UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
ENG. DE CONTROLE E AUTOMAÇÃO

LUCAS BORTOLANZA GRAZZIOTIM - 00275629

AGRUPAMENTO DE SENTENÇAS SEMANTICAMENTE SIMILARES APLICADO À DESCOBERTA DE NOVAS INTENÇÕES PARA *CHATBOTS* COGNITIVOS

Porto Alegre
2022

LUCAS BORTOLANZA GRAZZIOTIM - 00275629

AGRUPAMENTO DE SENTENÇAS SEMANTICAMENTE SIMILARES APLICADO À DESCOBERTA DE NOVAS INTENÇÕES PARA *CHATBOTS* COGNITIVOS

Trabalho de Conclusão de Curso (TCC-CCA) apresentado à COMGRAD-CCA da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de *Bacharel em Eng. de Controle e Automação* .

ORIENTADOR:
Prof. Dr. Marcelo Götz

CO-ORIENTADOR(A):
Profª. Drª. Ana Paula Appel

Porto Alegre
2022

LUCAS BORTOLANZA GRAZZIOTIM - 00275629

AGRUPAMENTO DE SENTENÇAS SEMANTICAMENTE SIMILARES APLICADO À DESCOBERTA DE NOVAS INTENÇÕES PARA *CHATBOTS* COGNITIVOS

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção dos créditos da Disciplina de TCC do curso *Eng. de Controle e Automação* e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn – Paderborn, Alemanha

Banca Examinadora:

Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn – Paderborn, Alemanha

Prof. Dr. João Cesar Netto, UFRGS
Doutor pela Universidade Católica da Lovaina – Louvain-la-Neuve, Bélgica

Prof. Dr. Rafael Antônio Comparsi Laranja, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

_____
Mário Roland Sobczyk Sobrinho
Coordenador de Curso
Eng. de Controle e Automação

Porto Alegre, maio de 2022.

AGRADECIMENTOS

À minha mãe, pelo carinho e pelo apoio incondicional em todas as etapas da minha vida.

À minha vó, pela ternura e pelo zelo no meu crescimento.

Ao professor orientador, pelo acompanhamento e pela revisão minuciosa do trabalho.

À professora co-orientadora, pela troca de ideias, pela disponibilidade e pelo interesse no desenvolvimento do trabalho.

Aos demais professores, por todo o conhecimento compartilhado.

Aos colegas, pela troca de experiências.

Aos amigos, por tornarem a caminhada mais agradável.

RESUMO

O trabalho apresenta uma solução de automação para o processo de descoberta de novas intenções para *chatbots* cognitivos. Para tanto, representações vetoriais das mensagens enviadas por usuários a um *chatbot* cognitivo são obtidas por meio de um modelo de linguagem neural para a língua portuguesa brasileira baseado no BERT, o BERTimbau. Na sequência, a fim de identificar os assuntos requisitados pelos usuários e viabilizar a construção de intenções, mensagens semelhantes são agrupadas por meio da execução de um algoritmo não supervisionado de agrupamento hierárquico aglomerativo sobre as suas representações vetoriais. Para que os agrupamentos obtidos possam ser investigados de uma maneira acessível, foi desenvolvida uma ferramenta de visualização na forma de uma aplicação *web*. Em um estudo de caso, a aplicação da solução proposta foi capaz de agrupar sentenças com sentido semelhante mesmo quando construídas com palavras distintas, possibilitando, com sucesso, a identificação de intenções a serem inseridas ao *chatbot* cognitivo a partir de mensagens enviadas pelos seus usuários.

Palavras-chave: Processamento de linguagem natural, *chatbot* cognitivo, mineração de intenções, BERT, incorporação de sentenças, semelhança textual semântica, agrupamento de sentenças.

ABSTRACT

The work presents an automation solution for the process of discovering new intents for cognitive chatbots. To do so, sentence embeddings of messages sent by users to a cognitive chatbot are obtained by means of a neural language model for Brazilian Portuguese based on BERT, which is known as BERTimbau. Subsequently, in order to identify the subjects requested by users and enable the creation of intents, similar messages are grouped together by the execution of an unsupervised agglomerative hierarchical clustering algorithm over their vector representations. So that the clusters obtained can be investigated in an accessible way, a visualization tool was developed as a web application. In a case study, the application of the proposed solution was able to group sentences with similar meaning even when made up of different words, successfully enabling the identification of intents to be added to the cognitive chatbot from messages sent by its users.

Keywords: natural language processing, cognitive chatbot, intent mining, BERT, sentence embeddings, semantic textual similarity, sentence clustering.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API	*Application Programming Interface* (Interface de Programação de Aplicações)
BERT	*Bidirectional Encoder Representations from Transformers* (Representações Bidirecionais de Codificador obtidas de Transformadores)
BoW	*Bag-of-Words* (Saco-de-Palavras)
MLM	*Masked Language Modeling* (Modelagem de Linguagem Mascarada)
NLP	*Natural Language Processing* (Processamento de Linguagem Natural)
NSP	*Next Sentence Prediction* (Predição de Próxima Sentença)

1 INTRODUÇÃO

Chatbots cognitivos já são uma realidade no cenário de autoatendimento brasileiro, seja no suporte interno aos funcionários de uma instituição ou em serviços de atendimento externo. Utilizando processamento de linguagem natural (NLP, do termo em inglês *Natural Language Processing*), ramo da inteligência artificial dedicada a possibilitar que computadores entendam a linguagem humana (também chamada de linguagem natural) (BROWNLEE, 2017c), é possibilitada a automação de atendimentos repetitivos por meio de plataformas digitais de conversação. Dessa forma, é possível escalar o serviço de atendimento, reduzir custos operacionais, oferecer um meio de comunicação disponível 24 horas por dia, prover respostas em tempo real a um número indefinido de usuários simultaneamente, reduzir o tempo de espera do encaminhamento a um suporte humano (ao qual são direcionadas as solicitações mais complexas que o *chatbot* não foi capaz de resolver) e aumentar a satisfação geral de quem utiliza os canais de atendimento.

A alta disponibilidade e escalabilidade do serviço resultam em números de atendimento expressivos. No primeiro semestre de 2020, um *chatbot* cognitivo de uma varejista brasileira registrou uma média de 8,5 milhões de interações ao mês, relativos a uma média de 1,4 milhão de atendimentos ao mês (BERETTA, 2020). Segundo a Pesquisa Panorama Mobile Time — Mapa do Ecossistema Brasileiro de Bots — Agosto de 2021 (PAIVA, 2021), realizada entre junho e julho de 2021, o volume médio mensal de mensagens trafegadas por *bots* em 2021 foi de 2,8 bilhões, com crescimento de 27% em relação ao ano anterior. Já em 2020, impulsionadas pelas restrições impostas pela pandemia de COVID-19, que acentuou a necessidade de transformação digital das organizações, as empresas pesquisadas registraram o valor de 2,2 bilhões, com crescimento de 120% em relação ao ano anterior.

Embora muitas das solicitações sejam resolvidas pelo *chatbot*, há interações que não são bem-sucedidas, seja porque o seu modelo de NLP não foi capaz de entender a mensagem do usuário ou porque a ferramenta ainda não é capaz de tratar sobre aquele assunto. Por conseguinte, faz-se necessário coletar os registros de conversação para agrupar as mensagens que tratam do mesmo assunto e, então, melhorar o modelo no entendimento dos temas já conhecidos pelo *chatbot* ou adicionar novos assuntos ao seu repertório.

Contudo, na escala de milhares ou milhões de mensagens ao mês, é impraticável realizar essa tarefa manualmente. A mineração de registros de conversação é um processo moroso e cansativo, exigindo muitas horas de investigação e um bom planejamento do time na definição dos conceitos que regem a identificação de sentenças consideradas similares. O presente trabalho, portanto, visa a resolver esse problema ao apresentar uma solução de automação do processo de agrupamento de mensagens pertencentes a tópicos semelhantes, facilitando a criação de novas intenções ou mesmo a melhoria das já existentes.

2 REVISÃO BIBLIOGRÁFICA

A seguir são introduzidos conceitos necessários ao entendimento do trabalho. A Seção 2.1 trata sobre como um *chatbot* cognitivo é capaz de entender o conteúdo da mensagem enviada pelo usuário. Já a Seção 2.2 aborda diferentes procedimentos que podem ser empregados na identificação de similaridade entre documentos textuais ou entre palavras do documento. Por fim, a Seção 2.3 apresenta mais detalhes sobre o modelo empregado durante o desenvolvimento deste trabalho.

