um processo de busca. De maneira geral, não é possível afirmar que a retirada das *stop words* melhoram ou pioram uma busca, é necessário considerar o contexto em cada caso. (MUELLER, MASSARON, 2016)

Por exemplo, o mecanismo de busca da *Google* leva em conta tratativas das *stop words*. Atualmente, a frase buscada é quebrada em *tokens* e o resultado de cada *token* é analisado separadamente (KANNAN *et al*, 2019). *Tokens* são

vetores contendo cada um deles uma combinação de palavras distintas. Por exemplo, na busca da frase “Os fluxogramas de processo”, os seguintes *tokens* são gerados com base no processo de *tokenização* padrão (SHALEV, DAVID, 2014):

[Os fluxogramas de processo] – 2,43
[fluxogramas de processo] – 2,60
[Os fluxogramas processo] – 2,10
[fluxogramas processo] – 2,86

Feito isso, é analisado o resultado de busca de cada um dos *tokens*, onde é levado em conta não só a quantidade de resultados obtidos como também a frequência de acesso de cada um desses resultados. Nesse caso em questão, a quantidade de resultados obtidos é apresentada juntamente com os *tokens* (em milhões). Mesmo não tendo a maior quantidade de resultados de busca, o *token* [fluxogramas de processo] é aquele recomendado pelo *Google*, dada a análise da frequência de acesso. Assim, a *stop word* “de” mostra-se relevante na busca, deixando claro que a retirada de *stop words* depende de cada caso.

No presente trabalho, a retirada das *stop words* é realizada em duas etapas. A primeira usa a métrica TF-IDF (*Term Frequency-Inverse Document Frequency*), que é uma medida estatística sobre a importância da palavra em relação ao todo; essa importância é baseada na freqüência e significância do termo (ECK *et al*, 2005). A segunda usa a plataforma NLTK (*Natural Language ToolKit)* da linguagem de programação *Python* (LOPER, BIRD, 2002).

3.3.2 Stemização (*Stemming*) de palavras

Stemização é o processo de redução das palavras flexionadas em sua raiz semântica. Pode ser visto como uma técnica de normalização de palavras. O objetivo principal nesse caso é a redução do vocabulário e a abstração de significado (SANTOS *et al.*, 2015). Considere o grupo de palavras a seguir.

Andei - Ande - Andarei – Andamento - Andando - Andante

Todas as palavras podem ser reduzidas ao prefixo “And” ou à palavra “Andar”. Isso faz com que o conjunto de dados de palavras (*dataset)* se torne mais denso, reduzindo o tamanho do dicionário de palavras e as dimensões de entrada a serem trabalhadas. Como resultado, técnicas de análise estatísticas e de aprendizado de máquina têm melhor performance quando lidam com *datasets* que passaram pelo processo de stemização.

Além de reduzir as palavras aos seus radicais, a stemização também trata palavras que tenham letras maiúsculas e minúsculas e que contenham acentos ou hífens, tornando ainda mais simplificada e rápida a análise subsequente. A stemização também atua com grande eficiência em erros ortográficos. Por realizar a transformação das palavras em sua raiz semântica, isso acaba fazendo com que a grande maioria dos erros ortográficos existentes nas palavras sejam descartados juntamente com aquela parte que não faz parte da raiz semântica da palavra.

É uma técnica já consolidada no ramo de análise computacional linguística, com algoritmos da década de 1980 sendo amplamente usados até hoje, como por exemplo o Algoritmo de Porter (RAJARAMAN, ULMAN, 2011).

No atual trabalho, para a stemização nos textos estudados, foi utilizado o pacote de *stemming* da plataforma NLTK do *Python*.

3.3.3 Modelo N-Gram

Um *n*-grama é uma sequência de *n* itens de uma determinada fonte, podendo ela ser, por exemplo, um texto ou uma sequência de palavras. Dada uma sentença, podemos construir uma lista de *n*-gramas a partir desta sentença, encontrando grupos de palavras que ocorrem uma ao lado da outra (CAVNAR et al., 1994).

O prefixo “N” representa a quantidade de componentes que estes grupos terão, podendo ter dois componentes (bigrama), três (trigrama) ou mais. O processo é geralmente feito em todo o espaço amostral de palavras, levando em conta o acréscimo sequencial destas como mostra a Figura 5, que usou uma frase de um dos relatórios técnicos do estudo de caso do setor químico.

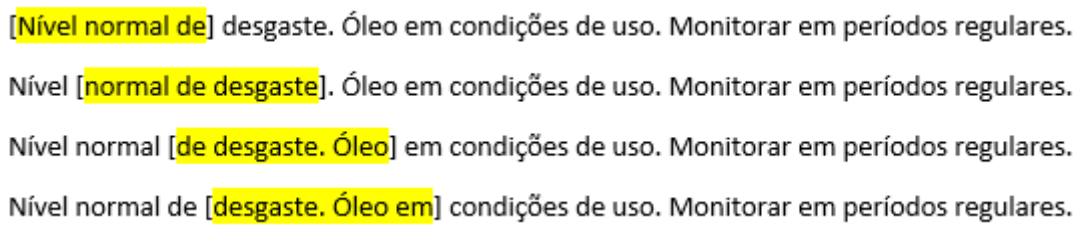


Figura 5: Exemplo de aplicação do N-Grama em um texto.

O N-Gram trabalha em conjunto com o cálculo de probabilidade que um dado n-grama, ou um conjunto de n-gramas, podem ocorrer naquele texto. Com essas probabilidades, é possível determinar, por exemplo, a probabilidade de uma tradução automática estar correta, de prever e sugerir a próxima palavra a ser escrita em uma frase, de gerar automaticamente texto a partir da fala, ou de automatizar um mecanismo de correção ortográfica (SIAU, WANG, 2018).

Existem diversas maneiras de se calcular a probabilidade e frequência de cada n-grama em um dado espaço amostral. O presente trabalho usa um método estatístico conhecido na literatura devido ao seu uso em conjunto com o N-Gram, o TF-IDF.

3.3.4 Métrica TF-IDF (*Term Frequency-Inverse Document Frequency*)

O TD-IDF é um método estatístico que tem como propriedade indicar quão relevante é um termo ou uma sequência de termos. Em outras palavras, é uma técnica de recuperação de informações que mensura a frequência de um termo (TF) e sua frequência inversa de documentos (IDF). Tendo cada palavra suas respectivas pontuações TF e IDF, o produto desses fatores recebe o nome de peso TF-IDF. Para um termo t em um documento d, o peso $W_{t,d}$ do termo t é dado pela Equação 1, em que $TF_{t,d}$ é o número de ocorrências de t no documento d, DF_t é o número de documentos contendo o termo t, e N é o número total de documentos analisados.

$$W_{t,d} = TF_{t,d} * \log\left(\frac{N}{DF_t}\right) \quad (1)$$

Quanto maior o valor do peso, mais raro é o termo; de modo contrário, um termo comum tem um peso baixo (ECK *et al.*, 2005). Por exemplo, quando um documento contendo 100 palavras contém o termo “energia” 12 vezes, o TF para a palavra energia será conforme a Equação 2.

$$TF_{energia} = \frac{12}{100} = 0{,}12 \quad (2)$$

Além disso, levando em conta neste exemplo que o termo “energia” apareça 300 vezes em um espaço amostral de 1000 documentos, o peso $W_{t,d}$ será calculado conforme a Equação 3. Esses valores de peso são usados em conjunto com o *N*-Gram onde são calculados os pesos de cada palavra contida nos n-gramas.

$$W_{t,d} = 0{,}12 * \log\left(\frac{1000}{300}\right) = 0{,}0627 \quad (3)$$

Uma medida de similaridade entre documentos é calculada a partir da distância de Jaccard, que é um dos modos mais intuitivos para esse tipo de tarefa pois mede a semelhança e a diversidade dos conjuntos de amostras, no caso, dos n-gramas gerados. (FERDOUS, 2009). O cálculo é realizado conforme a Equação 4, em que d1 e d2 representam diferentes documentos, $d1 \cap d2$ são o que d1 e d2 têm em comum e $d1 \cup d2$ representa todos os itens em d1 e d2 juntos. Um exemplo, dado d1 = [1 3 2] e d2 = [5 0 3], $d1 \cap d2$ = [3] e $d1 \cup d2$ = [0 1 3 2 5]. Logo, $J(d1, d2) = \frac{1}{5}$ = 0,2.

$$J(d1, d2) = \frac{d1 \cap d2}{d1 \cup d2} \quad (4)$$

Para este tipo de cálculo de medidas entre semelhanças dos conjuntos de amostra também é bastante comum o uso da distância euclidiana. Este método foi testado apresentando resultados piores, o que fez com que a distância de Jaccard fosse utilizada para esse trabalho.

3.3.5 Vetorização de palavras

Vetorização é a transformação de palavras em vetores numéricos. Normalmente, isso é usado para que as palavras possam ser representadas em um formato que possa ser matematicamente tratado.

Existem diversas maneiras de se implementar vetorização de palavras, muitas delas já bastante consolidadas na literatura, como por exemplo, *bag of words, word2vec, skip-thought vectors* e TF-IDF. Com o intuito de comparar a acurácia dos métodos no processo de vetorização, Stecanella (2017) testou diferentes métodos para um mesmo conjunto de dados variando o seu tamanho. O autor reportou que para conjuntos de dados com quantidades de textos acima de 4.000, o TF-IDF apresentou os melhores resultados.

Por esse motivo, neste trabalho, para realizar a vetorização, utilizou-se o método TF-IDF, que é mesmo modelo usado anteriormente em conjunto com a modelagem *N*-Gram. Utilizam-se os resultados da aplicação da Equação 1 para se construir um vetor de cada frase em que cada número no vetor é o resultado do cálculo do peso TF*IDF que representa, desse modo, a respectiva palavra no vetor da frase. Um exemplo de vetorização é colocado na Figura 6.

Óleo condições uso = [0,976; 0,745; 0,612]

Figura 6: Exemplo de vetorização de palavras.

3.4 Aprendizado de máquina

O aprendizado de máquina (em inglês, *machine learning*), usado para a construção de modelos, é um ramo da inteligência artificial baseado na ideia de que sistemas podem aprender a partir de experiências, ou seja, com dados, com o objetivo final de identificar padrões e tomar decisões com o mínimo de intervenção humana (MUELLER, MASSARON, 2016).

Existem inúmeras abordagens e métodos de aprendizado de máquina, que podem ser classificados como métodos de aprendizado supervisionado ou não-supervisionado. O que difere as duas classificações é a presença ou não de um rótulo ou classificador nos dados a serem analisados.

No atual trabalho, todos os dados são textuais e não contêm classificação prévia. Por esse motivo, para o alcance do objetivo proposto neste trabalho, são utilizados modelos de aprendizado de máquina não supervisionados.

3.4.1 Redução de dimensionalidade

A redução de dimensionalidade é uma técnica poderosa que é amplamente usada em análise de dados e ciência de dados para facilitar a visualização de dados, a seleção de características relevantes, e o treinamento de modelos com eficiência. Essa redução consiste na aplicação de métodos para obter dados em dimensões reduzidas do problema, a partir do conjunto original de dados em altas dimensões. Por exemplo, se existe um conjunto de dados com cem colunas (variáveis, ou características), a redução de dimensionalidade faz com que o número de colunas seja reduzido a um número significativamente menor, com a máxima preservação possível da informação original (MARSLAND, 2014).

Em resumo, é possível afirmar que a redução de dimensionalidade traz como beneficios, tornar o conjunto de dados mais fácil de ser usado, reduzir o custo computacional de diversos algoritmos, remover ruídos nos dados, e facilitar o entendimento dos resultados. Descrevem-se, a seguir, duas técnicas usuais de redução de dimensionalidade.

3.4.1.1 Análise de Componentes Principais (PCA)

Na Figura 7*,* tem-se, inicialmente, uma nuvem de pontos em um plano definido pelos eixos X e Y. Além disso, tem-se a representação de três eixos candidatos: A, B e C, que tentam cobrir a variação presente no conjunto de dados, sendo o segmento B, a melhor opção. Esse segmento é o resultado da rotação do eixo X, de um ângulo, segundo o critério de explicação de máxima variância. A segunda melhor opção nesse caso, ortogonal ao eixo B, é o eixo C, que explica o restante da variância dos dados originais. Desse modo, tem-se a rotação do sistema original de coordenadas, definido pelos eixos X e Y, que é definida, nesse caso, pelos eixos rotacionados B e C. Ao final, é possível explicar a maior parte da variância dos dados originais a partir de um número

relativamente menor de eixos (rotacionados), que são denominados de componentes principais, alcançando-se assim, a redução da dimensão do problema. Esse possibilidade facilita o tratamento e a visualização do problema. Esse procedimento é o princípio da técnica estaítica multivariada denominada, Análise de Componentes Principais (*Principal Component Analysis*; PCA) (HARRINGTON, 2012).

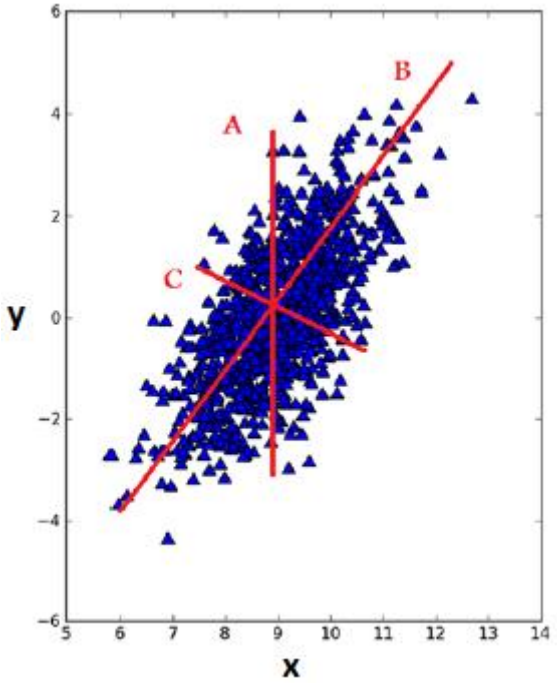
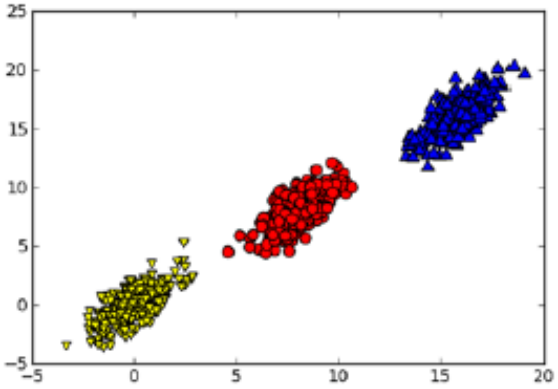


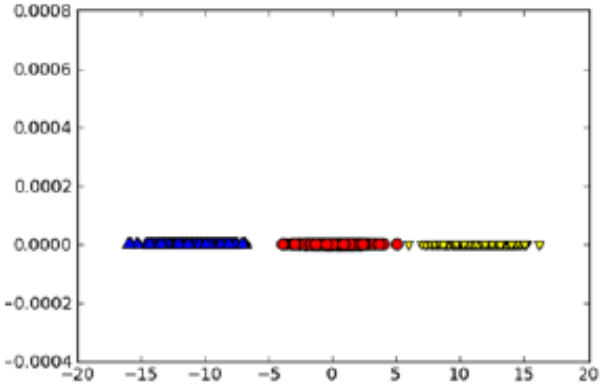
Figura 7: Opções de linhas que cobrem o conjunto de dados (Fonte: HARRINGTON, 2012)

Considerando a Figura 8(a), com três classes plotadas em um plano, ao reduzir a dimensionalidade desse conjunto de dados com PCA, pode-se obter um classificador unidimensional, representado pela Figura 8(b). Isto é, uma dimensão é suficiente para a descrição do problema, que é simples nesse caso; porém, de grande utilidade em problemas multivariados, como é o caso do presente trabalho. O ponto de partida para o cálculo dessa rotação é a matriz de variâncias e covariâncias do conjunto de dados originais, seguido da obtenção dos seus autovetores (as componentes principais) e autovalores associados (as respectivas variâncias das componentes principais). Para um problema *p*-dimensional, obtêm-se *p*-componentes princpais, onde deseja-se que apenas *k* delas, com $k << p$, sejam capazes de explicar a maior parte da variância dos dados originais. Lembra-se que o somatório das variáncias das

variáveis originais é igual ao somatório das variáncias das componentes principais, dadas pelos autovalores (HARRINGTON, 2012).