2.1 Intenções em *Chatbots* Cognitivos

Uma das características que distinguem *chatbots* cognitivos daqueles baseados em regras é o entendimento da intenção da mensagem do usuário (RAMOS, 2020). Nos *chatbots* baseados em regras, são oferecidas ao usuário listas de opções pré-determinadas, permitindo que a conversa prossiga apenas pela seleção dessas opções, seja pelo pressionamento de botões ou pelo envio de números ou palavras-chave referentes às opções. Nesse modelo rígido de interação, o usuário é obrigado a explorar menus e submenus para encontrar a informação que precisa. Já nos *chatbots* cognitivos, o usuário fica livre para escrever, em linguagem natural, o que ele deseja obter daquela conversa, tornando a experiência, quando bem projetada, mais humanizada, agradável e ágil.

Para que o *chatbot* seja capaz de entender a mensagem do usuário, são utilizadas intenções. Uma intenção é composta de diferentes frases que representam distintas formas do usuário expressar um mesmo objetivo (NICKERSON, 2020). A intenção representa um assunto, um tópico, sobre o qual o *chatbot* sabe conversar. O conjunto de várias intenções permite, então, que o modelo de NLP entenda a mensagem do usuário para direcioná-lo ao fluxo de conversação apropriado para tratar daquele assunto.

A definição de quantas e quais intenções criar, juntamente do grau de especificidade delas, depende dos objetivos de negócio, do volume de usuários que procura por aquele assunto e da capacidade do time de desenvolvimento na manutenção do modelo. Um *chatbot* de certa empresa, por exemplo, pode ter uma intenção para perguntas sobre o horário de funcionamento da sua loja e outra sobre o seu endereço, ou pode agrupá-las em uma única intenção, mais genérica, para tratar sobre informações gerais da loja.

Quando o usuário envia uma mensagem ao *chatbot*, o modelo de NLP deve então resolver um problema, de aprendizagem de máquina, supervisionado de classificação, no qual uma classe, que representa a intenção entendida, deve ser atribuída à mensagem (HAJJAR, 2022). Para se obter tal resultado, para cada intenção é calculada uma pontuação padronizada (de 0 a 1), comumente chamada de pontuação de confiança (RÖSCH, 2022), com base na semelhança entre a frase do usuário e as frases que compõem a intenção. Aquela que apresentar maior confiança será então a intenção atribuída àquela frase.

No entanto, quando a maior confiança é pequena, ou seja, abaixo de determinado limite (o qual é específico para cada *chatbot*), a classificação não é relevante. Nesse cenário, o modelo não foi capaz de entender, com convicção, a mensagem enviada pelo usuário. Isso ocorre quando a mensagem difere das frases quem compõem a intenção que se esperava ter sido identificada (sendo necessário melhorar a construção da intenção), quando não há uma intenção para aquele assunto ou quando a mensagem enviada não faz sentido, sendo impossível, mesmo a um humano, entender o seu conteúdo. Ademais, também podem haver casos nos quais, por um desbalanceamento na definição das intenções, a mensagem seja erroneamente classificada.

2.2 Similaridade entre Documentos

Mensagens de texto são dados não estruturados com elevada complexidade de construção e representação, para os quais métodos de aprendizagem de máquina podem ser empregados na identificação de frases semelhantes. Contudo, para que os algoritmos possam manipular essa informação, os textos precisam ser convertidos a uma representação numérica (BROWNLEE, 2017a) com a qual os computadores possam trabalhar.

Na sequência, são apresentadas algumas das abordagens mais difundidas no alcance de tal objetivo. Nesse contexto, é comum denominar documento cada entrada textual sendo observada. Assim, no cenário abordado, um documento representa uma mensagem enviada pelo usuário ao *chatbot*.

2.2.1 *Bag-of-Words*

O modelo de *Bag-of-Words* (BoW, traduzido do inglês como Saco-de-Palavras) é um método simples de se obter uma representação numérica de textos, no qual descarta-se a informação sobre a ordem de ocorrência das palavras no documento, como se elas fossem colocadas em um saco (BROWNLEE, 2017a). Assim, utiliza-se apenas a contagem das suas ocorrências como parâmetro, de modo que dois documentos são considerados similares quando contêm as mesmas palavras com contabilizações semelhantes. Nota-se que tal abordagem permite inferir a semelhança entre documentos, mas não entre palavras.

Inicialmente, o algoritmo cria um vocabulário de palavras presentes nos documentos. Nessa etapa, é comum realizar operações de limpeza dos textos, tais como: remoção de capitalização (troca de letras maiúsculas por minúsculas), de pontuação e de *stopwords* (palavras com baixo valor semântico), lematização ou stemização (do inglês, *stemming*).

Na sequência, a cada documento é atribuído um vetor relativo à contabilização das suas palavras. Como cada posição do vetor refere-se a uma das palavras do vocabulário, um grande vocabulário resulta em vetores esparsos de alta dimensão, os quais podem degradar os resultados obtidos pelo algoritmo que fará o agrupamento dos vetores, e portanto, dos documentos. O valor de cada elemento do vetor pode ser obtido da ocorrência da respectiva palavra no documento, da sua frequência em relação às outras palavras ou do cálculo de TF-IDF (*Term Frequency – Inverse Document Frequency*, traduzido do inglês como Frequência do Termo – Frequência Inversa de Documento) (BROWNLEE, 2017a).

Embora simples, esse método não se adapta bem ao cenário de estudo. As frases enviadas a *chatbots* possuem poucas palavras, de modo que é bem possível que frases similares não tenham palavras em comum (como em “quero assinar um plano de dados móveis mais rápido” e “preciso aumentar a velocidade de *internet* do meu contrato”). De maneira contrária, frases distintas podem ter intersecção no seu conjunto de palavras (como em “como fazer assinatura de um plano de *internet* de alta velocidade” e “qual a

velocidade do plano de *internet* da minha assinatura?"). Ademais, tratando-se de mensagens de texto com o público em geral, espera-se que vários dos documentos contenham erros ortográficos, dificultando a identificação e a contabilização de uma mesma palavra.

2.2.2 *Word Embeddings*

Técnicas de obtenção de *word embeddings* (traduzido do inglês como incorporação de palavras) englobam um conjunto de métodos nos quais palavras de um vocabulário são representadas por vetores de números reais (PIETRO, 2020), de modo que palavras semanticamente semelhantes estejam próximas no espaço vetorial. Tal construção baseia-se na hipótese distribucional, segundo a qual palavras que ocorrem em contextos similares possuem sentido similar. Assim, o significado da palavra é entendido observando-se as palavras vizinhas que a acompanham (JURAFSKY; MARTIN, 2022).

Como essa operação é executada para cada palavra, para se obter a representação vetorial do documento inteiro é então necessário reduzir os vetores para um vetor resultante por meio de alguma função de agregação. Tais vetores são densos e de dimensionalidade bem menor que a produzida pelo BoW (embora cada dimensão não tenha uma clara interpretação), os quais tendem a gerar melhores resultados em tarefas de NLP (JURAFSKY; MARTIN, 2022).

Sendo frequentemente empregados em aplicações interessadas no sentido das palavras, os modelos de *word embeddings* podem ser divididos em dois tipos. O primeiro, explorado na Seção 2.2.2.1 produz incorporações estáticas das palavras (em inglês, *static word embeddings*). O segundo, explorado na Seção 2.2.2.2, produz incorporações dinâmicas, mais conhecidas como contextuais (em inglês, *contextual word embeddings*).

2.2.2.1 Static Word Embeddings

Quando o método gera, invariavelmente, um único vetor para determinada palavra, denomina-se que a representação vetorial daquela palavra é estática. De tal forma, independentemente do contexto no qual ela está inserida, o resultado é unívoco. Fazem parte desse grupo famílias de algoritmos como: word2vec (MIKOLOV et al., 2013), GloVe (PENNINGTON; SOCHER; MANNING, 2014) e fastText (BOJANOWSKI et al., 2016).

A obtenção de tais vetores é realizada de maneira iterativa. Partindo-se de uma disposição aleatória, eles são movidos no espaço vetorial com base nas demais palavras que circundam a palavra sendo analisada. Assim, palavras com sentido similar recebem posições próximas no espaço vetorial, pois são acompanhadas por palavras (contextos) semelhantes (BROWNLEE, 2017b).

Vale ressaltar, todavia, que o contexto no qual a palavra está inserida é considerado somente durante a etapa de treinamento do modelo, na qual são construídos os vetores. No momento de conversão das palavras de um documento a vetores, as palavras vizinhas são descartadas, impossibilitando a obtenção de diferentes vetores para palavras homônimas.

Dessa forma, atribui-se o mesmo vetor para a palavra "banco", seja ela referente a uma instituição financeira, a um assento, ou ainda quando acompanhada de locuções adjetivas como "de dados". Nesse cenário, o modelo busca construir um vetor para a palavra que possa acomodar todas as suas possíveis interpretações, tal que palavras como "dinheiro", "cadeira" e "armazenamento" sejam similares a "banco" com base nas suas co-ocorrências nos documentos utilizados para o seu treinamento. Ademais, "banco" pode ser entendido como uma conjugação do verbo "bancar" (sustentar financeiramente), de modo que não apenas um novo sentido é dado à palavra, como também uma função sintática distinta.

Por fim, erros ortográficos, se não estiverem presentes nos documentos de treinamento, geram palavras fora do vocabulário, para as quais não é possível obter um vetor (com exceção do fastText, no qual cada palavra é representada por sub-palavras (BOJANOWSKI et al., 2016)). Ou ainda, é possível que o erro gere outra palavra para a qual existe uma representação vetorial, mas que carrega outro sentido, deteriorando a conversão do texto (a palavra "dívida", por exemplo, escrita sem acento agudo, transforma-se no modo imperativo do verbo "dividir").

2.2.2.2 Contextual Word Embeddings

Em contrapartida, quando a representação vetorial considera o contexto do texto, obtêm-se diferentes vetores para uma mesma palavra para cada vez em que ela é acompanhada por um distinto conjunto de palavras (JURAFSKY; MARTIN, 2022). Assim, é possível obter diferentes vetores a partir de uma mesma palavra, de modo a representar seus diferentes significados.

Modelos como ELMo (PETERS et al., 2018), GPT (RADFORD; NARASIMHAN, 2018) e BERT (DEVLIN; CHANG et al., 2019) são capazes de gerar representações vetoriais contextuais utilizando um modelo de linguagem (no caso, um modelo de linguagem neural) como objetivo de aprendizagem de uma rede neural (GOMEZ-PEREZ; DENAUX; GARCIA-SILVA, 2020). Um modelo de linguagem é uma distribuição de probabilidades que prediz a probabilidade de determinada sequência de palavras aparecer na linguagem de estudo, sendo obtido por métodos não supervisionados de aprendizagem de máquina (LIU; KUSNER; BLUNSOM, 2020).

As representações estáticas, apresentadas na Seção 2.2.2.1, funcionam como um mapeamento de palavras para vetores, de modo que o modelo em si, após o treinamento para obtenção dos vetores, é descartado. Já as representações contextuais dependem da utilização do modelo de linguagem treinado para calcular, no momento de análise, o vetor referente à palavra (SIEG, 2019). Tais representações conseguem capturar várias propriedades sintáticas e semânticas, de modo que elas atingem performance no estado da arte em uma grande variedade de tarefas de NLP (LIU; KUSNER; BLUNSOM, 2020). Contudo, como os modelos precisam observar todas as palavras da sentença, a obtenção dos vetores exige maior gasto computacional.

2.3 *Bidirectional Encoder Representations from Transformers*

O BERT (*Bidirectional Encoder Representations from Transformers*, traduzido do inglês como Representações Bidirecionais de Codificador obtidas de Transformadores) é o primeiro modelo de linguagem não supervisionado, profundamente bidirecional e pré-treinado utilizando conjuntos de textos não anotados (DEVLIN; CHANG, 2018). Desenvolvido pela Google, o BERT foi incorporado ao seu serviço de busca, o Google Search, no final de 2019 (NAYAK, 2019). Segundo a empresa, o uso do modelo possibilita que o contexto das palavras na frase seja entendido pela ferramenta. Assim, no lugar da popular busca por palavras-chave, é possível fazê-la utilizando uma linguagem mais natural aos seres humanos.

Por profundamente bidirecional, refere-se ao fato de que o modelo considera ambas as palavras à esquerda e à direita daquela sendo observada. Demais modelos (como o GPT) observam apenas uma direção ou são rasamente bidirecionais (como o ELMo), concatenando os resultados obtidos independentemente para cada direção (DEVLIN; CHANG et al., 2019).

Um dos maiores desafios em NLP é obter um grande volume de dados de treinamento específicos à tarefa em questão. Como o pré-treinamento do BERT é não supervisionado, é possível aproveitar recursos gratuitos e já existentes, como a Wikipédia, para pré-treinar o modelo de modo que a rede neural aprenda a linguagem de maneira genérica. Assim, obtém-se um modelo de linguagem de propósito geral ao qual, na sequência, são feitos pequenos ajustes na rede neural e novas rodadas de treinamento a fim de atender à tarefa de NLP de interesse (DEVLIN; CHANG, 2018) em um processo conhecido como *fine-tuning* (do inglês, ajuste fino). Como o modelo já conhece a linguagem de maneira genérica, grande parte da rede neural já está com seus parâmetros configurados, de modo que para uma tarefa de classificação, por exemplo, pode ser necessário treinar apenas a última camada adicionada à rede. Dessa forma, reduz-se significativamente o tempo e o poder de processamento necessários ao treinamento de modelos com elevado número de parâmetros. Nos últimos anos, diversos métodos têm sido elaborados utilizando essa técnica, chamada de aprendizagem por transferência, os quais têm aprimorado o estado da arte em uma ampla variedade de tarefas de NLP (RUDER et al., 2019).

O pré-treinamento do BERT é composto de duas tarefas de aprendizagem não supervisionadas, a saber: *Masked Language Modeling* (MLM, traduzido do inglês como Modelagem de Linguagem Mascarada) e *Next Sentence Prediction* (NSP, traduzido do inglês como Predição de Próxima Sentença). Na primeira, a rede neural deve predizer o valor dos *tokens* (segmentos do texto, explorados na sequência desta seção) que foram mascarados aleatoriamente, de modo que o modelo de linguagem aprende a relação entre as palavras e o seu contexto. Na segunda, aprende-se a relação entre duas sentenças treinando-se um classificador binário que deve predizer se determinada sentença é a próxima daquela sendo observada (DEVLIN; CHANG et al., 2019).

Os *tokens* são as unidades básicas que compõem um texto, obtidos pela segmentação (tokenização) do texto em palavras, sub-palavras, letras, números ou símbolos (PAI, 2020). No caso do BERT, é utilizado o algoritmo WordPiece para segmentar o texto em sub-palavras (DEVLIN; CHANG et al., 2019), de modo que os N *tokens* mais frequentes compõem o vocabulário do modelo de linguagem. Após determinado o tamanho do vocabulário, o algoritmo é treinado a fim de definir qual conjunto de *tokens* minimiza a quantidade de *tokens* necessários para representar todo o corpo textual de treinamento (WU et al., 2016), dado que várias palavras podem ser construídas pela combinação de termos comuns (as sub-palavras). Dessa estratégia de segmentação, não ocorre o descarte de palavras fora do vocabulário, pois caso uma palavra desconhecida ou com erros ortográficos seja alimentada ao tokenizador (o modelo de tokenização), ela será segmentada em sub-palavras e/ou caracteres, evitando-se a perda de informação.

O termo *transformer* refere-se a uma arquitetura de rede neural desenvolvida para problemas de tradução, a qual apresenta resultados melhores que as arquiteturas empregadas até então (VASWANI et al., 2017), como a LSTM utilizada pelo ELMo (PETERS et al., 2018). Tais melhorias incluem modelagem mais efetiva da dependência entre palavras em frases longas (POGIATZIS, 2019), bidirecionalidade no entendimento do contexto e treinamento mais rápido e eficiente ao processar as palavras simultaneamente (CODEEMPORIUM, 2020). Composta por duas partes, o *encoder* (do inglês, codificador) transforma as palavras de entrada em vetores, enquanto o *decoder* (do inglês, decodificador) transforma tais vetores em palavras de saída (CODEEMPORIUM, 2020). No caso do BERT, a arquitetura da sua rede neural é composta de vários *encoders* em sequência, apresentando dois tamanhos distintos na sua publicação: BERT-Base e BERT-Large (DEVLIN; CHANG et al., 2019) (ou seja, do inglês, tamanhos básico e grande).

3 DESENVOLVIMENTO

As seções a seguir descrevem a solução desenvolvida, a qual visa, para um determinado conjunto de mensagens, a agrupar aquelas com sentido semelhante, ou seja, que tratam sobre um assunto comum. Para tanto, a Seção 3.1 trata sobre o modelo de linguagem escolhido para gerar a representação vetorial das mensagens, a qual é analisada com mais detalhes na Seção 3.2. Na sequência, a Seção 3.3 descreve a métrica empregada para computar a similaridade entre dois vetores e, por consequência, das suas respectivas mensagens. A Seção 3.4 faz uma análise sobre o conjunto de dados utilizado, composto por mensagens de usuários enviadas a um *chatbot* cognitivo, e sobre a influência de alguns parâmetros do tokenizador no processo de segmentação das sentenças. Ademais, é apresentada a distribuição de similaridades entre as mensagens. Em seguida, a Seção 3.5 trata do método de aprendizagem de máquina não supervisionado utilizado no agrupamento das mensagens do conjunto de dados. Por fim, a Seção 3.6 apresenta a ferramenta desenvolvida para apresentar os resultados obtidos de uma maneira acessível e interativa.