(a) Sistema de coordenadas original.



(b) Sistema de coordenadas, rotacionado via PCA.

Figura 8: Redução de dimensionalidade com PCA. (Fonte: HARRINGTON, 2012)

3.4.1.2 Mapas Auto-Organizáveis (SOM)

Os mapas auto-organizáveis (*Self-Organizing Maps*; SOM) contêm apenas duas camadas, a de entrada e a de saída. A camada de entrada representa o conjunto de dados do vetor de entrada X com dimensões I x N, em que I é o número total de amostras e N é o número de variáveis; desse modo, $x_i = [x_{i1}, x_{i2}, \ldots, x_{in}, \ldots, x_{iN}]$, $i = 1,2,3,\ldots,I$, é um vetor de observações, ou amostra. A camada de saída é uma coleção ordenada de neurônios, usualmente organizados como reticulados hexagonais ou retangulares. A rede hexagonal é geralmente preferida porque oferece uma melhor visualização. Cada neurônio da camada de saída é conectado à camada de entrada através de vetores de

pesos (vetores de referência). A dimensionalidade do vetor de pesos de cada neurônio de saída é a mesma que a dimensionalidade dos vetores de entrada, portanto, $w_j = [w_{j1}, w_{j2},\ldots, w_{jn}, \ldots, w_{jN}]$, j = 1,2,3,…,J, onde J é o número total de neurônios no mapa reticulado (KLOBUCAR, SUBASIC, 2012).

Na etapa de treinamento, os vetores de pesos dos neurônios da camada de saída são comparados com os vetores de entrada, dados pelas amostras, a partir de uma medida de distância, a fim de se determinar o grau de ativação dos neurônios. A distância Euclidiana é a métrica geralmente usada como critério de comparação entre o vetor de entrada e o vetor de pesos dos neurônios de saída. Após a comparação, o neurônio cujo vetor de referência apresenta a menor distância de x_i é definido como o neurônio vencedor, que também é chamado de melhor unidade de correspondência (BMU) para tal entrada, como mostra a Equação 5.

$$b_i = argmin \left\| x_i - w_j \right\| para\ i = 1, 2, 3, \ldots, I\ e\ j = 1, 2, 3, \ldots, J \qquad (5)$$

Após a seleção do neurônio vencedor, o seu vetor de pesos é atualizado por certa quantia de modo a aproximá-lo do vetor de entrada em questão. Além disso, os vetores de pesos dos neurônios vizinhos também são atualizados; porém, em menor grau. Como consequência desses ajustes, os vetores de peso atualizados tornam-se mais semelhantes ao vetor de entrada. A função de atualização dos pesos é mostrada na Equação 6, onde $\alpha(t)$ é o parâmetro de taxa de aprendizagem e $h_{bj}(t)$ é a função de vizinhança.

$$w_j(t+1) = w_j(t) + \alpha(t) h_{bj}(t) \left\| x_i(t) - w_j(t) \right\| \qquad (6)$$

Geralmente, o parâmetro de taxa de aprendizagem e a função de vizinhança são gradualmente decrescidos com o avanço do processo de treinamento. Considerar o *j*-ésimo neurônio de saída ser r_j, e r_b ser o neurônio vencedor no mapa de saída. Uma escolha popular para a função de vizinhança é a função gaussiana (Equação 7), onde $\sigma(t)$ é a largura da vizinhança dado o neurônio vencedor. O parâmetro σ(t) diminui com o tempo de modo a controlar a atualização dos neurônios vizinhos no tempo *t*. Para garantir a convergência,

$\sigma(t)$ e $\alpha(t)$ (taxa de aprendizagem) são inicialmente dotados de um grande valor e depois decrescidos monotonicamente com t. Quando $t \rightarrow \infty$, então $\alpha(t) \rightarrow 0$ e $\sigma(t)$ se aproxima de um valor pequeno (tipicamente 1).

$$h_{bj}(t) = \exp\left(\frac{\|r_b - r_j\|^2}{2\sigma^2(t)}\right) \quad (7)$$

A fase de treinamento termina quando o número de iterações excede o número máximo prédeterminado de iterações, e os neurônios no mapa de saída serão então rotulados com seus nomes de entrada correspondentes. Em resumo o algoritmo do SOM pode ser explicado em seis passos (CHEN, YAN, 2012), conforme a seguir:

1. Inicialização dos vetores de pesos de todos os neurônios do mapa; definição do parâmetro de vizinhança $h_{bj}(t)$ e da taxa de aprendizagem $\alpha(t)$.
2. Escolha de um vetor de entrada de forma aleatória.
3. Cálculo da distância entre o vetor de entrada e os vetores de pesos para todos os neurônios do mapa; então, encontrar o neurônio vencedor segundo a Equação 5 e atualizar os pesos de todos os neurônios na região do neurônio vencedor, segundo a Equação 6.
4. Atualização dos valores do parâmetro de vizinhança $h_{bj}(t)$ e da taxa de aprendizagem $\alpha(t)$.
5. Teste dos critérios de parada. Se o critério de parada for satisfeito, então, seguir para o passo 6, caso contrário, repete-se o processo a partir do passo 2.
6. Rotula-se o dado de entrada no mapa de forma que cada vetor de entrada busque por seu neurônio vencedor para que o rótulo do dado de entrada seja feito nesse neurônio.

Existem dois parâmetros que devem ser especificados: o número de neurônios no mapa (J) e a relação de aspecto da grade bidimensional. O número dos neurônios determina a precisão e a capacidade do modelo SOM. O tamanho maior do mapa terá o menor erro de quantização que é uma

medida de quão bom o mapa pode caber nos dados de entrada; porém, maior o erro topográfico, que mede quão bem a topologia é preservada pelo mapa, e maior o custo computacional. Uma solução razoável do número de neurônios no mapa reticulado é a fórmula heurística segundo a Equação 8. Já a relação de aspecto é especificada pela raiz quadrada da razão entre os dois maiores autovalores da matriz de variâncias e covariâncias de X (XINYI, XUEFENG, 2013).

$$J = 5\sqrt{I} \tag{8}$$

3.4.2 Análise de agrupamentos (Clusterização)

A análise de agrupamentos (*clustering*) é um tipo de aprendizado não supervisionado que busca a identificação de agrupamentos de itens semelhantes (SCULLEY, 2010). A principal diferença com a classificação é que nesta, sabe-se, *a priori*, o que está sendo procurado. No atual trabalho, serão abordados dois métodos de clusterização, o *k-means* e o DBSCAN.

3.4.2.1 Algoritmo *k*-means

O *k*-means (*k*-médias) é um algoritmo de aprendizado não supervisionado usado para encontrar grupos similares em um dado conjunto de dados (SCULLEY, 2010).

Para aplicar o método, primeiro define-se a quantidade de agrupamentos (*k*) que se deseja obter. Em seguida, inicia-se no espaço de busca, usualmente de modo aleatório, *k* centróides. Em seguida, para cada ponto no mapa, encontra-se o centróide mais próximo. O ponto é atribuído para o *cluster* referente a esse centróide. Feita essa etapa para todos os pontos, calculam-se os novos centróides através das médias dos pontos anteriormente alocados nos respectivos agrupamentos. Já com os novos centros, repete-se essa etapa de modo iterativo, até que seja alcançado o número máximo de iterações ou não ocorram realocações de pontos entre os agrupamentos. Desse modo, tem-se, ao final, os centróides finais dos agrupamentos (HARRINGTON, 2012).

O *k*-means tem a vantagem de ser um algoritmo de fácil implementação, e a desvantagem é que é lento para grandes conjuntos de dados, além de poder convergir para mínimos locais.

3.4.2.2 Algoritmo DBSCAN

O Agrupamento Espacial de Aplicações com Ruído baseado em Densidade (*Density-Based Spatial Clustering of Applications with Noise*; DBSACN) é um algoritmo de clusterização de dados comumente utilizado em mineração de dados e aprendizado de máquina.

Com base em um conjunto de pontos assumindo um espaço bidimensional, o DBSCAN agrupa pontos próximos uns dos outros com base em uma medida de distância (geralmente a distância Euclidiana) e um número mínimo de pontos. Também define como discrepantes os pontos que estão em regiões de baixa densidade (HAHSLER, 2019). Desse modo, o algoritmo requer basicamente dois parâmetros, conforme a seguir.

- **ε:** Também chamado de eps, específica quão próximos os pontos devem ser levados em conta para serem considerados parte de um *cluster*. Isso significa que, se a distância entre dois pontos for menor ou igual a esse valor (ε), esses pontos serão considerados vizinhos
- *minPts*: É o número mínimo de pontos para formar uma região densa. Por exemplo, se for definido o parâmetro *minPts* como 5, serão necessários pelo menos 5 pontos para a formação de uma região densa.

Usando ε e *minPts*, é possível classificar cada ponto conforme a seguir.

- Ponto principal: Um ponto que possui pelo menos um número mínimo de outros pontos (*minPts*) dentro do seu raio ε.
- Ponto de fronteira: Um ponto está dentro do raio ε de um ponto central; porém, tem menos do que o número mínimo de outros pontos (*minPts*) dentro de seu próprio raio ε.
- Ponto de ruído: Um ponto que não é nem um ponto central nem um ponto de borda.

A Figura 9 apresenta um exemplo com *minPts* = 3, onde os pontos quadrados são classificados como pontos principais, os pontos redondos cheios são classificados como pontos de borda, e os pontos redondos vazados são classificados como pontos de ruído (HAHSLER, 2019).

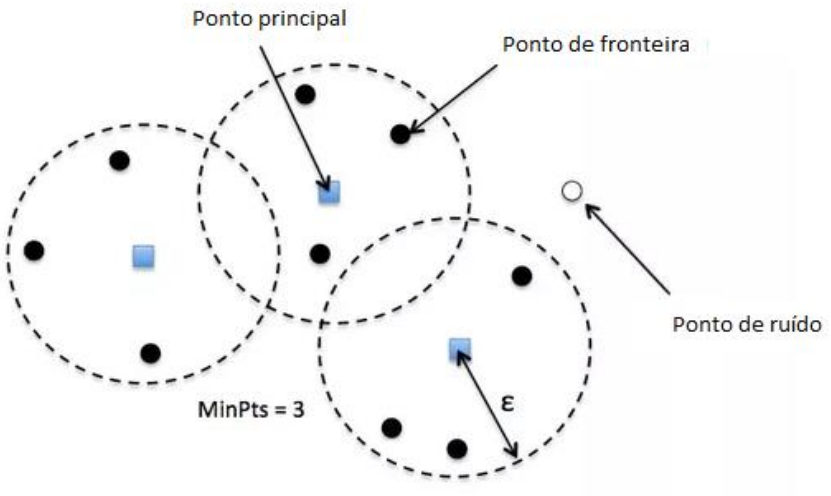


Figura 9: Exemplo aplicação DBSCAN

A estimativa de parâmetros é um tópico importante nesse caso, sendo importante ter um conhecimento prévio básico sobre o conjunto de dados que será usado (HAHSLER, 2019).

- ε: Se o valor eps escolhido for muito pequeno, grande parte dos dados não será agrupada. Será considerado discrepante porque não satisfaz o número de pontos para criar uma região densa. Por outro lado, se o valor escolhido for muito grande, os *clusters* serão mesclados e a maioria dos objetos estará no mesmo *cluster*. O parâmetro ε deve ser escolhido com base na distância do conjunto de dados (pode-se usar um gráfico de distância k para encontrá-lo), mas, em geral, pequenos valores de ε são preferíveis.
- *minPts*: Como regra geral, um mínimo de *minPts* pode ser derivado de várias dimensões (D) no conjunto de dados, como *minPts* ≥ D. Valores maiores geralmente são melhores para conjuntos de dados com ruído e formarão *clusters* mais significativos. O valor mínimo para os *minPts*

deve ser 3, mas, quanto maior o conjunto de dados, maior o valor de *minPts* que deve ser escolhido.

3.5 Métricas de performance

O intuito com o uso de métricas de performance ou de desempenho é determinar a qualidade dos resultados obtidos pelas técnicas de clusterização, o que é uma questão fundamental em qualquer atividade de aprendizado de máquina. Muitos autores discutem os recursos desejáveis de bons algoritmos e como mensurar a qualidade dos resultados obtidos por estes algoritmos (DINH *et al*. 2019).

Para algoritmos de aprendizado de máquina não supervisionados, uma métrica bastante usada e eficiente é o coeficiente de *silhouette* (DINH *et al*, 2019). Esse coeficiente é uma métrica que não precisa saber a rotulagem inicial do conjunto de dados. Ele é composto basicamente por dois elementos, conforme a seguir.

- A distância média entre uma amostra e todos os pontos da mesma classe (a).
- A distância média entre uma amostra e todos os pontos no *cluster* mais próximo (b).

Os parâmetros a e b estão visualmente representados na Figura 10.

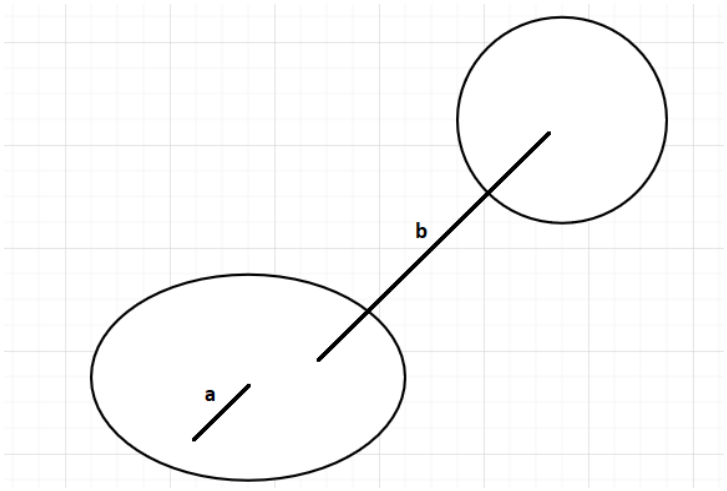


Figura 10: Exemplo de *clusters* para o cálculo do coeficiente de silhouette.

A Equação 9 define o cálculo do coeficiente que é representado por s. O valor do coeficiente varia de -1 a +1, onde um valor alto (próximo de 1) indica que o dado é bem alocado em seu próprio *cluster*, e mal alocado com relação aos *clusters* vizinhos. É importante salientar que o coeficiente de *silhouette* mostrado nos resultados é a média dos coeficientes obtidos por todos os pontos do espaço amostral calculado para aquela configuração de métodos definida.

$$s = \frac{b-a}{\max(a,b)} \tag{9}$$

4. METODOLOGIA

A metodologia proposta no presente trabalho foi aplicada visando obter cada um dos objetivos específicos, e desse modo, alcançar o objetivo geral.

Os experimentos foram executados em máquinas virtuais *Standard* E2 v3 do Azure em execução no Ubuntu 18.04. Suas configurações tinham uma vCPU de dois núcleos baseados em um processador XEON E5-2673, 16 GB de RAM, um HD de 30 GB e 32 GB de memória de troca em um HD separado. A implementação foi feita usando o *Python* 3.7.3, com as bibliotecas *scikit-learn* 0.21.3, *numpy* 1.16.4, *pandas* 0.24.2, *scipy* 1.3.0 e *minisom* 2.1.9.

Como o trabalho é dividido em duas frentes, um problema envolvendo o setor elétrico e outro envolvendo o setor químico, uma metodologia em comum foi criada com o intuito de atender ambos os casos, mas particularidades de cada caso foram acrescentadas a fim de se obter um melhor resultado.

A Figura 11 representa a metodologia geral do processo, que reúne os pontos descritos anteriormente na Revisão bibliográfica. As particularidades serão discutidas nas respectivas sessões a seguir.