3.1 Modelo de Linguagem

Dadas as características apresentadas na Seção 2.3, escolheu-se o BERT como o modelo de linguagem para se obter o entendimento do conteúdo das mensagens. Embora disponibilizado de maneira *open source* (do inglês, código aberto) no GitHub (DEVLIN; CHANG, 2018), não há um modelo específico para a língua portuguesa brasileira. Há modelos para as línguas inglesa e chinesa e modelos multilíngue, os quais são treinados em 104 línguas diferentes, dentre elas o português, embora não seja descrita qual variante (DEVLIN; CHANG et al., 2019).

No entanto, foram encontrados dois modelos BERT treinados, por outros pesquisadores, apenas em português brasileiro, os quais são apelidados pelos seus autores como BERTimbau-Base e BERTimbau-Large (SOUZA; NOGUEIRA; LOTUFO, 2020) (junção de BERT com "berimbau"). Refletindo os distintos tamanhos de arquitetura da rede neural dos modelos originais (BERT-Base e BERT-Large), os modelos melhoraram o estado da arte, apresentando performance superior aos modelos BERT multilíngue, em todas as três tarefas de NLP avaliadas: similaridade textual entre sentenças, reconhecimento de vinculação textual e reconhecimento de entidade mencionada.

Treinar um modelo BERT exige elevados recursos computacionais. Os modelos originais em língua inglesa, BERT-Base (110 milhões de parâmetros) e BERT-Large (340 milhões de parâmetros), foram pré-treinados utilizando-se o BooksCorpus (800 milhões de palavras) e a Wikipédia (2.500 milhões de palavras) (DEVLIN; CHANG et al., 2019). Tal processo, é caro e demorado. Estima-se que um pré-treinamento do BERT-Large, empregando a Cloud TPU v2 — *Cloud Tensor Processing Units*, do inglês, Unidade de

Processamento de Tensor em Nuvem, criada para o desenvolvimento de modelos de aprendizagem de máquina de última geração com serviços de inteligência artificial no Google Cloud (CLOUD..., 2022) — por quatro dias, custa US$ 6.912. Já o BERT-Base, utilizando configurações mais modestas, custa US$ 500 para um treinamento de duas semanas (PENG; SARAZEN, 2019).

Assim, havendo a disponibilidade de modelos *open source* prontos, com pré-treinamento na linguagem de estudo, eficácia comprovada e gratuitos, optou-se pelo uso do BERTimbau. Quanto aos dois tamanhos de arquitetura, escolheu-se a variante Base, pois seu menor número de parâmetros em relação à Large resulta em processamentos mais rápidos durante o desenvolvimento do trabalho, sem perda de generalidade.

Neste estudo, para realizar a tarefa de agrupamento de mensagens semelhantes, foi utilizado um conjunto de dados disponibilizado por uma instituição financeira, o qual é composto por mensagens enviadas pelos seus clientes ao seu *chatbot* cognitivo por meio do WhatsApp. Embora haja certa especificidade quanto ao domínio do conjunto de dados utilizados, sendo composto de questionamentos e pedidos relacionados ao universo financeiro, não se espera que tal população utilize um linguajar extremamente técnico. Pelo contrário, espera-se que as mensagens, enviadas pela população em geral em um aplicativo de mensagens, tenham termos comuns e construções simples. Por conseguinte, para o cenário abordado, não é necessário construir um modelo de linguagem para um domínio específico, tal como o BioBERT, treinado ao entendimento de textos de biomedicina (LEE et al., 2019). Logo, supõe-se que um modelo de linguagem genérica em língua portuguesa, como o BERTimbau-Base, seja suficiente ao entendimento do conteúdo das mensagens.

Embora as sentenças utilizadas tratem sobre assuntos relacionados ao setor financeiro, ressalta-se que *chatbots* cognitivos são aplicáveis a qualquer domínio, em especial à engenharia. Potencializado pela associação a programas que convertem a fala em texto, e vice-versa, o *chatbot* pode ser utilizado como uma interface conversacional para enviar comandos destinados ao acionamento de atuadores, tal como em sistemas de automação residencial. Ademais, seu uso pode facilitar ao profissional de engenharia a procura por informações necessárias à instalação, à operação e à manutenção de máquinas e equipamentos, servindo como intermediário entre a pessoa e uma base de conhecimento. Em um cenário aplicado à indústria automobilística, por exemplo, os clientes de uma fabricante de veículos podem rapidamente acessar informações contidas no manual do veículo ao perguntar, diretamente ao *chatbot* cognitivo, a informação de interesse (VOLKSWAGEN..., 2017).

Os modelos BERTimbau estão disponíveis para *download* no GitHub, com versões compatíveis às bibliotecas *open source* para aprendizagem de máquina PyTorch e TensorFlow. No entanto, o uso da versão para PyTorch é facilitado pela compatibilidade com a biblioteca Transformers, de modo que os modelos estão disponíveis publicamente como modelos da comunidade (SOUZA; NOGUEIRA; LOTUFO, 2020).

A biblioteca Transformers é uma biblioteca de NLP, desenvolvida pela Hugging Face (THE..., 2022) para a linguagem de programação Python, que fornece APIs (plural de API — *Application Programming Interface*, traduzida do inglês como Interface de Programação de Aplicações) que facilitam o *download* e o treinamento de modelos no estado da arte e pré-treinados, os quais podem ser utilizados em tarefas envolvendo texto, imagem e áudio (TRANSFORMERS..., 2022). Embora tenham nomes similares, não se deve confundir a biblioteca Transformers com a arquitetura de rede neural Transformer.

3.2 Representação Vetorial das Mensagens

Por utilizar a mesma arquitetura do BERT, diferenciando apenas na linguagem dos textos utilizados durante o pré-treinamento, o BERTimbau pode ser instanciado no código utilizando-se as mesmas classes da biblioteca Transformers destinadas ao BERT. Além da classe base que representa o modelo, há classes destinadas a tarefas específicas, as quais já possuem uma camada extra ao final da rede neural a fim de realizar o ajuste fino do modelo à tarefa de NLP de interesse (BERT..., 2022). Nenhuma delas, contudo, é destinada a tarefas não supervisionadas de agrupamento. Assim, pode-se utilizar a rede neural base de modo a se obter representações vetoriais das mensagens presentes no conjunto de dados, as quais são agrupadas por outro algoritmo numa etapa posterior.

O primeiro passo para se obter a representação vetorial de um texto, utilizando um modelo do tipo BERT, consiste da sua tokenização, a qual deve seguir a mesma metodologia, baseada no WordPiece, que foi utilizada no momento de pré-treinamento daquele modelo. Assim, juntamente do BERTimbau, é disponibilizado pela biblioteca Transformers o seu respectivo modelo de tokenização.

Nessa etapa, além de segmentar o texto em sub-palavras, o tokenizador adiciona *tokens* especiais ao resultado. Ao seu início, inclui-se um *token* especial de classificação ([CLS]), utilizado para tarefas de classificação. Já ao seu final, inclui-se um *token* especial de separação ([SEP]). Caso o texto seja composto por um par de sentenças, esse mesmo *token* é inserido entre elas (DEVLIN; CHANG et al., 2019). Na sequência, a lista de *tokens* é convertida a uma lista de números inteiros (um vetor), na qual cada elemento representa o índice daquele *token* no vocabulário (MCCORMICK, 2019).

Todas as listas de *tokens* obtidas do conjunto de dados devem ter o mesmo comprimento, de modo que os textos mais curtos devem ser preenchidos com o *token* especial de preenchimento ([PAD]). Assim, para cada entrada no conjunto de dados, um segundo vetor é criado, com valor 1 nas posições com *tokens* significativos e com valor 0 nas posições com *token* de preenchimento. Por fim, um terceiro vetor é criado, com valor 0 nas posições relativas aos *tokens* da primeira sentença e com valor 1 nas da segunda sentença (quando ela existir). No caso abordado, mesmo que a mensagem seja composta, gramaticamente, por mais de uma frase, ela é considerada como uma única sentença, pois, para a tarefa de NLP abordada, não é de interesse saber a probabilidade daquela sequência de frases ocorrer na língua portuguesa. Assim, daqui em diante, cada mensagem do conjunto de dados também será referenciada por sentença, mesmo que, gramaticamente, a mensagem seja composta de várias frases, de modo a generalizar a explicação para qualquer tipo de corpo textual.