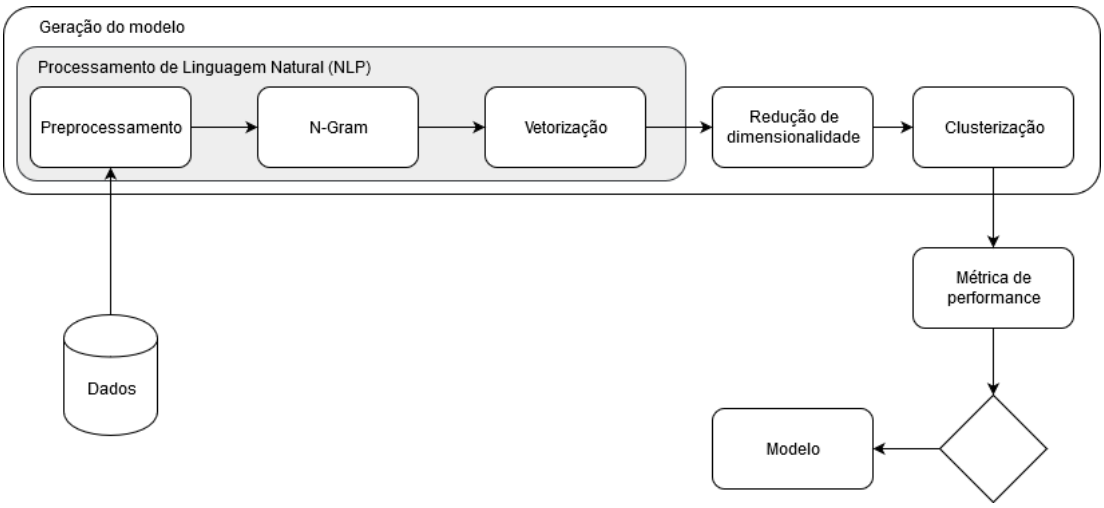


Figura 11: Metodologia geral do processo.

4.1 Setor Elétrico

4.1.1 Visão geral do problema

As concessionárias de energia que formam o Sistema Brasileiro Integrado de Energia estão divididas em três grandes setores: geração, transmissão e distribuição. Entre esses, o setor de distribuição é o único responsável pelo relacionamento direto com o cliente. Portanto, o setor de distribuição no Brasil é o responsável pela melhoria contínua dos canais de comunicação com os clientes, o que é necessário para um melhor engajamento dos usuários. A concessionária dessa aplicação já mantém muitos canais de comunicação diferentes, com o objetivo de melhorar e facilitar a comunicação com os clientes.

Com o intuito de facilitar e melhorar o relacionamento com o cliente, a empresa tem buscado criar um *framework* que esteja integrado através de um *chatbot* em todos os canais de comunicação através de uma arquitetura *ominichannel*, que consiga atender aos clientes de forma direta e fluida.

Para isso, inicialmente, é necessário realizar um estudo do comportamento dos clientes em relação aos canais de comunicação, já com o objetivo voltado para a criação do *chatbot* (a criação desse robô está além do escopo deste trabalho). Esse estudo teve como ideia a busca de *entities* e *intents* no conjunto de dados de mensagens de texto trocadas entre a empresa e seus usuários via SMS e redes sociais. A Figura 12 exemplifica os termos, *utterances*, *entities* e *intents*, conforme as definições a seguir.

- *Utterance*: Qualquer expressão que o usuário disser. Por exemplo, se um usuário digitar "estou sem luz em minha casa", a frase inteira será uma *utterance*.
- *Intent*: Uma *intent* é a vontade do usuário. Por exemplo, se um usuário digitar "me mande a segunda via da conta ", a vontade do usuário é conseguir uma outra via da conta de energia. As *intents* recebem um nome, geralmente um verbo e um substantivo, como "segundaVia".
- *Entity*: Uma entidade modifica uma *intent*. Por exemplo, se um usuário digitar "me mostre meu histórico de conta", as entidades serão

"histórico" e "conta". As entidades recebem um nome, como "dateTime" e "billType". Às vezes, as entidades são chamadas de *slots*.

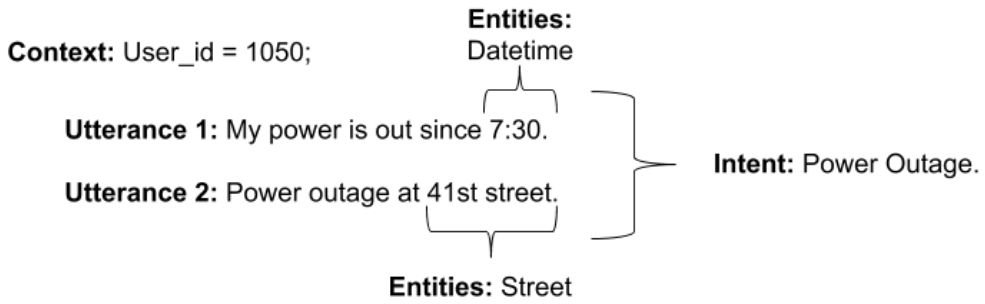


Figura 12: Relação entre *intents*, *entities* e *utterances*.

A busca de *intents* já é uma abordagem conhecida para o entendimento do comportamento de clientes. Historicamente, essa tarefa já é feita por várias empresas de diferentes segmentos com o intuito de melhor entender o comportamento dos clientes, só que de forma manual por pessoas especializadas em análise textual e busca de *intents*. Só que essa busca se tornava lenta, exaustiva e ocupava várias pessoas em uma única tarefa por ter normalmente um volume muito grande de texto a ser analisado. Nos últimos anos com a melhora do processamento computacional, o processamento de linguagem natural se tornou cada vez melhor e mais usual. Inclusive, mecanismos automáticos de processamento de linguagem natural como o LUIS da Microsoft e o Watson da IBM, que foram desenvolvidos e aprimorados nos últimos anos, usam desses conceitos para auxiliar a processar e identificar o que o cliente quis dizer (RAHMAN, 2017). Em resumo, o objetivo do presente trabalho é identificar *entities* e *intents* com base no conjunto de mensagens de SMS e de redes sociais entre os clientes e a empresa, por meio de uma abordagem não supervisionada, uma vez que não se tem *a priori* qualquer classificação das mensagens. É importante salientar que no atual trabalho, depois de encontrar os *intents,* estes passaram por um processo de confirmação e validação com profissionais especializados da área.

4.1.2 Conjunto de dados de textos

O conjunto de dados analisado compreendeu o ano completo de 2018. Ele foi extraído de mensagens SMS e em redes sociais, coletadas da interação entre clientes e um *chatbot* da empresa restrito a algumas funções, que até então reconhecia apenas palavras-chave sem empregar processamento de linguagem natural (NLP). Esse *chatbot* simples possui três intenções diferentes, e um conjunto com 2.383.296 mensagens. As intenções usadas são, "Luz", "Leitura" e "Conta". "Luz" significa que o usuário está relatando uma queda de energia; "Leitura" significa que o usuário deseja enviar sua própria leitura do medidor de energia; e "Conta" significa que o usuário deseja conhecer sua dívida atual com a empresa. A maioria dos usuários sabia que estava conversando com um *chatbot* simples; porém, é possível observar a construção inadequada de frases. Essa é mais uma motivação para o uso de NLP, de modo a não ser necessário um *input* rígido pelo usuário para essas frases.

Para a execução do presente estudo, apenas os dados dos casos que falharam foram utilizados, pois somente nesses casos podem haver novas palavras chave. Esses casos representam 33,1% do conjunto total de mensagens, ou seja, 789.347 mensagens. Uma falha ocorre quando o *chatbot* não consegue localizar a *intent* ou as *entities*, quando falha ao processar uma *entity* ou por qualquer outro motivo, como falha em envio de SMS ou localização de banco de dados *off-line*.

Foi decidido *a priori*, como valor de entrada, o uso de 2.500 e 5.000 frases. A razão é que o número de mesagens mostrou-se plausível para um tempo de processamento que não inviabilizasse o desenvolvimento do trabalho, pois através de testes empíricos, foi possível perceber que o tempo computacional crescia exponencialmente em função do número de frases usadas, ao mesmo tempo que, conforme a literatura, representa uma quantidade considerada significativa para um estudo (CHATTOPADHYAY *et al.*, 2011

A estrutura de cada mensagem registrada no conjunto de dados contém uma mensagem enviada pelo cliente, e na sequência, a mensagem de retorno da empresa produzida pelo *chatbot*. A Tabela 1 contém alguns exemplos dessas mensagens. Na primeira coluna, encontram-se as mensagens enviadas pelos clientes, e na segunda coluna, a resposta dada pela empresa.

Tabela 1: Exemplos de mensagens trocadas entre clientes e a empresa.

Mensagem enviada pelo cliente	**Resposta da concessionária**
Estao vindo da onde?	Que tipo de atendimento voce deseja? 1-Luz 2-Leitura ou 3-Conta Responda grátis com sua opção
Falta energia luz	Envie o numero do CPF do titular da conta (Ex: 33344455566) ou CNPJ para empresas (Ex: 11222333444455)
Luz045976597-5	Numero invalido. Envie o numero do CPF do titular da conta (Ex: 33344455566) ou CNPJ para empresas (Ex: 11222333444455)
Luz 333657416	Nao foi possivel abrir o chamado. Uma manutencao esta sendo executada ou ja existe um chamado aberto para sua regiao.

4.1.3 Pré-processamento de dados de texto

O pré-processamento dos dados é feito a partir de duas etapas: a remoção das *stop words* e a aplicação do *stemming*.

Primeiramente, o conjunto de dados é filtrado, removendo as *stop words*. Isso descarta possíveis discrepâncias gramaticais e semânticas, diminui o ruído e reduz o custo computacional das etapas subsequentes, que são fatores que, de alguma maneira, influenciam os resultados de clusterização. A execução dessa etapa foi feita usando o pacote do *Python* chamado NLTK (*Natural Language ToolKit*), que é bastante conhecido por ser referência nesse tipo de aplicação (LOPER, BIRD, 2002).

O processo de retirada de *stop words* no presente trabalho leva em conta uma análise feita através de dois métodos. Primeiro é feita uma análise de frequência das palavras através do uso do TF-IDF, onde os resultados são comparados com os resultados gerados pelo NLTK, que também faz uma

análise de frequência, mas levando em conta também a posição da palavra no texto. As *stop words* comuns nos dois métodos são retiradas.

A seguir, mostra-se o vetor com todas as palavras que foram consideradas *stop words*, e portanto, retiradas das mensagens. Uma observação é que, no dado contexto, a palavra “CPF” foi considerada uma *stop word*, podendo ser retirada sem afetar o resultado semântico da análise.

```
stop_words = ['de', 'a', 'o', 'que', 'e', 'é', 'do', 'da', 'em', 'um',
'para', 'com', 'não', 'uma', 'os', 'no', 'se', 'na', 'por', 'mais',
'as', 'dos', 'como', 'mas', 'ao', 'ele', 'das', 'à', 'seu', 'sua',
'ou', 'quando', 'muito', 'nos', 'já', 'eu', 'também', 'só', 'pelo',
'pela', 'até', 'isso', 'ela','cpf', 'tiver', 'tivermos', 'tiverem',
'terei', 'terá', 'teremos', 'terão', 'teria', 'teríamos', 'teriam'])
```

Em seguida, é feita a etapa do *stemming*. Neste estágio, cada palavra passa primeiro por um mecanismo de derivação. Isso consiste em remover sufixos ou prefixos com o intuito de transformá-la em sua raiz. Esse processo é feito usando uma biblioteca específica de *machine learning* para *Python* chamada *scikit-learn*, onde existe uma subrotina chamada CountVectorizer. É importante pontuar que a metodologia proposta neste trabalho foi testada com e sem a aplicação do *stemming*, com o objetivo de comparação de resultados.

4.1.4 Model *N-Gram*

Esta etapa também é realizada em dois passos: primeiramente, se aplica o *N-Gram*, e na sequência, o TF-IDF, que é utilizado em conjunto com o *N-Gram*.

A biblioteca usada para a criação dos *n*-gramas é a mesma do pré-processamento, a *CountVectorizer* do *scikit-learn*. Esse processo é realizado em todas as mensagens do conjunto de dados, sendo investigados os formatos de bigrama e de trigrama. A Tabela 2 mostra um exemplo desse processo com a frase “Não foi possível abrir o chamado”, a partir de um conjunto de dados com 2500 frases.

Tabela 2: Frequência de bigramas por conjunto de dados.

Bigrama	**Frequência de ocorrência**

Não foi	511
foi possível	530
possível abrir	428
abrir chamado	622

Na sequência, é calculada a métrica TF-IDF, onde é aplicada a Equação 1 para cada palavra. Antes de seu cálculo, é realizada uma análise de frequência de cada bigrama em relação a todo o conjunto de dados. No presente trabalho não foi aplicado nenhum ponto de corte sobre a frequência.

4.1.5 Vetorização de *N-Grams*

O processo de vetorização acontece em conjunto com a aplicação do *N-Gram* e do TF-IDF. Inclusive, a biblioteca do *scikit-learn*, *CountVectorizer* já possibilita, a partir de seus parâmetros de entrada, que o processo de vetorização de palavras ocorra em conjunto. Neste processo, cada palavra recebe o seu respectivo valor proveniente da Equação 1, e esse valor substitui cada palavra em seu bigrama. Na implementação em *Python*, a chamada feita para a vetorização é a mesma feita para a criação dos *n*-gramas e para o cálculo do TF-IDF.

É importante colocar que é possível, através dos parâmetros dessa chamada, limitar a similaridade entre os documentos (*n-grams*) através dos limites máximo e mínimo para a equação de distância de Jaccard (Equação 4). No presente trabalho, não foi definido um limite mínimo, e o limite máximo definido foi 1,0, ou seja, mensagens iguais têm sua quantidade total reduzida a apenas uma mensagem no processo de vetorização. Foi feito dessa forma para garantir a permanência de mensagens similares que abordam o mesmo tipo de problema, mas evitar possíveis mensagens automáticas que, em alguns casos, podem ocorrer com frequência devido a algum problema técnico no próprio sistema de mensagens.

4.1.6 Redução de dimensionalidade

A etapa de redução de dimensionalidade foi testada com dois métodos, de modo independente: análise por componentes principais (PCA) e mapas auto-organizáveis (SOM).

Os parâmetros de entrada e saída são essenciais para um bom desempenho de ambos os métodos. Não existe um procedimento geral que determine para um dada dimensão de entrada, qual a melhor dimensão de saída. Ainda assim, existem algumas diretrizes que podem ser tomadas como base de decisão.

Na aplicação do PCA, para o conjunto com 5.000 mensagens, e uma vetorização com dimensão 230, foram selecionadas 150 componentes principais, que juntas contribuem com 85% da variância todos dos dados originais.

Já na aplicação do SOM, para o mesmo número de mensagens, deve ser determinado *à priori* o tamanho do mapa de neurônios que representam a saída do processo de redução de dimensionalidade. A Equação 8 é uma heurística usada para determinar a quantidade inicial de neurônios da camada de saída (KLOBUCAR, SUBASIC, 2012). Apesar do resultado do cálculo da Equação 8 ser de 354 neurônios, foi usado um mapa de saída com dimensões 50x10, ou seja, 500 neurônios. Esse acréscimo foi feito de forma empírica pela percepção de resultados melhores. Conforme Villmann (1998), o tamanho do mapa e o tipo dos dados de entrada são variáveis que podem tornar a aplicação do SOM inviável dependendo do problema.

Para o caso de 2.500 mensagens, foram usados, para o PCA e para o SOM, 125 componentes principais, e 250 neurônios, respectivamente.

Para a aplicação do PCA, foi usado um pacote específico da biblioteca em *Python*, chamado *sklearn.decomposition*. Já para a aplicação do SOM, foi usada a biblioteca oficial em *Python* chamada *minisom*.

4.1.7 Clusterização de *N-grams*

A etapa de clusterização também foi feita usando duas metodologias diferentes, de modo independente: o *k-means*, o Mini-batch *k*-means, que é uma varinate do *k*-means, e o DBSCAN.