No caso de várias sentenças sendo processadas concomitantemente, os três vetores descritos acima tornam-se em três matrizes, as quais são alimentadas ao modelo do tipo BERT. Como, para a tarefa de estudo, utiliza-se o modelo "base" (desprovido de uma camada extra à saída da rede neural, a qual seria responsável por de fato realizar a tarefa específica de NLP), não existe uma saída ou um resultado explícito ao processamento das sentenças. Nesse caso, a abordagem adotada pelo trabalho que concebeu o BERT faz uso dos valores de ativação de uma ou mais camadas da rede neural para obter as representações vetoriais (DEVLIN; CHANG et al., 2019). Tais valores de ativação são armazenados em um tensor denominado *hidden_states* (do inglês, estados ocultos) composto por quatro dimensões (MCCORMICK, 2019), as quais são exploradas na sequência:

- 1ª dimensão: representa as camadas da rede neural, havendo uma posição adicional para a representação inicial do *token* (12+1 posições para o modelo Base, 24+1 posições para o modelo Large);

- 2ª dimensão: representa as sentenças que foram processadas, tendo número de posições igual ao número de sentenças no conjunto de dados processado;

- 3ª dimensão: representa os *tokens* da sentença. Devido ao processo de preenchimento descrito, a terceira dimensão tem, para todas as sentenças, número de posições igual ao maior número de *tokens* obtidos para cada sentença ou a um limite estabelecido pelo tokenizador;

- 4ª dimensão: representa o vetor do *token* (3ª dimensão) daquela sentença (2ª dimensão) naquela camada da rede neural (1ª dimensão), contendo 768 (modelo Base) ou 1024 (modelo Large) posições com números reais.

De todo exposto, percebe-se que não há um elemento nesse tensor referente a toda a sentença alimentada ao modelo. Dessa forma, deve-se definir uma operação sobre ele tal que o resultado seja a representação vetorial almejada.

No trabalho que originou o BERT (DEVLIN; CHANG et al., 2019), é descrito um problema que também deve fazer uso desse tensor, mas em uma tarefa supervisionada de NLP a nível de palavras. Nela, seis diferentes abordagens foram adotadas para se obter a representação vetorial dos *tokens*: utilizar apenas a última ou a penúltima camada da rede, somar as quatro últimas ou todas as camadas, concatenar as quatro últimas camadas e utilizar a primeira camada. Para aquela tarefa, todas as abordagens mencionadas apresentaram ótimos resultados, com medidas de F1 *score* entre 91,0% e 96,1%.

Já no caso da tarefa abordada por este trabalho a nível de sentenças, além da escolha da camada da rede neural a ser utilizada (ou das camadas), deve-se, ainda, reduzir a representação dos *tokens* a uma representação resultante da sentença. No entanto, não há uma metodologia definida e clara para realizar tal operação. Vale notar que também não há como avaliar, de maneira direta, se os vetores obtidos às sentenças são corretos.

Assim, a representação vetorial da sentença foi definida como a média entre todos os vetores, da última camada da rede, relativos aos *tokens* significativos que compõem a sentença, ou seja, desconsiderando-se os vetores relativos aos *tokens* de preenchimento ([PAD]). Dessa forma, o significado, atrelado ao contexto, de todas as sub-palavras da sentença contribuem à construção de um vetor resultante, o qual, espera-se, carrega o sentido da sentença. No entanto, a determinação da melhor estratégia pode variar dependendo da finalidade dada aos vetores (MCCORMICK, 2019).

Há referências, contudo, que argumentam que tal abordagem não resulta em boas representações vetoriais às sentenças, de modo que novas técnicas baseadas no BERT, como o Sentence-BERT (SBERT, traduzido do inglês como BERT para Sentenças), foram desenvolvidas (REIMERS; GUREVYCH, 2019), sendo reportados resultados melhores que aqueles obtidos utilizando-se o BERT em tarefas supervisionadas a nível de sentença. Todavia, assim como o modelo BERT original, não é disponibilizado um modelo SBERT específico à língua portuguesa. Assim, para utilizar tal modelo, é necessário ou realizar o treinamento de um novo modelo ao idioma de interesse ou utilizar um modelo multilíngue. Para este trabalho, a primeira opção é inviável devido ao tempo exigido ao seu desenvolvimento (coleta de textos, tratamento dos dados, treinamento do modelo, avaliação dos seus resultados e refinamento). Já a segunda é simples de ser implementada, pois

os modelos SBERT estão disponíveis no repositório de modelos da Hugging Face e também podem ser utilizados a partir da biblioteca Sentence Transformers, em Python. No entanto, o modelo multilíngue é treinado em mais de 50 línguas distintas, de modo que o entendimento de linguagem do modelo é compartilhado entre um conjunto de idiomas não necessariamente afins. Dessa forma, descartou-se a utilização do Sentence-BERT.

Uma das dificuldades deste trabalho reside na natureza não supervisionada da tarefa de agrupamento. Em um cenário de classificação, por exemplo, seria possível obter várias combinações de representações vetoriais (de diferentes modelos de linguagem) e de algoritmos de classificação, escolhendo-se o melhor arranjo pela avaliação das métricas apropriadas, como acurácia, sensibilidade e precisão. Todavia, não é possível fazer o mesmo com o conjunto de dados não anotados sendo explorado. Assim, o desenvolvimento do trabalho seguiu com a utilização do modelo BERTimbau, pois é um modelo específico à língua portuguesa brasileira, com arquitetura de tamanho Base, pois exige menor gasto computacional, produzindo representações vetoriais das sentenças a partir da média entre os vetores, da última camada da rede neural, relativos aos *tokens* significativos.

3.3 Métrica de Similaridade

Dado que as sentenças foram convertidas a vetores de números reais, é necessário definir uma métrica de similaridade entre os vetores, a qual deve refletir a similaridade entre o sentido das sentenças. Assim, espera-se que sentenças similares gerem vetores próximos no espaço vetorial, enquanto sentenças diferentes gerem vetores afastados. Dessa forma, a métrica deve ser capaz de converter tal interpretação a uma representação numérica.

Uma forma intuitiva de se definir tal similaridade entre dois vetores u e v é a partir da medida de distância entre eles, tal como pela distância euclidiana, cujo cálculo, apresentado em (1) resulta em valores pertencentes ao intervalo $[0, +\infty)$. Assim, vetores próximos apresentam distância próxima a zero, a qual cresce conforme os vetores se afastam. Como não há um limite superior, percebe-se que diferentes conjuntos de dados podem apresentar diferentes valores máximos à distância entre vetores muito afastados (ou seja, entre sentenças muito diferentes). Tal característica dificulta a interpretação dos valores obtidos e a definição de uma faixa de valores que represente sentenças similares ou diferentes.

$$d_e(u,v) = \sqrt{\sum_{i=1}^{n}(u_i - v_i)^2} \tag{1}$$

É possível também obter medidas de similaridade, das quais uma das mais comuns é a similaridade por cosseno. Calculada conforme (2), ela considera apenas a orientação dos vetores, descartando a sua magnitude. Dessa equação, facilmente explica-se o seu nome, pois a medida corresponde ao cálculo do cosseno do ângulo entre os vetores u e v. Como seus valores estão limitados ao intervalo $[-1, 1]$, o resultado da operação apresenta interpretação direta, independentemente das características do conjunto de dados empregado. Logo, similaridades em direção a 1 representam vetores próximos (sentenças similares) e similaridades em direção a -1 representam vetores afastados (sentenças diferentes). Ademais, essa é a métrica utilizada pela biblioteca Sentence Transformers no cálculo da similaridade entre sentenças (REIMERS; GUREVYCH, 2019). Assim, definiu-se o uso da similaridade por cosseno neste trabalho como a métrica de similaridade empregada.

$$sim\_cos(u,v) = \frac{u \cdot v}{\|u\| \, \|v\|} \tag{2}$$

Alguns algoritmos de aprendizagem de máquina utilizam, contudo, medidas de distância para realizar o agrupamento dos elementos. Embora não seja formalmente uma métrica de distância por não satisfazer a desigualdade triangular (VALLERIAN, 2021), é possível definir a distância por cosseno tal como apresentada em (3).

$$d_{cos}(u,v) = 1 - sim\_cos(u,v) = 1 - \frac{u \cdot v}{\|u\| \, \|v\|} \tag{3}$$

As diferentes arquiteturas dos modelos do tipo BERT geram representações vetoriais de diferentes dimensionalidades (768 para o Base, 1024 para o Large). Assim, nota-se que o aumento do número de elementos nos vetores gerados pela arquitetura Large em comparação à Base tende a aumentar os valores de distância euclidiana entre os vetores, enquanto a similaridade por cosseno é mantida dentro do intervalo $[-1,1]$. Logo, a troca de tamanho do modelo altera a interpretação dos resultados obtidos pela distância euclidiana, enquanto aquela obtida pela distância por cosseno é mantida.

Cenário similar é observado caso seja empregado um método de redução de dimensionalidade aos vetores, o qual tem o efeito oposto de reduzir a distância euclidiana entre eles. Seu uso pode ser interessante para economizar espaço de armazenamento das representações vetoriais e recursos computacionais na execução do algoritmo de agrupamento das mensagens. Novamente, percebe-se que a similaridade por cosseno mantém-se dentro do mesmo intervalo fechado. Assim, seu uso é incentivado por resultar numa interpretação mais clara, constante e intuitiva.

3.4 Análise Exploratória dos Dados

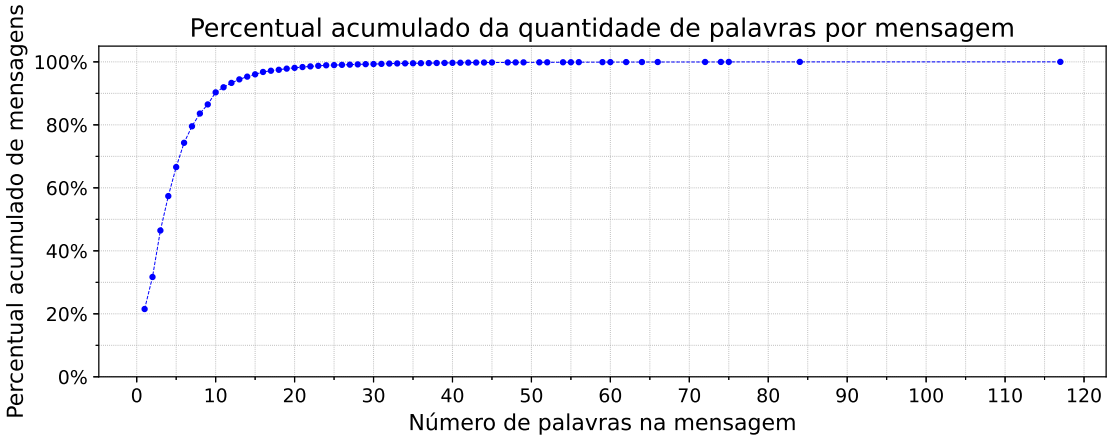
Com a definição do modelo de linguagem e da métrica de similaridade, realizou-se uma análise exploratória do conjunto de dados utilizado para melhor conhecê-lo. Assim, na Seção 3.4.1 são observados os comprimentos das mensagens e filtros são aplicados na remoção daquelas muito curtas ou muito longas. Na sequência, a Seção 3.4.2 apresenta como alguns parâmetros referentes ao tokenizador associado ao modelo de linguagem influenciam no resultado obtido à segmentação das sentenças. Por fim, a definição da métrica de similaridade empregada leva à análise de similaridades entre as sentenças do conjunto de dados exposta na Seção 3.4.3.

3.4.1 Quantidade de Palavras

O conjunto de dados utilizado no estudo de caso é composto por 10 mil sentenças, as quais possuem comprimento maior que 10 caracteres cada. A obtenção desses elementos aplicou tal condição de modo a descartar mensagens muito curtas por serem desprovidas de valor à tarefa de agrupamento, sendo geralmente utilizadas no interior do fluxo de conversação em respostas objetivas, tais como: “Sim”, “Não”, “Boleto” e “Débito”.

Na Figura 1 é possível observar a distribuição acumulada da quantidade de palavras por mensagem no conjunto de dados. Nesse contexto, uma palavra é qualquer sequência de caracteres delimitada por espaço em branco. Do gráfico, percebe-se que textos curtos (na ordem de uma dezena de palavras) são predominantes, sendo condizentes ao meio de comunicação empregado: um aplicativo de mensagens instantâneas. Nota-se que mensagens muito longas devem ser mais difíceis de serem agrupadas, pois, além de não corresponderem ao modo como a maioria dos usuários escreve, são compostas por longas explicações ou por mais de um assunto. Assim, removeram-se as mensagens com mais de 20 palavras, as quais representam apenas 1,90% do conjunto de dados original.

Figura 1: *Percentual acumulado da quantidade de palavras por mensagem no conjunto de dados.*



Fonte: **O Autor**

Mensagens muito curtas como “Financiamento” e “Cartão de crédito”, no entanto, também não são relevantes ao presente estudo, pois elas não possuem valor semântico significativo, ou seja, não é possível determinar o assunto relativo à mensagem. Dessa forma, é impossível saber se o cliente quer, por exemplo, “contratar um financiamento” ou “saber as taxas aplicadas ao serviço”, “pedir uma segunda via do cartão de crédito” ou “informar que compras indevidas estão sendo feitas com ele”. Ainda, tais mensagens curtas apresentam-se em grande quantidade no conjunto de dados, dificultando a identificação de bons e específicos exemplos que devem ser utilizados no treinamento do *chatbot* cognitivo no entendimento das intenções. Contudo, se as mensagens forem compostas por verbos, como em “Aumentar limite”, é possível identificar o assunto genérico daquela mensagem. Logo, embora sentenças tão curtas não sejam bons exemplos a serem utilizados na composição de intenções, elas podem colaborar, no processo de agrupamento das mensagens, na identificação de assuntos genéricos que estão sendo requisitados pelos usuários. Assim, dado esse compromisso, optou-se pela remoção das mensagens com menos de 3 palavras, as quais representam 31,69% do conjunto de dados original.

A aplicação de ambas as condições, quanto aos números mínimo e máximo de palavras que devem compor as sentenças, reduz as 10 mil mensagens para 6.641. Tal metodologia de filtragem podia ter consultado quanto à presença de verbos nas mensagens. No entanto, é comum que erros ortográficos sejam encontrados, como “*almentar*” no lugar de “aumentar”, dificultando a análise. Tais exemplos são importantes, pois é necessário que o modelo do *chatbot* conheça a forma de comunicação dos usuários.

Ademais, deve-se observar quanto à repetição das sentenças no conjunto de dados. Mensagens repetidas, compostas por muitas palavras, geralmente são enviadas pelo mesmo usuário que, ao perceber que o *chatbot* não entendeu o seu questionamento, tenta novamente. Já quando as mensagens são compostas por poucas palavras, elas ocorrem em grande quantidade, novamente dificultando a identificação de bons exemplos para a construção das intenções (por exemplo, no conjunto de dados, “Depósito de cheque” ocorre 33 vezes, “Cartão de crédito”, 27). Por conseguinte, optou-se pela remoção das duplicatas, reduzindo-se o conjunto de dados a 5.259 mensagens.

3.4.2 Características do Tokenizador

O processo de segmentação dos textos pelo tokenizador do BERTimbau pode sofrer mudanças de acordo com algumas configurações definidas na sua instanciação. Na sequência, dois parâmetros são explorados.

É comum que textos digitados em aplicativos de mensagens não sigam o padrão da norma culta quanto à capitalização das palavras, de modo que pode haver mistura desordenada entre letras maiúsculas e minúsculas. Ainda, espera-se que poucas letras sejam maiúsculas, como no início de frases, em nomes próprios ou no nome de algum produto específico. No entanto, elas não são importantes à identificação do sentido da sentença. Ademais, nota-se que a frequência de letras minúsculas é maior que a de maiúsculas na língua portuguesa, de modo que o tokenizador é capaz de obter representações com menor número de *tokens* quando a grafia da palavra apresenta apenas letras minúsculas, retendo-se mais informação à lista de *tokens* resultante. Tal comportamento pode ser observado na Tabela 1 para a palava "trabalhar".

Tabela 1: *Tokenização das palavras "trabalhar" e "verificação" para diferentes capitalizações.*

Palavra	Tokenização	Número de *tokens*
trabalhar	trabalhar	1
Trabalhar	Trabalh-ar	2
TRABALHAR	T-RA-BA-L-HA-R	6
verificação	verifica-ção	2
Verificação	Ver-ificação	2

Fonte: **O Autor**

Verificou-se também que grafias em letras minúsculas apresentam melhor segmentação quando diferentes sub-palavras podem ser identificadas. A palavra "verificação", por exemplo, possui as sub-palavras "ver" e "verifica". Embora não exista a definição de uma tokenização correta, é desejável que a sub-palavra obtida pelo tokenizador seja "verifica", pois ela apresenta sentido mais próximo da palavra original. Tal cenário ocorre, conforme a Tabela 1, quando todas as letras são minúsculas.