Em relação ao *k-means*, investigaram-se números de agrupamentos (k) iguais a 5, 7, 10, 15 e 20. O limite máximo do valor de *k* foi colocado em 20; pois não foram obtidos melhores resultados acima desse valor. Além disso, foi utilizado também uma variação do *k*-means, o Mini-batch *k*-means, que é um algoritmo de agrupamento que usa pequenos lotes de dados para reduzir o tempo de processamento computacional. A partir do conjunto de dados originais, geram-se subconjuntos de dados de maneira sucessiva e aleatória, processando-se o *k*-means em cada iteração de cada subconjunto (SCULLEY, 2010).

Para o DBSCAN, foi utilizada uma distância máxima entre documentos (eps) igual a 0,3, 0,5, 0,7, 1, 5, 10 e 15. A diferença entre esses valores distintos de eps foi definida de acordo com as variações de sensibilidade entre os resultados obtidos com o PCA e SOM na etapa anterior de redução de dimensionalidade, conforme sugestão em Hemavathi *et al.* (2019). Tendo como parâmetro para definir os valores de eps, o comportamento dos dados de entrada para diferentes dimensões de clusterização. Valores menores (entre 0,3 e 1) foram adicionados em maior quantidade pois a sensibilidade dos resultados se mostraram maiores entre eles. Valores de eps acima de 15 não foram utilizados, pois a sensibilidade dos resultados mostraram-se bastante baixas a partir desse valor.

Para a aplicação tanto do *k*-means, e sua variação em mini-batch, quanto do DBSCAN, foi usado o pacote *sklearn.cluster* da biblioteca *Python sklearn*.

4.1.8 Métrica de performance

Usando os resultados gerados como entrada, é então calculado o coeficiente de *silhouette*, comumente utilizado em aplicações de clusterização. A sua aplicação é feita usando, como parâmetros de entrada, os resultados das clusterizações dos algoritmos *k*-means e DBSCAN, sendo a métrica usada para o cálculo das distâncias, a distância Euclidiana.

Para a aplicação do método, é usado um pacote da biblioteca *sklean* chamado *sklearn.metrics.silhouette_score*.

4.2 Setor Químico

A metodologia aplicada no caso do setor químico, foi feita de maneira similar ao do setor elétrico, com um conjunto de dados de texto de uma empresa do setor químico. É importante salientar que todo o procedimento foi realizado na mesma máquina, com os mesmos requisitos descritos anteriormente.

4.2.1 Visão geral do problema

No estudo realizado na área da indústria química, tem-se uma empresa de vendas de máquinas para os setores de construção, mineração, energia e petróleo, que periodicamente realiza análises de óleo das máquinas vendidas. Estas análises geram relatórios com pareceres em formato texto. As sessões a seguir discutirão, mais detalhadamente, a metodologia proposta para a padronização desse processo de análise laboratorial.

A empresa realiza em torno de 100 análises diárias. Tem-se a análise de diversas características físico-químicas do óleo, relacionados com a presença de contaminantes químicos, de partículas indesejáveis, de fuligem, entre outras características. O detalhamento das soluções oferecidas para análise de óleo pode ser descrito conforme a seguir.

- Condições físico-químicas do óleo lubrificante
- Elementos químicos metálicos
- Contaminação de óleo
- Presença de partículas metálicas e não metálicas
- Líquido arrefecedor
- Interpretação de resultados com foco na manutenção do equipamento

A empresa tem uma grande equipe de analistas que, após cada análise, deve fornecer um parecer técnico, por escrito. Essas análises técnicas geralmente contêm diretrizes, como por exemplo, sobre a necessidade de troca de óleo ou a necessidade de realização de uma nova coleta, devido a inconclusões na interpretação dos resultados. Essas diretrizes são feitas por

escrito, sem uma padronização exata. Por esse motivo, existem frases escritas de maneira diferente; porém, que dizem respeito à uma mesma intenção. Na prática, esse fato pode gerar interpretações diferentes.

Para evitar essa interpretação errônea, o presente trabalho propõe o uso da metodologia descrita anteriormente, que se baseia no uso de técnicas de aprendizado não supervisionado, para identificar intenções no texto. Esse resultado será usado como ponto de partida para a padronização de sentenças nos relatórios técnicos das análises laboratoriais (esse processo de padronização está além do escopo deste trabalho).

4.2.2 Conjunto de dados de texto

O conjunto de dados obtido foi fornecido pela empresa já mencionada. Existem 48.212 amostras, entre janeiro de 2017 e agosto de 2019, cada um das quais é um relatório técnico elaborado por um analista. Cada relatório é escrito em um único parágrafo e é orientado por diretrizes fornecidas em relação ao resultado da análise de óleo. A seguir, exemplos de pareceres.

- "Nível normal de desgaste. Óleo em condições de uso. Monitor em intervalos regulares. Amostra anormal devido à falta de horas de óleo. Esta informação interfere na avaliação. Informe os dados sobre o tipo de óleo usado para melhor avaliação e monitoramento dos resultados."
- "Verifique os procedimentos de coleta e identificação de amostras. Resultados discrepantes. Envie uma nova amostra imediatamente para confirmação dos resultados. A amostra enviada não possui características de óleo de transmissão."
- "A contaminação excessiva da água pode causar degradação do óleo e falha na lubrificação. Verifique imediatamente os danos ao selo, vazamento de água e procedimentos de coleta. A contaminação excessiva da água tornou impossível a análise completa. Troque o óleo imediatamente."

Pelo mesmo motivo do estudo de caso anterior de tempo de processamento razoável, levando em conta que este tempo se mostrou empiricamente em

crescente exponencial em função do número de frases usadas, decidiu-se por trabalhar com um conjunto de 3.000 relatórios. Conforme a literatura, essa quantidade é significativa para a realização do estudo (KANNAN, SURTI, 2019). Além disso, para facilitar a explicação metodológica a seguir e para a validação inicial, selecionou-se um conjunto de 16 relatórios.

4.2.3 Pré-processamento de dados de texto

O pré-processamento depende consideravelmente do tipo de dado em questão, e não existe receita geral para o tratamento dos dados. Neste conjunto de dados, além de todas as etapas utilizadas para a empresa da primeira aplicação, o primeiro passo no pré-processamento dos textos foi separar cada relatório técnico em frases. Nesse caso, como observado na seção anterior, os relatórios são compostos por uma sequência de sugestões separadas por pontos finais; então, assumiu-se que uma frase era o conjunto de todas as palavras entre os pontos finais consecutivos.

Também é possível notar a partir dos exemplos na seção anterior que, ao separar os relatórios técnicos em frases independentes, as frases geradas têm uma intenção muito clara que muitas vezes não têm conexão direta com a frase anterior ou subsequente do mesmo relatório. Portanto, essa separação garante um agrupamento de melhor qualidade, uma vez que as intenções e assuntos contidos em um relatório são diversos; porém, ao avaliar as frases individualmente, percebe-se que é possível criar grupos de frases com intenções e propósitos similares, provenientes de relatórios diferentes.

Com o intuito de validar a metodologia proposta, foram selecionados *a priori*, 16 relatórios, mostrados a seguir por completo. Eles são apresentados em dupla, pois pares de relatórios com intenções similares entre si foram selecionados de modo deliberado. Cada dupla de relatórios tem uma intenção diferente. Além de facilitar a compreensão da metodologia, esse procedimento inicial também foi usado para a validação inicial.

- Nível normal de desgaste. Óleo em condição de uso. Monitorar nos intervalos regulares. Amostra anormal por faltar horas do óleo. Esta

informação interfere na avaliação. Informar os dados quanto ao tipo de óleo utilizado para melhor avaliação e acompanhamento dos resultados.

- Nível normal de desgaste. Monitorar nos intervalos regulares. Foi trocado o óleo.

- A redução da viscosidade sinaliza possível transferência de óleo que pode gerar desgaste prematuro. Verificar na inspeção diária danos nas vedações e ruídos anormais. Na próxima revisão verificar aumento no nível de óleo e presença de limalhas. Foi trocado o óleo. Coletar nova amostra regularmente.
- A redução da viscosidade sinalizam possível transferência de óleo de freio. Verificar na inspeção diária danos nas vedações e ruídos anormais. Na próxima revisão verificar aumento no nível de óleo e presença de limalhas. Trocar o óleo e coletar nova amostra regularmente.

- A XXXXXXXXXXX não recomenda o uso de óleos industriais, pois estes não foram desenvolvidos para operar nas temperaturas e pressões dos sistemas hidráulicos XXX. A amostra apresentou alta contaminação por partículas, o que pode ocasionar desgaste. Código ISO recomendado XXXXXXXXXXX 18/15. Efetuar a troca do óleo e utilizar um lubrificante conforme recomendação do manual. Trocar o óleo e coletar nova amostra regularmente. Foi informado vazamento no cilindro.
- A XXXXXXXXXX não recomenda o uso de óleos industriais, pois estes não foram desenvolvidos para operar nas temperaturas e pressões dos sistemas hidráulicos XXX. A amostra apresentou impurezas na análise visual. Esta contaminação pode ocasionar desgaste prematuro. Verificar na inspeção diária danos nas vedações e ruídos anormais. Na próxima revisão verificar o procedimento de coleta. Efetuar a troca do óleo e utilizar um lubrificante conforme recomendação do manual. Trocar o óleo e coletar nova amostra regularmente.

- A Contaminação excessiva por água pode causar degradação do óleo e falha na lubrificação evidenciado por traços de limallhas. Verificar imediatamente danos nas vedações, infiltrações de água e procedimentos de coleta. (A contaminação excessiva por água impossibilitou análise completa). Foi trocado o óleo. Coletar nova amostra com 125 horas.
- A Contaminação excessiva por água pode causar degradação do óleo e falha na lubrificação que gerar desgaste prematuro evidenciado por traços de limalhas. Verificar imediatamente danos nas vedações, infiltrações de água e procedimentos de coleta. (A contaminação excessiva por água impossibilitou análise completa). Foi trocado o óleo. Coletar nova amostra na metade do período normal de coleta.

- A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio associados aos traços de limalhas ferrosas podem indicar entrada de poeira, desgastes nas camisas, eixos comandos, engrenagem, anéis de segmento, pistões, arruelas de encostos, casquilhos e buchas da turbina. Verificar na inspeção diária ruídos anormais e condição dos filtros de ar. Na próxima revisão verificar sistemas de admissão, injeção e limalhas no filtro de óleo. Foi trocado o óleo. Coletar nova amostra regularmente.
- A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio podem indicar entrada de poeira, desgastes em componentes internos e degradação do óleo. O teor de Potássio pode indicar contaminação por líquido arrefecedor. Verificar imediatamente condição dos filtros de ar, sistema de admissão e injeção, limalhas no filtro de óleo, aumento da temperatura de operação e presença de óleo no radiador. Trocar o óleo e coletar nova amostra com 125 horas.

- A alta contaminação por água pode causar degradação do óleo e falha na lubrificaçã que pode gerar desgaste prematuro evidenciado pelos teores de Ferro e Cobre. Verificar imediatamente infiltrações de água,

danos nas vedações e procedimentos de coleta. Trocar o óleo e coletar nova amostra com 125 horas.

- A alta contaminação por água pode causar degradação do óleo e falha na lubrificaçã que pode gerar desgaste prematuro evidenciado pelos teores de ferro. Verificar imediatamente infiltrações de água, danos nas vedações e procedimentos de coleta. Trocar o óleo e coletar nova amostra com 125 horas.

- Os teores de Silício e Alumínio sinalizam entrada de poeira, que podem ocasionar desgaste nas camisas, anéis de segmento e pistões, indicado pelos níveis de Ferro e Cromo associado aos traços de limalhas ferrosas. Os teores de Sódio e Potássio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor. Verificar imediatamente ruídos anormais, condição dos filtros de ar, sistema de admissão, limalhas no filtro de óleo, sopro no cárter, presença de óleo no radiador e aumento da temperatura de operação. Pressurizar o radiador. Trocar o óleo e coletar nova amostra na metade do período normal de coleta.
- Os teores de Silício e Alumínio sinalizam entrada de poeira, que podem ocasionar desgaste nas camisas, anéis de segmento e pistões, indicado pelos níveis de Ferro e Cromo associado aos traços de limalhas ferrosas. Os teores de Sódio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor. Verificar imediatamente presença de limalhas no filtro de óleo, ruídos anormais, sistema de admissão, condição dos filtros de ar, presença de óleo no radiador e aumento da temperatura de operação. Pressurizar o radiador. Foi trocado o óleo. Coletar nova amostra na metade do período normal de coleta.

- amostra apresentou impurezas na análise visual. Esta contaminação pode ocasionar desgaste prematuro. Verificar na inspeção diária danos nas vedações e ruídos anormais. Na próxima revisão verificar o procedimento de coleta. Trocar o óleo e coletar nova amostra regularmente.

- ação do equipamento. A amostra apresentou alta contaminação por partículas, o que pode ocasionar desgaste. Os teores de Cobre podem indicar possíveis desgastes em mancais, buchas, engrenagens e discos de fricção. Verificar na inspeção diária danos nas vedações e ruídos anormais. Na próxima revisão trocar o filtro de óleo para um de alta eficiência. Código ISO recomendado XXXXXXXXXXX 18/15. Trocar o óleo e coletar nova amostra regularmente.

Com quebra desses 16 relatórios em frases, tem-se 45 frases, reproduzidas a seguir, que aparecem mais de uma vez nos relatórios.

1. Nível normal de desgaste
2. Óleo em condição de uso
3. Monitorar nos intervalos regulares
4. Amostra anormal por faltar horas do óleo
5. Esta informação interfere na avaliação
6. Informar os dados quanto ao tipo de óleo utilizado para melhor avaliação e acompanhamento dos resultados
7. Foi trocado o óleo
8. A redução da viscosidade sinaliza possível transferência de óleo que pode gerar desgaste prematuro
9. Verificar na inspeção diária danos nas vedações e ruídos anormais
10. Na próxima revisão verificar aumento no nível de óleo e presença de limalhas
11. Coletar nova amostra regularmente
12. A redução da viscosidade sinalizam possível transferência de óleo de freio
13. Trocar o óleo e coletar nova amostra regularmente.
14. A XXXXXXXXXXX não recomenda o uso de óleos industriais, pois estes não foram desenvolvidos para operar nas temperaturas e pressões dos sistemas hidráulicos XXX
15. A amostra apresentou alta contaminação por partículas, o que pode ocasionar desgaste
16. Código ISO recomendado XXXXXXXXXXX 18/15

17. Efetuar a troca do óleo e utilizar um lubrificante conforme recomendação do manual
18. Foi informado vazamento no cilindro
19. A amostra apresentou impurezas na análise visual
20. Esta contaminação pode ocasionar desgaste prematuro
21. Na próxima revisão verificar o procedimento de coleta
22. A Contaminação excessiva por água pode causar degradação do óleo e falha na lubrificação evidenciado por traços de limallhas
23. Verificar imediatamente danos nas vedações, infiltrações de água e procedimentos de coleta
24. A contaminação excessiva por água impossibilitou análise completa
25. Coletar nova amostra com 125 horas
26. Coletar nova amostra na metade do período normal de coleta
27. A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio associados aos traços de limalhas ferrosas podem indicar entrada de poeira, desgastes nas camisas, eixos comandos, engrenagem, anéis de segmento, pistões, arruelas de encostos, casquilhos e buchas da turbina
28. Verificar na inspeção diária ruídos anormais e condição dos filtros de ar
29. Na próxima revisão verificar sistemas de admissão, injeção e limalhas no filtro de óleo
30. A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio podem indicar entrada de poeira, desgastes em componentes internos e degradação do óleo
31. teor de Potássio pode indicar contaminação por líquido arrefecedor
32. Verificar imediatamente condição dos filtros de ar, sistema de admissão e injeção, limalhas no filtro de óleo, aumento da temperatura de operação e presença de óleo no radiador
33. A alta contaminação por água pode causar degradação do óleo e falha na lubrificaçã que pode gerar desgaste prematuro evidenciado pelos teores de Ferro e Cobre

34. A alta contaminação por água pode causar degradação do óleo e falha na lubrificaçã que pode gerar desgaste prematuro evidenciado pelos teores de ferro
35. Verificar imediatamente infiltrações de água, danos nas vedações e procedimentos de coleta
36. Os teores de Silício e Alumínio sinalizam entrada de poeira, que podem ocasionar desgaste nas camisas, anéis de segmento e pistões, indicado pelos níveis de Ferro e Cromo associado aos traços de limalhas ferrosas
37. Os teores de Sódio e Potássio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor
38. Os teores de Sódio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor
39. Verificar imediatamente ruídos anormais, condição dos filtros de ar, sistema de admissão, limalhas no filtro de óleo, sopro no cárter, presença de óleo no radiador e aumento da temperatura de operação
40. Verificar imediatamente presença de limalhas no filtro de óleo, ruídos anormais, sistema de admissão, condição dos filtros de ar, presença de óleo no radiador e aumento da temperatura de operação
41. Pressurizar o radiador
42. Trocar o óleo e coletar nova amostra na metade do período normal de coleta.
43. Coletar nova amostra na metade do período normal de coleta.
44. Os teores de Cobre podem indicar possíveis desgastes em mancais, buchas, engrenagens e discos de fricção
45. Na próxima revisão trocar o filtro de óleo para um de alta eficiência

Após esse pré-processamento baseado na divisão dos relatórios técnicos em frases, é realizado o mesmo procedimento anterior, de identificação e retirada de *stop words*, com a plataforma NLTK. A seguir, as palavras que foram classificadas como *stop words*.

```
stop_words = ['a', 'o', 'que', 'e', 'é', 'do', 'de', 'da', 'em', 'um',
'dum', 'para', 'com', 'pra', 'uma', 'os', 'dos, ''no', 'se', 'na',
```

```
'por', 'mais', 'as', 'dos', 'como', 'mas', 'ao', 'ele', 'das', 'à',
'seu', 'sua', 'ou', 'quando', 'muito', 'nos', 'já', 'eu', 'também',
'só', 'pelo', 'pela', 'até', 'isso', 'ela', 'dela', 'tiver',
'tivermos', 'tiverem', 'terei', 'terá', 'teremos', 'terão', 'teria',
'teríamos', 'teriam'])
```

Na sequência, é realizado o *stemming* usando o *CountVectorizer* da biblioteca *sklearn*. Da mesma forma que foi feita no estudo de caso anterior, a metodologia foi testada com e sem o uso do *stemming*, com o objetivo de comparação de resultados.

4.2.4 Modelo *N-Gram*

Nesta etapa, é aplicado o modelo *N-Gram*, e na sequência, é realizado o cálculo da métrica TF-IDF. Assim como no estudo de caso anterior, foram testados bigramas e trigramas. Em seguida, usando as mesmas bibliotecas, foi aplicado o TF-IDF para a verificação de frequência dos bigramas.

A Tabela 3 mostra um exemplo de frequência de frase que foi desconstruida. Foi usada a frase “Verificar na inspeção diária danos nas vedações e ruídos anormais”, que após a retirada das *stop words* e do *stemming*, resultou em “Verificar inspeção diária danos vedações ruídos anormais”.

Tabela 3: Frequência de bigramas por dataset segundo caso de estudo.

Bigrama	**Frequência de ocorrência**
Verificar inspeção	6
inspeção diária	6
diária danos	6
danos vedações	6
vedações ruídos	6
ruídos anormais	6

Percebe-se que a frequência dos bigramas é a mesma em todos os casos. Existem duas explicações técnicas que justificam esse resultado. A primeira delas é o conjunto de dados significativamente pequeno, com apenas 16 relatórios. Por ter um espaço amostral de 45 frases, é dificil encontrar bigramas

iguais que não na própria frase de origem. A segunda justificativa é a origem do conjunto de dados em si. Os relatórios técnicos têm alguns pontos pré-estabelecidos que devem ser respondidos. Por esse motivo, os textos do relatórios acabam tendo um espaço amostral de palavras e assuntos limitados a esses pontos pré-estabelecidos, diferente do conjunto de dados do estudo de caso anterior que, apesar de também tratar de assuntos específicos, parte da escrita livre de clientes de diferentes lugares e estilos de vida, o que torna a distribuição de palavras (consequentemente de bigramas) maior. A aplicação do *N-Gram* usa a mesma biblioteca do estudo de caso anterior.

4.2.5 Vetorização de *N-Grams*

Esta etapa também segue os mesmos passos utilizados no estudo de caso anterior, onde cada palavra recebe o seu respectivo valor proveniente da Equação 1. Esse valor então substitui cada palavra em seu bigrama. As bibliotecas usadas são as mesmas e para a equação de distância de Jacard, também não foi definido um limite mínimo, sendo o limite máximo igual 1,0. Como não existem frases idênticas, isso não alterou o resultado final.

4.2.6 Redução de dimensionalidade

Foram usados novamente os métodos SOM e PCA. Na aplicação do PCA, com 45 frases de entrada, foram selecionadas 25 componentes principais como saída, que juntas explicam 80% da variância total dos dados originais. Na aplicação do SOM, empregou-se um mapa com 35 neurônios, um valor próximo àquele dado pela Equação 8. Para o conjunto de 3.000 frases, são usados como saída, 150 componentes principais no PCA, e 500 neurônios no mapa do SOM. Foram usadas as mesmas bibliotecas do estudo de caso anterior.

4.2.7 Clusterização de *N-Grams*

Para a etapa de clusterização, os parâmetros dos algoritmos *k*-means e DBSCAN foram definidos de modo empírico. Para o caso de 45 frases, foram

usados valores de *k* iguais a 3, 5, 7 e 10. Já para o caso com 3.000 frases, no caso do *k*-means, percebeu-se um comportamento similar ao estudo de caso anterior, onde foram utilizados números de agrupamentos (*k*) iguais a 5, 7, 10, 15 e 20; pois para valores acima de 20, não foram obtidos resultados melhores. Para o DBSCAN, no caso de 45 frases, utilizou-se eps (distância máxima entre documentos) igual a 0,3, 0,5,0,7 e 1, e no caso de 3.000 frases, os valores também foram idênticos ao estudo de caso anterior, iguais a 0,3, 0,5, 0,7, 1, 5, 10, 15 e 20. Para a aplicação do *k*-means e do DBSCAN, foram empregadas as mesmas bibliotecas em Python do caso anterior.

4.2.8 Métrica de performance

Como no estudo de caso anterior, calculou-se, a partir dos resultados de clusterização, o coeficiente de *silhouette*. Também se utilizou o pacote da biblioteca *sklean* chamado *sklearn.metrics.silhouette_score.*

5. RESULTADOS E DISCUSSÃO

5.1 Setor Elétrico

A metodologia explicada anteriormente foi aplicada na sequência que foi explicada, confrome apresentado na Figura 11. O processo foi realizado com e sem a etapa de *stemming*. A idéia é verificar a hipótese de que a etapa de *stemming*, nesse estudo de caso, pode ser retirada por não alterar o resultado principalmente por não se tratar da língua inglesa (TOMLINSON, 2003).

Além disso, foram testados bigramas e trigramas, e observou-se que para os dois casos, o bigrama mostrou melhores resultados, sendo então usado como base para aplicação dos outros métodos.

A Tabela 4 e a Tabela 5 mostram os resultados utilizando o DBSCAN como algoritmo de clusterização. São apresentados tanto os valores do coeficiente de *silhouette* quanto o tempo de processamento computacional, para ambos os algoritmos de redução de dimensionalidade: PCA e SOM, e para diversos valores do parâmetro eps.

Tabela 4: Resultados aplicando-se o DBSCAN para o conjunto de 2500 mensagens.

2500 mensagens				
	PCA		**SOM**	
eps	**Com *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,97	0,40	0,94	0,24
0,5	0,97	0,40	0,96	0,23
0,7	0,97	0,39	0,97	0,27
1	0,97	0,40	0,97	0,34
5	0,97	0,44	0,97	0,30
10	0,97	0,42	0,97	0,32
15	0,97	0,39	0,97	0,45
eps	**Sem *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,97	3,35	0,97	3,27
0,5	0,97	3,35	0,97	3,32
0,7	0,97	3,36	0,97	3,33
1	0,97	3,36	0,97	3,41
5	0,97	3,44	0,97	3,49

10	0,98	3,46	0,97	3,49
15	0,98	3,22	0,97	3,26

Tabela 5: Resultados aplicando-se o DBSCAN para o conjunto de 5000 mensagens.

5000 mensagens				
	PCA		**SOM**	
eps	**Com *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,93	1,12	0,95	2,01
0,5	0,93	1,13	0,96	0,89
0,7	0,93	1,13	0,97	0,85
1	0,93	1,14	0,97	2,54
5	0,88	1,21	0,97	3,09
10	0,97	1,20	0,97	3,24
15	0,97	1,11	0,97	3,33
eps	**Sem *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,93	18,02	0,93	16,24
0,5	0,93	17,99	0,93	16,30
0,7	0,93	18,48	0,93	16,34
1	0,93	18,65	0,93	16,84
5	0,88	20,78	0,88	19,10
10	0,98	21,60	0,97	19,63
15	0,98	21,04	0,97	19,02

Se tratando do DBSCAN em si, existem alguns questionamentos que podem ser gerados a partir das análises comparativas feitas. A primeira é se existe diferença no comportamento dos resultados entre os conjuntos de 2.500 e 5.000 mensagens. A segunda é se existe diferença no processo com a retirada do procedimento de *stemming*. A terceira, qual dos dois métodos de redução de dimensionalidade gerou melhor resultado, e por fim, se existe algum valor de eps que se destacou em relação à faixa de variação.

A primeira conclusão a ser tomada é que não existe diferença entre os conjuntos de 2.500 e 5.000 mensagens quando analisados os valores do coeficiente de *silhouette*. Isso leva a afirmar que 2.500 mensagens já são suficientes para se ter uma análise em grande escala. Esse fato é importante por possibilitar um maior número de simulações, uma vez que o tempo de gasto computacional é menor. Com relação ao tempo de processamento

computacional, no caso do uso de PCA, observa-se que o conjunto com 5.000 mensagens consumiu, em média, o dobro do tempo para o processamento com *stemming*, e seis vezes o valor para o processamento sem *stemming*, em relação ao conjunto de 2.500 mensagens. Além disso, percebe-se que para um mesmo conjunto de mensagens, o consumo de tempo é consideravelmente maior quando não se usa *stemming*, ou seja, apesar da etapa de *stemming* não influenciar no valor do coeficiente de *silhouette*, ele influencia diretamente no tempo de processamento computacional. Isso ocorre porque, como o próprio método se propõe, o *stemming* simplifica significativamente o conjunto de dados, simplificando também o seu tempo de análise (TARASIEV, 2019). Essas observações respondem aos dois primeiros questionamentos.

Para a comparação dos métodos de redução de dimensionalidade: PCA e SOM, e dos valores de eps, faz-se uso da Figura 13. A partir dos dados com *stemming*, relativo ao conjunto de 5.000 frases, é possível perceber que o valor do coeficiente de *silhouette* se mantém constante. Para alguns valores de eps, há valores de *silhouette* relativamente menores; porém, não de modo significativo (4,63% de diferença em média). A mesma afirmação pode ser feita quando comparados os métodos de redução de dimensionalidade; não há variação suficientemente grande que justifique a afirmação de que um dos dois métodos é superior (3,06% de diferença em média).

Portanto, é possível concluir que a quantidade de dados definida para os conjuntos de mensagens foi satisfatória, que a presença de *stemming* é significativa apenas para o tempo de processamento computacional, e que não há diferença significativa entre os valores de eps escolhidos e nem entre os métodos de redução de dimensionalidade. De modo mais exato, observa-se valores maiores e estáveis para o coeficiente de *silhouette*, independentemente da técnica de redução de dimensionalidade, para valores de eps iguais a 10 e 15.

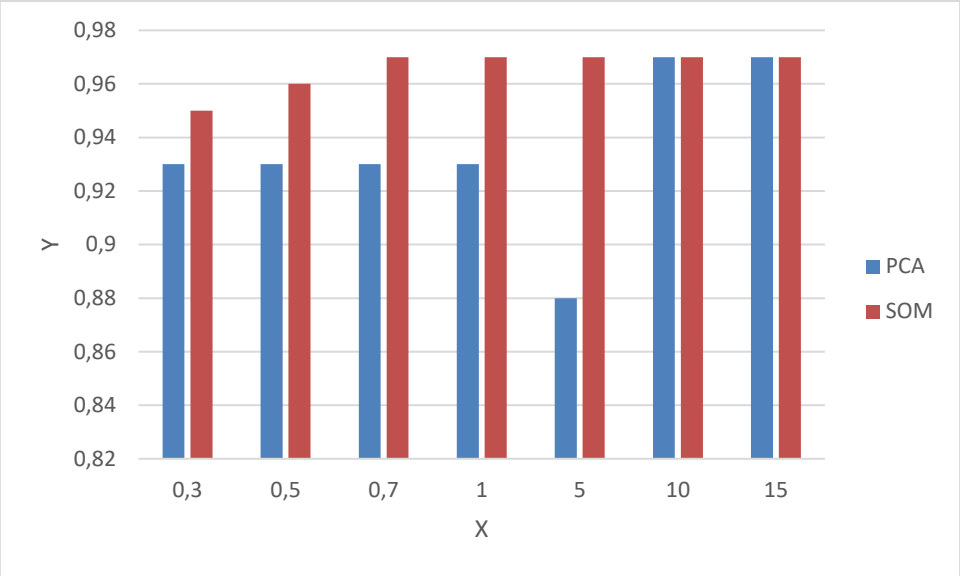


Figura 13: Resultados para diferentes valores de eps, com PCA e SOM, usando o DBSCAN.

A Tabela 6 e a Tabela 7 repetem a análise anterior; porém, ao se empregar o algoritmo Mini-Batch como método de clusterização. Os mesmos questionamentos podem ser feitos, se há diferença no comportamento dos resultados entre os conjunto de 2.500 e 5.000 mensagens, se há diferença no processo com e sem *stemming*; qual dos métodos de redução de dimensionalidade gerou o melhor resultado; e se há existe algum valor de *k* (número de *clusters*) que se destacou em relação aos demais.

Tabela 6: Resultados aplicando-se o Mini-Batch para o conjunto de 2500 mensagens.

2500 mensagens				
	PCA		**SOM**	
k	**Com stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	0	0,27	0,01	0,24
7	-0,05	0,26	0,02	0,23
10	-0,09	0,31	-0,10	0,27
15	0,02	0,28	-0,05	0,34
20	0,05	0,43	0,02	0,30
k	**Sem stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	-0,07	0,88	0,02	0,86
7	-0,10	0,85	-0,01	0,87
10	-0,12	0,89	-0,08	0,89

15	-0,18	0,89	-0,17	0,88
20	-0,10	1,06	-0,15	1,17

Tabela 7: Resultados aplicando-se o Mini-Batch para o conjunto de 5000 mensagens.

5000 mensagens				
	PCA		**SOM**	
k	**Com stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	-0,05	0,72	-0,09	0,18
7	0,06	0,82	-0,11	0,18
10	-0,22	0,72	0,15	0,18
15	0,06	0,72	-0,08	0,22
20	0,02	0,90	0,04	0,23
k	**Sem stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	-0,12	5,21	-0,15	0,92
7	-0,16	5,57	-0,20	0,97
10	-0,21	5,56	-0,19	1,01
15	-0,15	5,27	-0,23	1,12
20	-0,19	6,00	-0,27	1,29

O primeiro ponto que se pode afirmar é que o comportamento dos dois conjuntos de dados, em relação ao coeficiente de *silhouette*, são similares, independentemente do número de mensagens. O segundo ponto, diferentemente do DBSCAN, é que o uso de *stemming* mostrou-se consideravelmente negativo, o que pode ser explicado pelo próprio comportamento do método Mini-Batch, que trabalha com a aplicação do *k*-means em pequenos lotes em sequência. A explicação mais plausível para justificar esses resultados é que, com a redução do número de palavras, elas se tornam consideravelmente uniformes, não contendo um mínimo de variabilidade para que o método consiga discriminar os agrupamentos. Isso faz com que diversos agrupamentos sejam criados, mesmo contendo o mesmo tipo de padrão que poderiam colocar estes dados de diferentes agrupamentos, juntos em um só (SANTURKAR, 2018). Percebe-se também, como para o DBSCAN, que o uso de *stemming* reduz significativamente o tempo computacional.