Dessa forma, configurou-se que o tokenizador deve remover a capitalização de todas as letras que compõem a sentença sendo segmentada, as quais são convertidas a minúsculas. Tal procedimento pode fazer com que sentenças, antes distintas pela capitalização, tornem-se iguais. Assim, após a conversão, uma nova remoção de duplicatas foi efetuada, reduzindo as 5.259 sentenças obtidas ao final da Seção 3.4.1 a 5.170.

Outra configuração possível é referente à remoção dos sinais diacríticos: acentos, til, trema e cedilha (MORENO, 2022). Tais sinais gráficos são importantes na definição das palavras em língua portuguesa, tanto na sua pronúncia quanto no seu significado. Embora seja comum a sua supressão na digitação de mensagens de texto, não há ganhos na sua remoção por parte do tokenizador. Na Tabela 2, é possível perceber como a falta dos sinais diacríticos deteriora a qualidade da segmentação obtida devido ao incremento no número de *tokens* obtidos. Assim, o tokenizador foi configurado de modo a não remover sinais diacríticos.

Tabela 2: *Tokenização das palavras "automação" e "rápido" com e sem sinais diacríticos.*

Palavra	Tokenização	Número de *tokens*
automação	automa-ção	2
automacao	automa-ca-o	3
rápido	rápido	1
rapido	rap-ido	2

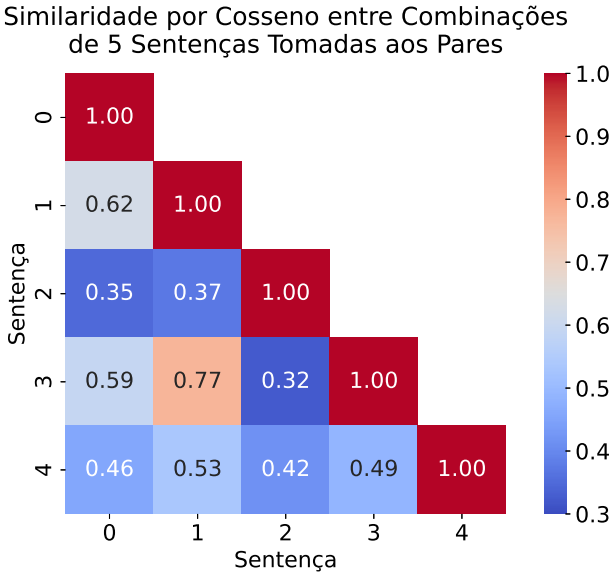
Fonte: **O Autor**

3.4.3 Similaridades por Cosseno

Com a tokenização das sentenças e a obtenção das três matrizes descritas na Seção 3.2, é possível alimentar o modelo BERT com tal informação para se obter a representação vetorial das sentenças do conjunto de dados. Como o algoritmo de agrupamento deve criar grupos com base na similaridade entre as sentenças, estudou-se a distribuição de similaridades por cosseno dentro do conjunto. Para tanto, obteve-se a similaridade para todas as combinações de todos os 5.170 elementos tomados de dois em dois.

A estrutura do resultado obtido, simplificada para apenas 5 elementos (enumerados de 0 a 4), pode ser observada na Figura 2. Como a ordem dos elementos na Equação 2 não afeta o resultado, a matriz de similaridades é simétrica. Assim, apenas um dos triângulos da matriz (inferior ou superior) precisa ser calculado. Nota-se também que não é necessário calcular a diagonal principal, cujos elementos são sempre 1, pois a similaridade por cosseno entre dois vetores idênticos resulta em 1.

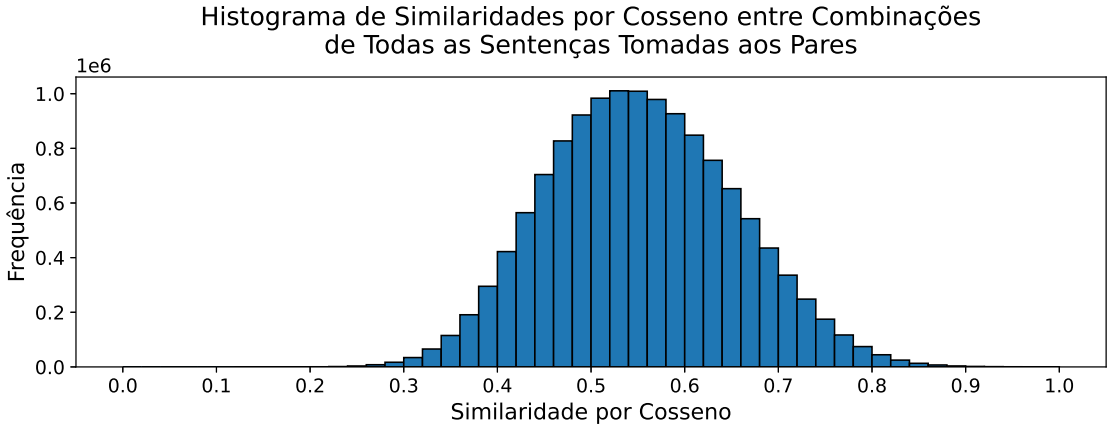
Figura 2: *Similaridade por cosseno entre combinações de 5 sentenças tomadas aos pares.*



Fonte: **O Autor**

De posse da matriz calculada para todas as 5.170 sentenças, construiu-se o histograma apresentado na Figura 3 para se entender a distribuição de frequências das similaridades por cosseno entre todos os elementos do conjunto de dados. Devido às características da matriz, removeram-se os valores da diagonal principal, pois a similaridade do elemento com si próprio não traz informação relevante, e o triângulo superior, pois, sendo a matriz completa simétrica, ele apenas acrescenta um fator de escala 2 ao resultado do histograma. Embora a Equação 2 permita resultados dentro do intervalo [-1, 1], não foram encontrados valores negativos.

Figura 3: *Histograma de similaridades por cosseno entre combinações de todas as sentenças tomadas aos pares.*

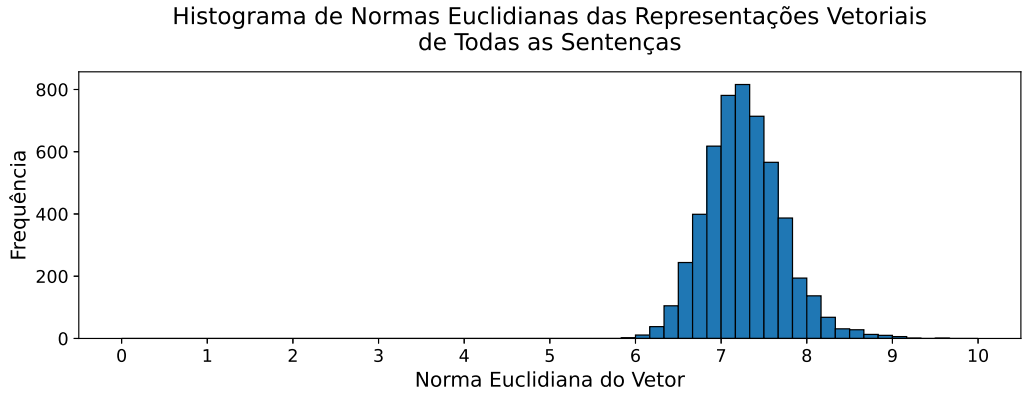


Fonte: **O Autor**

O propósito do histograma está na avaliação dos agrupamentos de mensagens a serem gerados. Assim, espera-se que a similaridade média entre os elementos de um mesmo grupo seja superior a 0,55, pois, conforme a Figura 3 essa é, aproximadamente, a similaridade média entre todos os elementos do conjunto de dados. Evidentemente, um valor maior de similaridade média pode ser usado na avaliação do agrupamento. Ademais, outros valores também podem ser definidos, tal como a similaridade mínima entre dois elementos quaisquer do agrupamento.

A partir da representação vetorial das mensagens, é possível analisar a contribuição das magnitudes (normas euclidianas) dos vetores no cálculo de distâncias. A distância euclidiana (Equação 1) considera tanto a magnitude quanto a orientação dos vetores, enquanto a distância por cosseno (Equação 3) considera apenas a orientação. Do histograma apresentado na Figura 4, é possível observar que a variação de magnitudes dos vetores é pequena: o primeiro quartil (6,97) está localizado 4,12% abaixo da média (7,27), já o terceiro quartil (7,53) está localizado 3,65% acima da média. Por conseguinte, a parcela de contribuição da magnitude dos vetores no cálculo da distância entre eles também é pequena, pois todos apresentam, aproximadamente, a mesma norma. Logo, não há perda significativa de informação no uso da distância por cosseno em detrimento da distância euclidiana.