A Figura 14, relativa ao uso de *stemming* e ao conjunto de 5.000 frases, mostra, como para o DBSCAN, que o método de redução de dimensionalidade não gera diferenças significativas entre os valores de *silhouette* (23,07% de diferença em média), assim como o número *k* de agrupamentos também não influencia muito os valores de *silhouette* (50,00% de diferença em média). Além disso, não se observa uma tendência de crescimento monotônico para os valores de *silhouette*, com o aumento de eps, independentemente da técnica de redução de dimensionalidade.

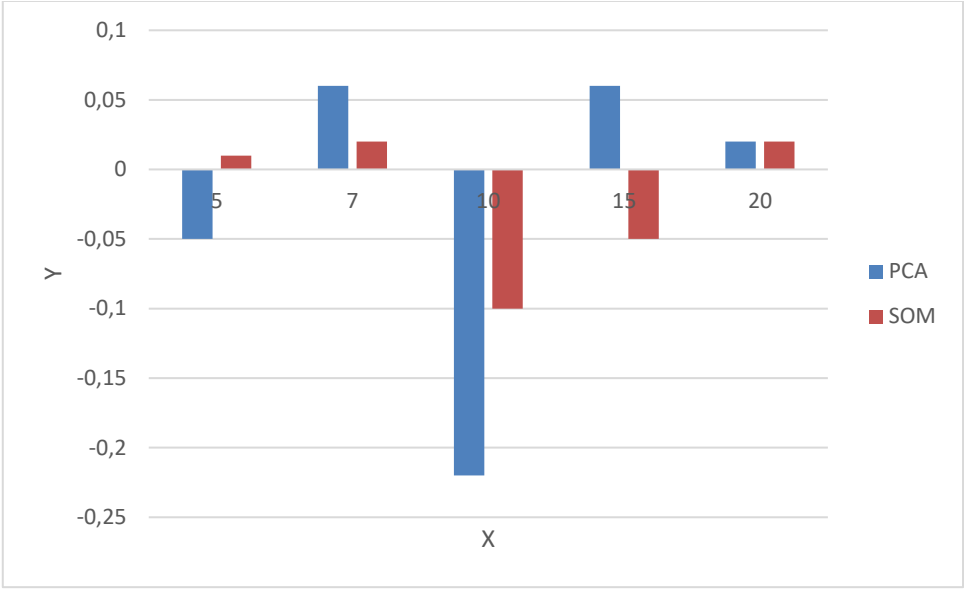


Figura 14:Resultados para diferentes *k*, com PCA e SOM, usando Mini-Batch.

A Tabela 8 e a Tabela 9 apresentam os resultados utilizando o *k*-means como algoritmo de clusterização.

Tabela 8:Resultados aplicando-se o K-Means para o conjunto de 2500 mensagens.

2500 mensagens				
	PCA		**SOM**	
k	**Com stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	0,94	0,76	0,97	0,19
7	0,96	0,80	0,97	0,19
10	0,96	0,87	0,96	0,20
15	0,97	0,95	0,93	0,26

20	0,97	1,05	0,93	0,42
k	**Sem stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	0,95	2,44	0,94	2,35
7	0,96	2,64	0,96	2,89
10	0,97	3,14	0,96	3,10
15	0,97	3,86	0,97	3,72
20	0,97	4,52	0,98	4,39

Tabela 9: Resultados aplicando-se o K-Means para o conjunto de 5000 mensagens.

5000 mensagens				
	PCA		**SOM**	
k	**Com stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	0,93	0,40	0,96	1,28
7	0,93	0,40	0,96	0,59
10	0,93	0,39	0,97	0,61
15	0,94	0,40	0,92	0,76
20	0,95	0,44	0,92	0,82
k	**Sem stemming**			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
5	0,95	3,35	0,93	11,90
7	0,96	3,35	0,93	12,92
10	0,97	3,36	0,94	15,13
15	0,97	3,36	0,93	18,06
20	0,97	3,44	0,94	21,28

Percebe-se que existe um comportamento bastante homogêneo com relação aos resultados de *silhouette*, independentemente do tamanho do conjunto de mensagens. O mesmo pode ser afirmado com relação ao *stemming*. Também se observa que o uso de *stemming* reduz, cerca de três vezes, o tempo de processamento computacional.

A Figura 15, relativa ao uso de *stemming* e ao conjunto de 5000 frases, compara os valores de *silhouette* entre SOM e PCA para os diferentes valores de *k*. Assim como em todos os casos anteriores, o método de redução de dimensionalidade não gera diferença significativa (2,10% de diferença em média), assim como o valor de *k* (1,06% de diferença em média).

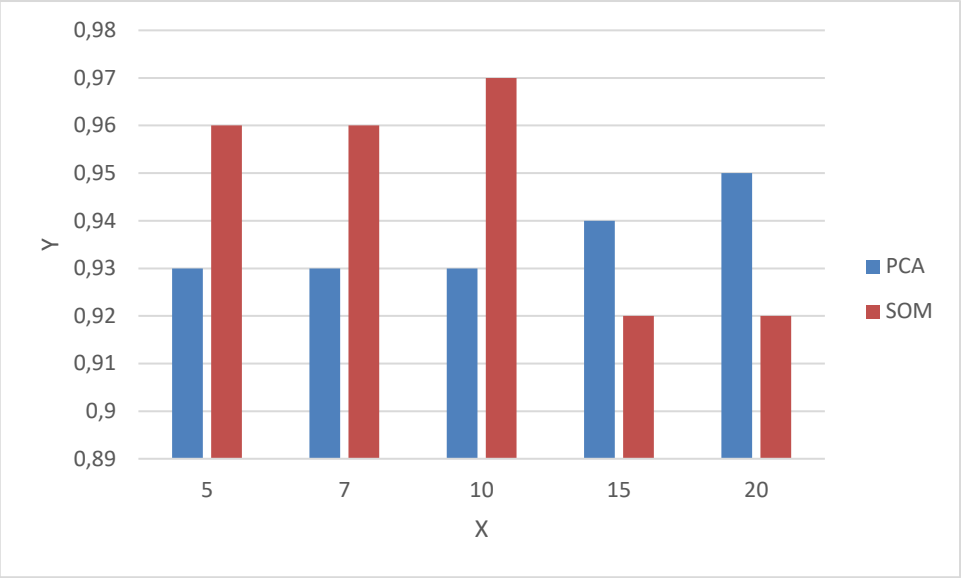


Figura 15: Resultados para diferentes *k*, com PCA e SOM, usando *k*-means.

Mesmo com valores significativamente altos para o coeficiente de *silhouette*, obtidos em geral nas aplicações anteriores, é fundamental verificar, com o auxílio de analistas experientes, se os agrupamentos encontrados fazem sentido na prática. Nessa direção, apresenta-se a Tabela 10*,* com algumas frases de cada um dos *clusters*. Esses resultados são para o conjunto com 5.000 frases, dado o uso de *stemming*, do PCA como técnica de redução de dimensionalidade, e do *k*-means como algoritmo de clusterização, com k = 7. A razão da seleção dessa combinação de parâmetros foi a menor variabilidade entre os valores de *silhouette*, e ainda que, com sete *clusters*, apenas um deles é classificado como “aleatório”, isto é, quando não é possível definir uma *intent* de modo claro. Todos os casos com *k* acima de 7 encontraram oito *intents*, sendo “Identificação” a oitava *intent*, estando as frases desta *intent* alocadas nos outros *clusters k* > 7.

Seguem outras observações sobre os resultados obtidos anteriormente. Para k = 10 ou mais, o número de agrupamentos aleatórios aumenta. No caso de k = 10, esse número variou de 2 a 3, dependendo do uso ou não de *stemming*. No caso de existir mais de um agrupamento aleatório, a somatória das frases existentes nos agrupamentos é computada de modo único na presente análise. Para k = 5, *intents* similares tendem a formar um só agrupamento; no caso, “Informar pagamento” e “Informar leitura” formam juntos um *cluster* e “Falta de luz” juntamente com “Reclamação” formam outro *cluster*,

sendo que estes acabam recebendo a maioria das sentenças que são classificadas como “aleatório” em análises com maiores valores de k.

Pela Tabela 10, pode-se observar, em relação aos *clusters* encontrados pela metodologia proposta neste trabalho, que fora o *cluster* denominado aleatório, foi possível identificar *intents* bastante perceptíveis, conforme o objetivo geral do trabalho. As sete *intents* encontradas são conforme a seguir: “Falta de luz”, “Saudações”, “Informar pagamento”, “Reclamação”, “Segunda via”, “Informar leitura”, e o agrupamento que contém todas as frases que não foram classificadas em nenhum dos agrupamentos anteriores, que foi denominada de agrupamento “aleatório”. Após encontrar estas *intents* citadas, elas passaram por processo de validação com profissionais da área. A classificação e nomeação destes agrupamentos também foi realizada por meio de análise humana, com auxílio de especialistas. Apresenta-se também o percentual de frases em cada agrupamento. Pode-se observar que o percentual referente ao agrupamento aleatório é sigficativamente baixo, igual a 6,24%. O resultado é análogo para o conjunto com 2.500 frases.

Tabela 10: Agrupamentos de *intents* encontrados

Cluster	**Exemplo de Frases**	***Intent***	**Quantidade (%)**
1	Acabou a luz Tá sem energia aqui em casa	Falta de luz	17,26
2	Bom dia Boa noite	Saudações	26,24
3	Já paguei a conta Paguei hoje	Informar pagamento	3,82
4	Cadê a assistência? Não tem ninguém pra me atender.	Reclamação	18,29
5	Queria a segunda via Manda a conta de novo	Segunda via	5,78
6	Leitura XXXXXXX A leitura é XXXXXX	Informar leitura	22,37
7	Olá meu CPF é XXXXX Quero religar a luz	Aleatório	6,24
Total			**100,0**

A Figura 16 e a Figura 17 mostram os desempenhos dos algoritmos de clusterização: *k*-means e o DBSCAN, em relação ao percentual de frases no agrupamento aleatório. Como o algoritmo Mini-Batch apresentou resultados instatisfatórios, não foi considerado nessa análise. Percebe-se um comportamento similar entre ambos, independentemente do tamanho dos conjuntos de frases. Os melhores desempenhos são com k = 7 e eps = 0,7, respectivamente.

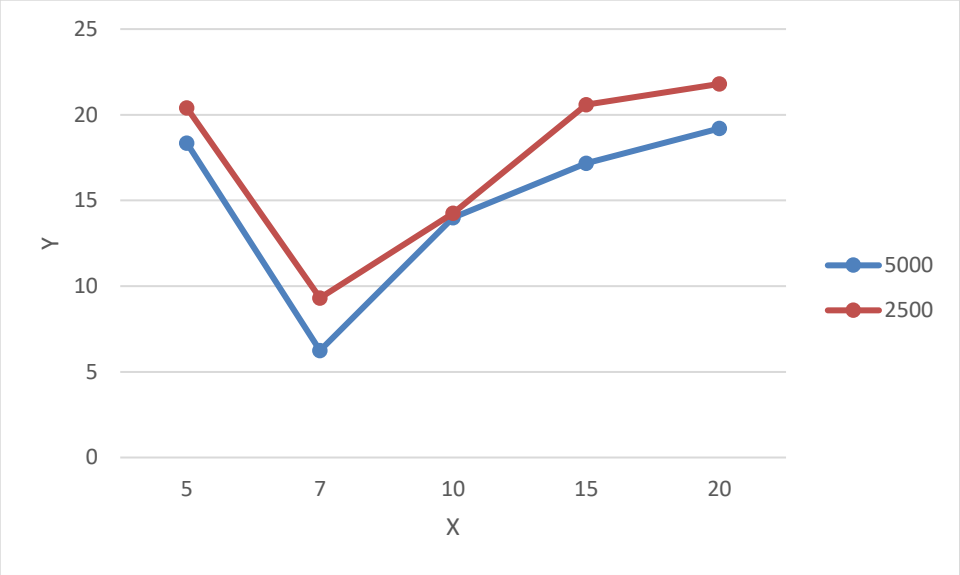


Figura 16: Percentual de frases no agrupamento "aleatório" usando *k*-means.

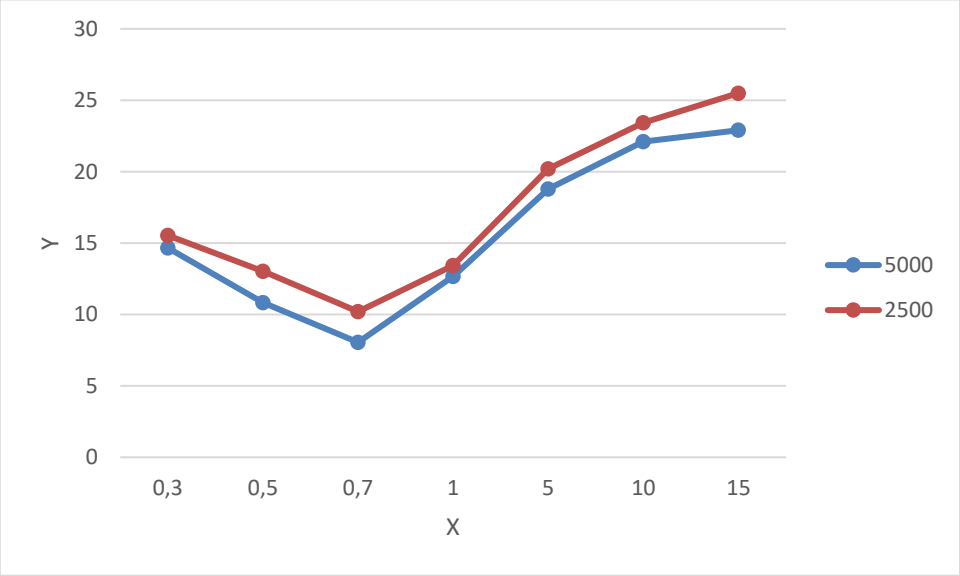


Figura 17: Percentual de frases no agrupamento "aleatório" usando DBSCAN.

Concluindo essa seção, é possível perceber, através dos resultados obtidos, o alcance dos objetivos específicos, portanto, do objetivo geral proposto neste trabalho. A metodologia proposta mostrou-se eficaz no processamento das mensagens de texto. Com isso, a partir de mensagens diferentes e aleatórias, é possível encontrar padrões, portanto, *intents*, em relação às demandas dos clientes da CEMIG. Desse modo, não é necessário orientar o cliente a fazer uso de palavras-chave para se comunicar com a empresa. Por fim, conforme colocado anteriormente, apesar de não fazer parte do escopo do presente trabalho, essa informação pode ser usada na construção de um *chatbot*.

5.2 Setor Químico

A metodologia segue o esquema da Figura 11. Trabalhou-se com conjuntos com 45 e 3.000 frases. Para o conjunto de 45 frases, foram escolhidos k = 3, 5 e 10 para o *k*-means, e eps = 0,3, 0,5, e 0,7 e 1 para o DBSCAN. Para o conjunto de 3.000 frases, as faixas de variação são as mesmas do estudo de caso do setor elétrico: k = 5, 7, 10, 15 e 20 e eps = 0.3, 0.5, 0.7, 1, 5, 10 e 15. O procedimento também foi realizado com e sem a etapa de *stemming*, de modo a verificar se há alteração no resultado final.

De modo análogo à primeira aplicação, a Tabela 11 (45 frases) e a Tabela 12Tabela 4: Resultados aplicando-se o DBSCANTabela 5 (3.000 frases) mostram os resultados usando o DBSCAN como algoritmo de clusterização, dadas ambas as técnicas de redução de dimensionalidade: PCA e SOM. Para verificação de desempenho, são apresentados os valores do coeficiente de *silhouette* e do tempo de processamento computacional.

Tabela 11: Resultados para o conjunto com 45 frases com DBSCAN.

45 mensagens				
	PCA		**SOM**	
eps	**Com *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,72	0,30	0,74	0,80
0,5	0,72	0,33	0,74	0,87
0,7	0,63	0,39	0,74	1,12
1	0,61	0,40	0,62	1,36

eps	Sem *stemming*			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,71	4,35	0,73	7,23
0,5	0,71	4,68	0,73	7,45
0,7	0,64	4,76	0,74	7,55
1	0,64	5,36	0,64	7,50

Tabela 12: Resultados para o conjunto com 3.000 frases com DBSCAN.

3.000 mensagens				
	PCA		**SOM**	
eps	**Com *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,88	2,22	0,86	3,01
0,5	0,88	2,25	0,86	3,89
0,7	0,88	2,30	0,86	3,85
1	0,88	2,28	0,88	3,54
5	0,90	2,31	0,88	3,09
10	0,93	2,45	0,90	4,24
15	0,93	2,53	0,90	4,33
eps	**Sem *stemming***			
	Silhouette	**Tempo (s)**	*Silhouette*	**Tempo (s)**
0,3	0,88	30,02	0,84	33,22
0,5	0,89	30,12	0,86	33,20
0,7	0,89	32,48	0,86	33,39
1	0,89	32,65	0,86	34,84
5	0,89	33,78	0,86	34,12
10	0,89	33,60	0,86	34,63
15	0,89	34,04	0,87	35,02

Os questionamentos levantados no estudo de caso anterior valem como base para a discussão corrente.

A primeira observação é que há diferença significativa entre os conjuntos de 45 e 3.000 frases em relação aos valores do coeficiente de *silhouette* (20,4% de diferença em média) e ao tempo de processamento computacional, sendo cerca de quatro vezes maior no caso de 3.000 frases.

Outra observação é que, assim como no estudo de caso anterior, o uso de *stemming* se mostrou com baixo impacto nos valores de *silhouette* (2,27% de

diferença na média), sendo significativo apenas em relação ao tempo de processamento computacional.

A Figura 18 apresenta um comparativo entre os métodos de redução de dimensionalidade ao longo da faixa de variação do eps, dado o uso do DBSCAN. O conjunto de dados é aquele com 3.000 frases, com uso de *stemming*. O PCA se mostrou um pouco melhor ao se comparar os valores de *silhouette* (5,26% em média). Com relação ao eps, a diferença entre o menor e o maior é significativa para os valores de *silhouette* (16,3% em média). Em relação ao eps, há diferenças significativas entre os valores utilizados, sendo que aqueles de valores maiores e médios geram os melhores resultados, indicando que não houve mesclas entre os agrupamentos, pois os valores de eps maiores indicam *clusters* muito bem separados uns dos outros. (SIAU, WANG, 2018).

É possível observar que a quantidade de 3.000 frases, ao contrário de apenas 45 frases, é satisfatória para esta etapa de análise de resultados.

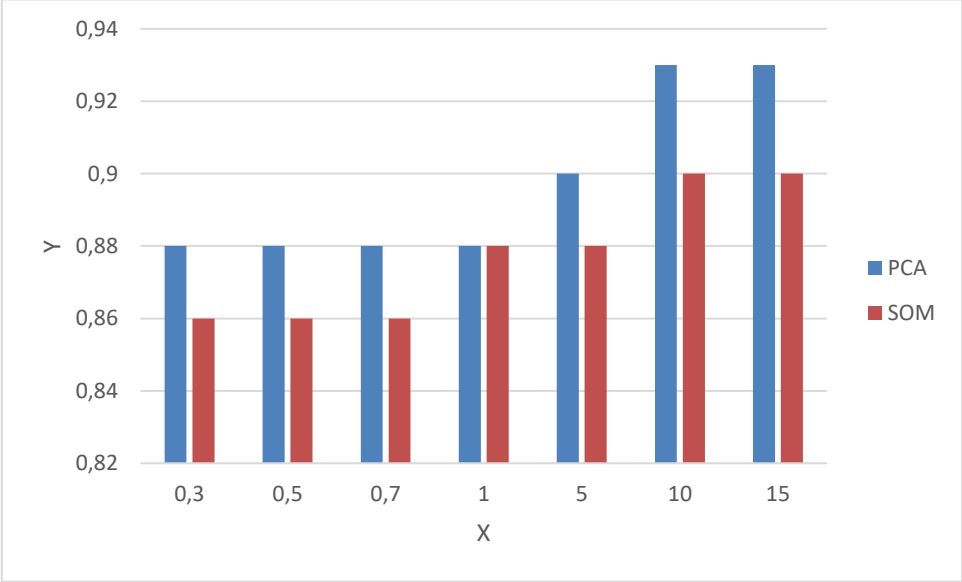


Figura 18: Comparação entre PCA e SOM usando o DSBCAN.

A Tabela 13 e a Tabela 14Tabela 7 são relativas ao uso do Mini-Batch como método de clusterização.

Tabela 13: Resultados para o conjunto com 45 mensagens com Mini-Batch.

45 mensagens

	PCA		**SOM**	
k	**Com *stemming***			
	Silhouette	**Tempo (s)**	***Silhouette***	**Tempo (s)**
5	-0,45	0,02	-0,46	0,02
7	-0,45	0,02	-0,46	0,02
10	-0,51	0,03	-0,46	0,02
k	**Sem *stemming***			
	Silhouette	**Tempo (s)**	***Silhouette***	**Tempo (s)**
5	-0,33	0,10	-0,34	0,08
7	-0,33	0,11	-0,34	0,10
10	-0,36	0,11	-0,34	0,10

Tabela 14:Resultados para o conjunto com 3.000 mensagens com Mini-Batch.

3.000 mensagens				
	PCA		**SOM**	
k	**Com *stemming***			
	Silhouette	**Tempo (s)**	***Silhouette***	**Tempo (s)**
5	-0,12	1,12	-0,14	1,32
7	-0,12	1,13	-0,14	1,34
10	-0,14	1,10	-0,18	1,45
15	-0,14	1,02	-0,18	1,39
20	-0,12	1,08	-0,18	1,40
k	**Sem *stemming***			
	Silhouette	**Tempo (s)**	***Silhouette***	**Tempo (s)**
5	0,01	2,32	0,05	3,92
7	0,01	2,86	0,05	4,12
10	0,03	2,22	0,06	4,32
15	0,03	2,76	0,06	4,74
20	0,03	2,80	0,05	4,30

Os comportamentos de ambos os conjuntos de dados são similares. Ambos obtiveram resultados insatisfatórios para o coeficiente de *silhouette*, independente do uso ou não de *stemming*. Isso pode ser justificado pelo algoritmo Mini-Batch trabalhar com a aplicação do *k*-means em pequenos lotes em sequência. As frases desse conjunto de dados são pré-definidas por seguirem as premissas contidas num roteiro que devem ser seguidas pelo menos de forma geral, pelos analistas que escrevem os relatórios técnicos, gerando padrões de resposta pré-definidos. A aplicação do Mini-Batch faz com

que agrupamentos similares sejam criados de modo separado, e padrões diferentes sejam misturados. Percebe-se também que o uso de *stemming* reduz o tempo de processamento, assim como no para o DBSCAN.

A Figura 19 diz respeito ao conjunto de 3.000 frases, com o uso de *stemming*. Apesar das diferenças entre os métodos de redução de dimensionalidade (17,6% de diferença em média), e os valores de *k* (13,3% de diferença em média), esses resultados são insatisfatórios.

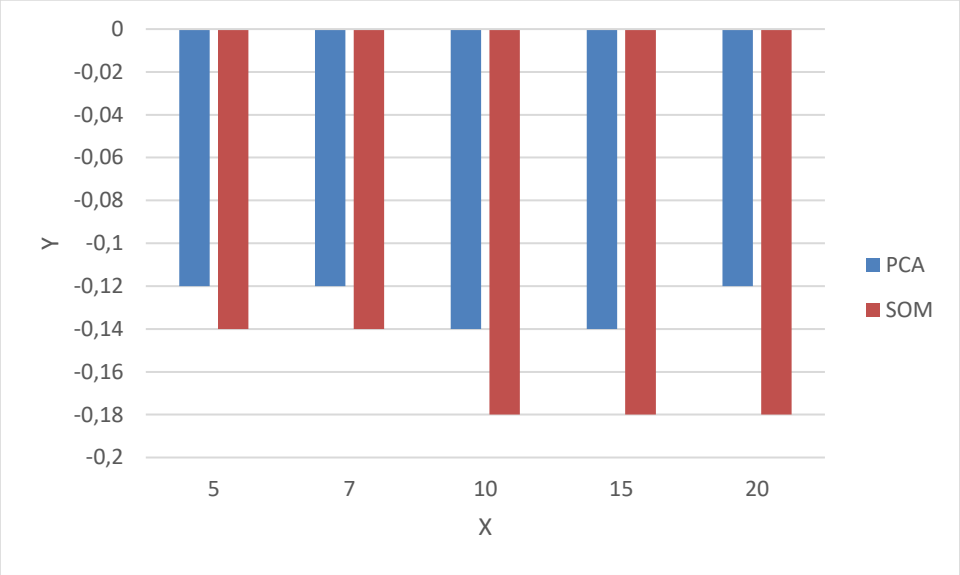


Figura 19: Comparação entre PCA e SOM com o Mini-Batch.

A Tabela 15 e a Tabela 16 apresentam os resultados utilizando o *k*-means como o método de clusterização.

Tabela 15: Resultados para o conjunto com 45 mensagens com o *k*-means.

45 mensagens				
	PCA		**SOM**	
k	**Com *stemming***			
	Silhueta	**Tempo (s)**	**Silhueta**	**Tempo (s)**
5	0,69	0,06	0,67	0,09
7	0,71	0,08	0,67	0,09
10	0,68	0,07	0,67	0,10
k	**Sem *stemming***			
	Silhueta	**Tempo (s)**	**Silhueta**	**Tempo (s)**
5	0,66	0,14	0,65	0,15
7	0,65	0,14	0,65	0,19

10	0,65	0,14	0,65	0,10

Tabela 16: Resultados para o conjunto com 3.000 mensagens com o *k*-means..

3.000 mensagens				
	PCA		**SOM**	
k	**Com *stemming***			
	Silhueta	**Tempo (s)**	**Silhueta**	**Tempo (s)**
5	0,99	0,40	0,97	1,28
7	0,99	0,40	0,96	0,59
10	0,98	0,39	0,96	0,61
15	0,98	0,40	0,96	0,76
20	0,99	0,44	0,96	0,82
k	**Sem *stemming***			
	Silhueta	**Tempo (s)**	**Silhueta**	**Tempo (s)**
5	0,97	3,35	0,96	11,90
7	0,97	3,35	0,96	12,92
10	0,96	3,36	0,96	15,13
15	0,97	3,36	0,95	18,06
20	0,97	3,44	0,96	21,28

Percebe-se que existe um comportamento bastante homogêneo com relação aos resultados de *silhouette*, independentemente do método de redução de dimensionalidade. O mesmo pode ser afirmado com relação ao *stemming*. O uso do *stemming* reduz, cerca de três vezes, o tempo de processamento computacional. Conforme esperado, o desepenho do conjunto de 3.000 frases é aquele do conjunot de 45 frases.

A Figura 15 mostra a relação dos valores de *silhouette* entre SOM e PCA para os diferentes valores de *k*, dado o conjunto de 3.000 frases e o uso de *stemming*. É possível concluir que a diferença entre os métodos não é significativa (2,10% de diferença em média), assim como em relação ao número de agrupamentos *k* (1,06% de diferença em média).

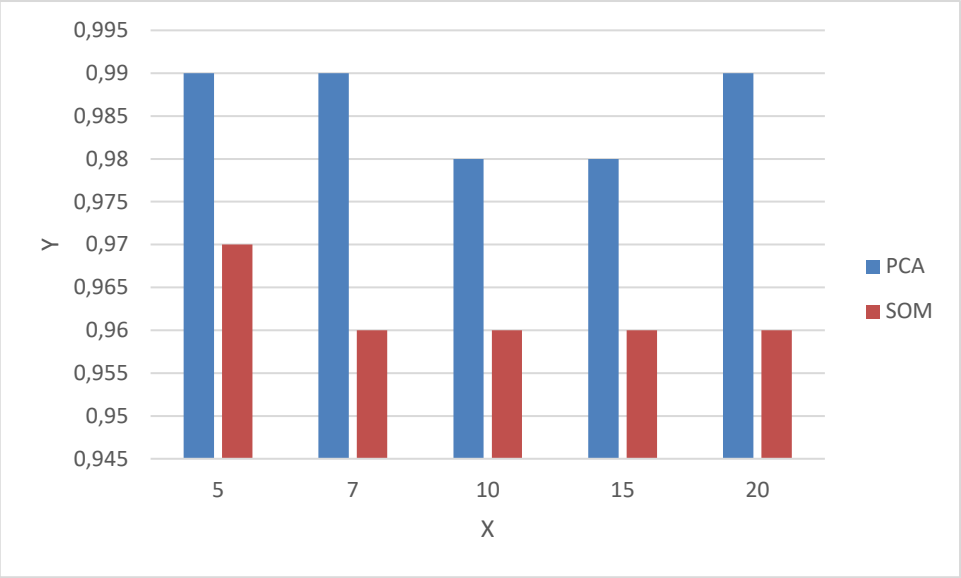


Figura 20: Comparação entre PCA e SOM para uso do *k*-means.

A Tabela 17 apresenta, de modo completo, os *clusters* formados para o conjunto de 45 frases, dado a combinação PCA/*k*-means com k = 7, que apresentou o melhor desempenho. As *intents* foram nomeadas e validadas com o auxílio de especialistas da área. A Tabela 18 contém o percentual de frases alocadas em cada agrupamento.

Tabela 17: Resultado da clusterização para o conjunto com 45 frases.

Frases	***Intent***
11. Coletar nova amostra regularmente. 13. Trocar o óleo e coletar nova amostra regularmente. 25. Coletar nova amostra com 125 horas 26. Coletar nova amostra na metade do período normal de coleta 43. Coletar nova amostra na metade do período normal de coleta.	Coletar amostra
27. A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio associados aos traços de limalhas ferrosas podem indicar entrada de poeira, desgastes nas camisas, eixos comandos, engrenagem, anéis de segmento, pistões, arruelas de encostos, casquilhos e buchas da turbina 30. A Fuligem e os teores de Ferro, Cobre, Silício e Alumínio podem indicar entrada de poeira, desgastes em componentes internos e degradação do óleo 33. A alta contaminação por água pode causar degradação do óleo e falha na lubrificação que pode gerar desgaste prematuro evidenciado pelos teores de Ferro e Cobre 34. A alta contaminação por água pode causar degradação	Teores de ferro e cobre

do óleo e falha na lubrificaçã que pode gerar desgaste prematuro evidenciado pelos teores de ferro 36. Os teores de Silício e Alumínio sinalizam entrada de poeira, que podem ocasionar desgaste nas camisas, anéis de segmento e pistões, indicado pelos níveis de Ferro e Cromo associado aos traços de limalhas ferrosas 44. Os teores de Cobre podem indicar possíveis desgastes em mancais, buchas, engrenagens e discos de fricção	
2. Óleo em condição de uso 4. Amostra anormal por faltar horas do óleo 6. Informar os dados quanto ao tipo de óleo utilizado para melhor avaliação e acompanhamento dos resultados 7. Foi trocado o óleo 8. A redução da viscosidade sinaliza possível transferência de óleo que pode gerar desgaste prematuro 8. A redução da viscosidade sinaliza possível transferência de óleo que pode gerar desgaste prematuro 12. A redução da viscosidade sinalizam possível transferência de óleo de freio 14. A XXXXXXXXXXX não recomenda o uso de óleos industriais, pois estes não foram desenvolvidos para operar nas temperaturas e pressões dos sistemas hidráulicos XXX 17. Efetuar a troca do óleo e utilizar um lubrificante conforme recomendação do manual 22. A Contaminação excessiva por água pode causar degradação do óleo e falha na lubrificação evidenciado por traços de limalhas 42. Trocar o óleo e coletar nova amostra na metade do período normal de coleta. 45. Na próxima revisão trocar o filtro de óleo para um de alta eficiência	Troca de óleo
9. Verificar na inspeção diária danos nas vedações e ruídos anormais 15. A amostra apresentou alta contaminação por partículas, o que pode ocasionar desgaste 21. Na próxima revisão verificar o procedimento de coleta 28. Verificar na inspeção diária ruídos anormais e condição dos filtros de ar 29. Na próxima revisão verificar sistemas de admissão, injeção e limalhas no filtro de óleo	Verificar próxima inspeção/revisão
23. Verificar imediatamente danos nas vedações, infiltrações de água e procedimentos de coleta 32. Verificar imediatamente condição dos filtros de ar, sistema de admissão e injeção, limalhas no filtro de óleo, aumento da temperatura de operação e presença de óleo no radiador 35. Verificar imediatamente infiltrações de água, danos nas vedações e procedimentos de coleta 39. Verificar imediatamente ruídos anormais, condição dos	Verificar danos

filtros de ar, sistema de admissão, limalhas no filtro de óleo, sopro no cárter, presença de óleo no radiador e aumento da temperatura de operação 40. Verificar imediatamente presença de limalhas no filtro de óleo, ruídos anormais, sistema de admissão, condição dos filtros de ar, presença de óleo no radiador e aumento da temperatura de operação	
31. teor de Potássio pode indicar contaminação por líquido arrefecedor 37. Os teores de Sódio e Potássio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor 38. Os teores de Sódio podem indicar contaminação por líquido de arrefecimento e/ou reação química com o trocador de calor	Teor sódio potássio
1. Nível normal de desgaste 3. Monitorar nos intervalos regulares 5. Esta informação interfere na avaliação 16. Código ISO recomendado XXXXXXXXXX 18/15 18. Foi informado vazamento no cilindro 19. A amostra apresentou impurezas na análise visual 20. Esta contaminação pode ocasionar desgaste prematuro 24. A contaminação excessiva por água impossibilitou análise completa 41. Pressurizar o radiador	Aleatório

Tabela 18: Percentual dos agrupamentos para conjunto com 45 frases.

Cluster	***Intent***	**Percentual (%)**
1	Coletar amostra	11,11
2	Teores de ferro e cobre	13,33
3	Troca de óleo	26,67
4	Verificar próxima inspeção/revisão	11,11
5	Verificar danos	11,11
6	Teor sódio potássio	6,67
7	Aleatório	20,00
Total		**100,0**

Para a mesma configuração anterior de clusterização, a Figura 21 apresenta o percentual de frases alocadas no agrupamento denominado Aleatório para a faixa de valores de *k*, para o conjunto com 3.000 frases. Pode-se observar o melhor desempenho para *k* = 7. Conforme escrito anteriormente, a configuração PCA/*k*-means mostrou o melhor desempenho para o coeficiente

de *silhouette*, e o tamanho de agrupamento aleatório. Mesmo para valores mais altos de *k*, foram encontrados apenas 7 agrupamentos com *intents* relevantes. Os demais agrupamentos contêm, de modo espalhado, frases aleatórias, ou são agrupamentos vazios.

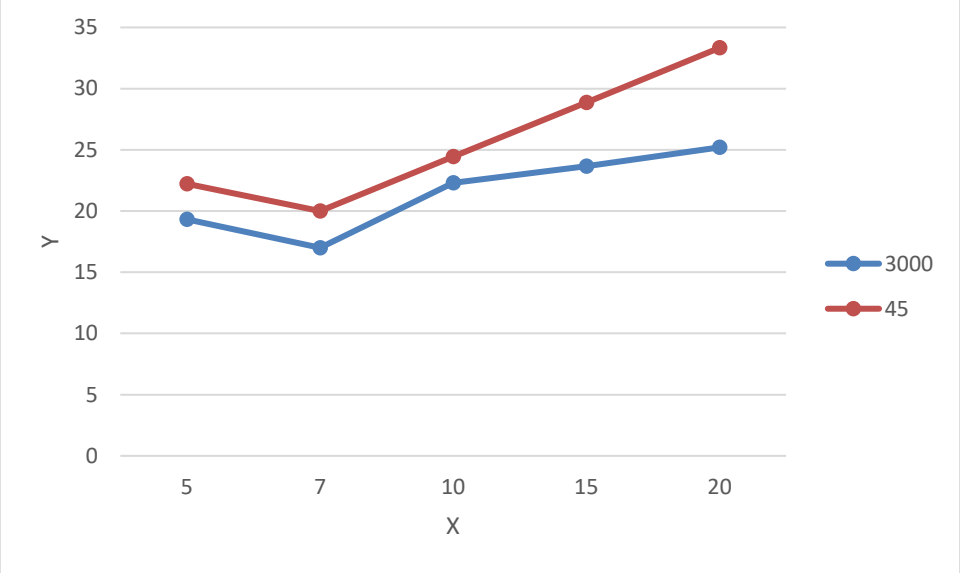


Figura 21: : Percentual de frases no agrupamento "aleatório" usando *k*-means.

6. CONCLUSÕES

O presente trabalho propõe uma metodologia para a identificação de padrões textuais através de uma abordagem não supervisionada. Testou-se, de modo satisfatório, essa porposta em duas aplicações reais. A primeira diz respeito a uma concessionária do setor de energia elétrica no Brasil, com o objetivo de agrupar as mesagens de texto de seus clientes, ou, em outras palavras, de descoberta de *intents* de seus usuários. Esse resultado apresenta vários ganhos, como por exemplo, a possibilidade de construção de um *chatbot* (a construção desse robô está além do escopo deste trabalho). A segunda refere-se a uma empresa de construção de máquinas para a construção civil, também no Brasil, com o objetivo de agrupar pareceres técnicos, em formato texto, de análises de laboratório de um óleo lubrificante utilizado nas máquinas. A alocação das frases nesses pareceres pode ser utilizada com o *input* para um processo de padronização das análised de laboratório (essa padronização está além do escopo deste trabalho). Essas análises são escritas por diferentes analistas; poe isso, a necessidade de uma padronização dessa informação.

A partir dos resultados, pode-se observar para o primeiro caso que foi possível identificar intenções bem definidas dos clientes, a partir de um conjunto de mensagens relativamente grande. Da mesma forma, para o segundo caso, foi possível identificar padrões de análises bem definidos nos relatórios técnicos.

A metodologia pode ser empregada em outros contextos. O que pode ser necessário são adaptações na etapa de pré-processamento das mensagens de texto, conforme ocorreu neste trabalho para ambas as aplicações. Essas adaptações são função da condição do banco de mensagens à disposição.

A combinação, PCA como método de redução de dimensionalidade e do *k*-means como algoritmo de clusterização, mostrou-se, em geral, a de melhor desempenho, segundo a métrica usual de avaliação denominada coeficiente de *silhouette*, o tamanho do agrupamento de dados denominado “aleatório”, que reune frases não devidamente alocadas, e o tempo de processamento computacional. Por fim, a etapa de *stemming* mostrou-se válida apenas para a redução do tempo de processamento computacional.

6.1. Sugestões de trabalhos futuros

Ambos os trabalhos apresentados podem ser vistos como parte de um trabalho maior. A continuidade destes pode ser dada por:

- A partir da base de dados criada com os grupos gerados na análise textual, o objetivo então pode ser a padronização dos diferentes textos que contenham o mesmo intuito;
- Gerar a partir da leitura automática de relatórios gerados por diferentes técnicos, relatórios que sejam formados por frases automáticas com base no intuito dado por cada técnico.
- Criar um *chatbot* que use a base com os perfis das diferentes intenções nas mensagens escritas pelos clientes e assim possa responder de forma automática e personalizada com base em cada intenção.
- Criar diferentes perfis de *chatbot* com base nos perfis comportamentais dos clientes com o objetivo de melhorar a qualidade de atendimento ao cliente.

7. REFERÊNCIAS BIBLIOGRÁFICAS

1. BISHOP, Christopher M. et al. **Neural networks for pattern recognition**. Oxford university press, 1995.
2. BRUNDAGE, Miles. Taking superintelligence seriously: Superintelligence: Paths, dangers, strategies by Nick Bostrom (Oxford University Press, 2014). **Futures**, v. 72, p. 32-35, 2015.
3. BURKOV, Andriy. **The hundred-page machine learning book**. Quebec City, Can.: Andriy Burkov, 2019.
4. CANONICO, Massimo; DE RUSSIS, Luigi. A comparison and critique of natural language understanding tools. **Cloud Computing**, v. 2018, p. 120, 2018.
5. CAVNAR, William B. et al. N-gram-based text categorization. In: **Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval**. 1994.
6. CHATTOPADHYAY, Manojit; DAN, Pranab K.; MAZUMDAR, Sitanath. Principal component analysis and self-organizing map for visual clustering of machine-part cell formation in cellular manufacturing system. In: **Systems Research Forum**. World Scientific Publishing Company, 2011. p. 25-51.
7. CHEN, Xinyi; YAN, Xuefeng. Using improved self-organizing map for fault diagnosis in chemical industry process. **Chemical engineering research and design**, v. 90, n. 12, p. 2262-2277, 2012.
8. CLARK, Alexander; FOX, Chris; LAPPIN, Shalom (Ed.). **The handbook of computational linguistics and natural language processing**. John Wiley & Sons, 2013.
9. DAVIDSON-PILON, Cameron. **Bayesian methods for hackers: probabilistic programming and Bayesian inference**. Addison-Wesley Professional, 2015.
10. DINH, Duy-Tai; FUJINAMI, Tsutomu; HUYNH, Van-Nam. Estimating the Optimal Number of Clusters in Categorical Data Clustering by Silhouette Coefficient. In: **International Symposium on Knowledge and Systems Sciences**. Springer, Singapore, 2019. p. 1-17.

11. ECK, Matthias; VOGEL, Stephan; WAIBEL, Alex. Low cost portability for statistical machine translation based on n-gram frequency and TF-IDF. In: **International Workshop on Spoken Language Translation (IWSLT) 2005**. 2005.
12. FELDMAN, Ronen et al. **The text mining handbook: advanced approaches in analyzing unstructured data**. Cambridge university press, 2007.
13. FERDOUS, Raihana et al. An efficient k-means algorithm integrated with Jaccard distance measure for document clustering. In: **2009 First Asian Himalayas International Conference on Internet**. IEEE, 2009. p. 1-6.
14. Future of data 2019. **The times**, Londres, 20 de março de 2019. Disponível em .<https://www.raconteur.net/future-data-2019>
15. GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep learning**. MIT press, 2016.
16. SETH, Grimes. Unstructured data and the 80 percent rule: investigating the 80%. **Clarabridge, Bridgepoints Q**, v. 3, 2008.
17. HAHSLER, Michael; PIEKENBROCK, Matthew; DORAN, Derek. dbscan: Fast density-based clustering with r. **Journal of Statistical Software**, v. 25, p. 409-416, 2019.
18. HARRINGTON, Peter. **Machine learning in action**. Manning Publications Co., 2012.
19. HASTIE, Trevor; TIBSHIRANI, Robert; FRIEDMAN, Jerome. **The elements of statistical learning: data mining, inference, and prediction**. Springer Science & Business Media, 2009.
20. HEMAVATHI, D.; SRIMATHI, H.; SORNALAKSHMI, K. A Hybrid Technique for Unsupervised Dimensionality Reduction by Utilizing Enriched Kernel Based PCA and DBSCAN Clustering Algorithm. In: **International Conference on Inventive Computation Technologies**. Springer, Cham, 2019. p. 476-488.
21. HOTHO, Andreas; NÜRNBERGER, Andreas; PAAß, Gerhard. A brief survey of text mining. In: **Ldv Forum**. 2005. p. 19-62.
22. HU, Wei; QU, Yuzhong. Discovering simple mappings between relational database schemas and ontologies. In: **The Semantic Web**. Springer, Berlin, Heidelberg, 2007. p. 225-238.

23. JAMES, Gareth et al. **An introduction to statistical learning**. New York: springer, 2013.
24. KANNAN, Vishwac Sena; SURTI, Tanvi Saumil. **Frequent sites based on browsing patterns**. U.S. Patent n. 10,375,186, 6 ago. 2019.
25. KLOBUCAR, Damir; SUBASIC, Marko. Using self-organizing maps in the visualization and analysis of forest inventory. **iForest-Biogeosciences and Forestry**, v. 5, n. 5, p. 216, 2012.
26. KOLLER, Daphne et al. **Introduction to statistical relational learning**. MIT press, 2007.
27. LESKOVEC, Jure; RAJARAMAN, Anand; ULLMAN, Jeffrey David. **Mining of massive data sets**. Cambridge university press, 2019.
28. LOPER, Edward; BIRD, Steven. NLTK: the natural language toolkit. **arXiv preprint cs/0205028**, 2002.
29. MAAS, Andrew L. et al. Learning word vectors for sentiment analysis. In: **Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1**. Association for Computational Linguistics, 2011. p. 142-150.
30. MARSLAND, Stephen. **Machine learning: an algorithmic perspective**. Chapman and Hall/CRC, 2014.
31. MUELLER, John Paul; MASSARON, Luca. **Machine learning for dummies**. John Wiley & Sons, 2016.
32. MÜLLER, Oliver et al. Using text analytics to derive customer service management benefits from unstructured data. **MIS Quarterly Executive**, v. 15, n. 4, p. 243-258, 2016.
33. NG, Andrew. Machine learning yearning. **URL: http://www. mlyearning. org/(96)**, 2017.
34. NOOTHIGATTU, Ritesh et al. A voting-based system for ethical decision making. In: **Thirty-Second AAAI Conference on Artificial Intelligence**. 2018.
35. NORD, Christiane. **Text analysis in translation: Theory, methodology, and didactic application of a model for translation-oriented text analysis**. Rodopi, 2005.
36. RAHMAN, A. M.; AL MAMUN, Abdullah; ISLAM, Alma. Programming challenges of chatbot: Current and future prospective. In: **2017 IEEE**

Region 10 Humanitarian Technology Conference (R10-HTC). IEEE, 2017. p. 75-78.

37. RAJARAMAN, A. Ullman. JD (2011)." Data Mining. **Mining of Massive Datasets**, p. 1-17.
38. RICHERT, Willi. **Building machine learning systems with Python**. Packt Publishing Ltd, 2013.
39. SANTOS, Breno Santana et al. Comparing text mining algorithms for predicting irregularities in public accounts. In: **Proceedings of the XI Brazilian Symposium on Information Systems (SBSI 2015)**. 2015. p. 667-674.
40. SANTURKAR, Shibani et al. How does batch normalization help optimization?. In: **Advances in Neural Information Processing Systems**. 2018. p. 2483-2493.
41. SCULLEY, David. Web-scale k-means clustering. In: **Proceedings of the 19th international conference on World wide web**. 2010. p. 1177-1178.
42. SHALEV-SHWARTZ, Shai; BEN-DAVID, Shai. **Understanding machine learning: From theory to algorithms**. Cambridge university press, 2014.
43. SIAU, Keng; WANG, Weiyu. Building trust in artificial intelligence, machine learning, and robotics. **Cutter Business Technology Journal**, v. 31, n. 2, p. 47-53, 2018.
44. STECANELLA, Rodrigo. **MonkeyLearn**, Palo Alto, 21 de setembro de 2017. Disponível em <https://monkeylearn.com/blog/beginners-guide-text-vectorization/>
45. TARASIEV, Andrey et al. Application of Stemming Methods to Development a Module of a Post-processing of Recognized Speech in Intelligent Automated System for Dialogue and Decision-Making in Real Time. In: **2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON)**. IEEE, 2019. p. 0104-0109.
46. TOADER, Diana-Cezara et al. The Effect of Social Presence and Chatbot Errors on Trust. **Sustainability**, v. 12, n. 1, p. 256, 2020.

47. TOMLINSON, Stephen. Lexical and algorithmic stemming compared for 9 European languages with Hummingbird SearchServer TM at CLEF 2003. In: **Workshop of the Cross-Language Evaluation Forum for European Languages**. Springer, Berlin, Heidelberg, 2003. p. 286-300.
48. XIE, Junyuan; GIRSHICK, Ross; FARHADI, Ali. Unsupervised deep embedding for clustering analysis. In: **International conference on machine learning**. 2016. p. 478-487.
49. XINYI, Chen; XUEFENG, Y. A. N. Fault diagnosis in chemical process based on self-organizing map integrated with fisher discriminant analysis. **Chinese Journal of Chemical Engineering**, v. 21, n. 4, p. 382-387, 2013.
50. VILLMANN, Th; BAUER, H.-U. Applications of the growing self-organizing map. **Neurocomputing**, v. 21, n. 1-3, p. 91-100, 1998.
51. ZHAO, Wei et al. Towards scalable and reliable capsule networks for challenging NLP applications. **arXiv preprint arXiv:1906.02829**, 2019.