Figura 4: *Histograma de normas euclidianas das representações vetoriais de todas as sentenças.*



Fonte: **O Autor**

3.5 Agrupamento das Mensagens

As seções a seguir apresentam como foi realizado o agrupamento das mensagens presentes no conjunto de dados após obtidas as suas representações vetoriais. De tal forma, a Seção 3.5.1 descreve o algoritmo empregado para realizar o agrupamento, enquanto a Seção 3.5.2 trata da filtragem dos agrupamentos obtidos, de modo a escolher aqueles relevantes à análise sendo empreendida.

3.5.1 Método de Agrupamento

Para realizar o agrupamento das sentenças (das mensagens do conjunto de dados), utilizou-se um método de aprendizagem de máquina não supervisionado, pois não sabe-se, a priori, quais são os grupos aos quais os dados devem ser rotulados. Dos algoritmos comumente empregados, geralmente, ao menos um dos seguintes parâmetros deve ser informado ao modelo para que o algoritmo de agrupamento seja executado: a quantidade de grupos a serem obtidos, a quantidade de elementos (mínima ou máxima) que cada grupo deve ter e um valor limiar de distância (que delimita se um elemento pertence ou não a um agrupamento).

Nenhum desses valores, no entanto, são conhecidos. Seria possível estabelecê-los definindo-se alguma relação baseada no conhecimento do número de elementos no conjunto de dados e/ou nas informações apresentadas no histograma da Figura 3. Contudo, seriam definições arbitrárias. Ademais, a performance de alguns métodos depende da estrutura apresentada pelos dados, a qual não é possível, de maneira simples e intuitiva, de ser visualizada para vetores em $\mathbb{R}^{768}$.

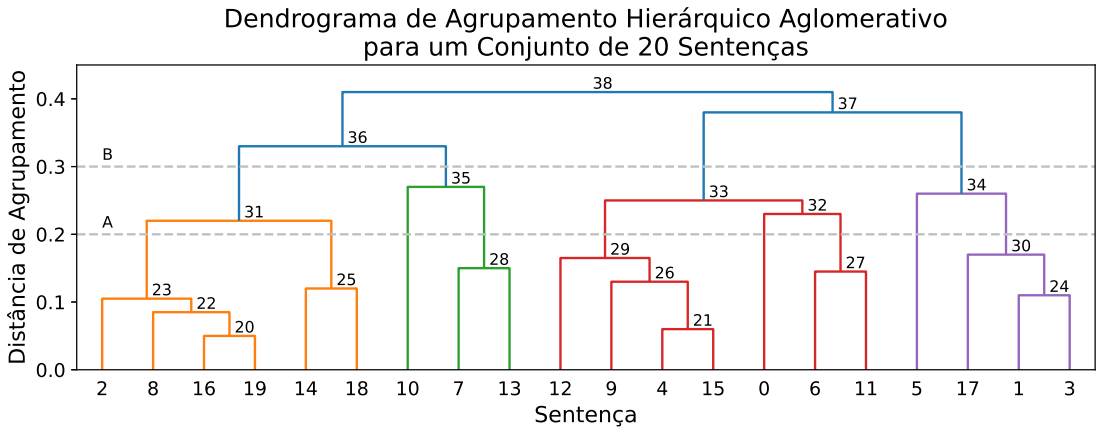
De tal forma, buscou-se um método de agrupamento que refletisse a estrutura de tópicos das mensagens dos usuários. Evidentemente, há mensagens quase idênticas umas das outras (variando-se a presença de algumas palavras conectivas, por exemplo), as quais apresentam valores quase unitários de similaridade por cosseno e, portanto, devem pertencer a um mesmo agrupamento. Na sequência, mensagens sobre um mesmo tópico, com maior heterogeneidade na sua sintaxe devem ser agrupadas. Assim, os grupos anteriores são reagrupados, agora dentro de um conjunto maior, com menor similaridade interna em relação ao original, mas ainda apresentando semelhança nos assuntos.

Um algoritmo de aprendizagem de máquina não supervisionado que realiza tal tipo de tarefa é o agrupamento hierárquico, no qual, ao final do seu processamento, a hierarquia de grupos é representada por meio de uma estrutura similar a uma árvore: o dendrograma. Esse algoritmo, dependendo da sua lógica de execução, pode ser classificado como aglomerativo ou divisivo (PERES; LIMA, 2015). O método aglomerativo segue uma abordagem *bottom-up* (do inglês, de baixo para cima), na qual cada elemento é inicializado como um grupo próprio (um grupo inicial de tamanho unitário). A cada iteração do algoritmo, o agrupamento é realizado tal que os dois grupos com menor distância são agrupados em um novo grupo até restar um único agrupamento contendo todos os elementos (a definição de distância entre dois grupos é apresentada posteriormente nesta seção). Já o método divisivo segue uma abordagem *top-down* (do inglês, de cima para baixo), na qual todos os elementos são inicializados em um único grupo, o qual é sucessivamente dividido em grupos menores até que todos os elementos estejam em grupos unitários.

Embora a estrutura final resultante de ambos os métodos seja a mesma, um dendrograma, percebe-se que a abordagem seguida pelo método aglomerativo é a que melhor se encaixa ao problema em questão, o qual é então utilizado na sequência deste trabalho. Não é de interesse analisar como o conjunto inicial se divide, mas sim como as mensagens individuais são agrupadas pela semelhança entre seus conteúdos, até que o agrupamento atinja um valor mínimo de similaridade (ou máximo de distância) além do qual novas aglomerações degradam a coesão do conjunto em torno de um determinado assunto.

Na Figura 5, é apresentado um exemplo de dendrograma obtido para um agrupamento hierárquico aglomerativo. A interpretação dos elementos que compõem o diagrama é dada na sequência.

Figura 5: *Dendrograma de agrupamento hierárquico aglomerativo para um conjunto de 20 sentenças.*



Fonte: **O Autor**

No eixo horizontal do dendrograma da Figura 5 estão demarcadas as 20 sentenças utilizadas pelo algoritmo (as folhas, enumeradas de 0 a 19), as quais são interpretadas como agrupamentos iniciais de tamanho unitário. As linhas verticais no interior do dendrograma (os ramos, enumerados de 0 a 37) representam os grupos gerados (com exceção do grupo 38, que não apresenta linha vertical). Desconsiderando-se os grupos unitários, a linha horizontal localizada na ponta inferior dos ramos 20 a 37, formando o símbolo $\perp$, representa a aglomeração dos grupos, que ocorre sempre aos pares. No eixo vertical, é

demarcada a distância de agrupamento para aquela aglomeração. O último grupo gerado (a raiz, de número 38) engloba todos os elementos e está localizado na parte superior do dendrograma. Vale notar que o número associado ao agrupamento tanto o identifica quanto informa a ordem de agrupamento, seguindo a ordem crescente do 20 ao 38.

Na primeira iteração do algoritmo, quando todos os grupos são compostos de apenas um elemento, basta medir a distância das combinações de todos os elementos, tomados de dois em dois, e unificar aqueles dois que apresentarem menor valor. Nesse cenário, a distância entre dois grupos é, simplesmente, igual à distância entre os elementos individuais, calculada pela Equação 3. Contudo, para as próximas iterações, é necessário estabelecer como medir a distância entre agrupamentos com mais de um elemento. Para tanto, diferentes medidas podem ser utilizadas, sendo as mais comuns apresentadas na sequência, juntamente das suas definições de distância entre dois grupos (KASSAMBARA, 2022):

- Ligação única (em inglês, *single linkage*): mede a distância entre o par de elementos mais próximos de cada grupo — é a distância mínima entre os grupos;
- Ligação completa (em inglês, *complete linkage*): mede a distância entre o par de elementos mais afastados de cada grupo — é a distância máxima entre os grupos;
- Ligação média (em inglês, *average linkage*): mede a média de distâncias entre todos os pontos pertencentes a cada grupo;
- Método do centroide (em inglês, *centroid method*): mede a distância entre o centroide de cada grupo, o qual é obtido da média dos vetores que o compõem;
- Método de Ward (em inglês, *Ward's method*): mede o aumento na variância causado pela aglomeração de dois grupos, o qual é dado pela diferença entre a variância do agrupamento resultante e a soma das variâncias dos agrupamentos originais.

Escolheu-se a ligação completa como medida de distância entre dois agrupamentos porque ela tende a gerar grupos compactos. Assim, aglomerando-se os dois grupos que apresentam a menor distância máxima, garante-se que os elementos daquele novo conjunto que foram agrupados são, no máximo, tão afastados (tão distintos) quanto a distância de agrupamento, ou seja, a distância entre os elementos é sempre menor que ou igual à distância de agrupamento. Ademais, também garante-se que as distâncias de agrupamento e distâncias entre os elementos de todos os agrupamentos realizados anteriormente são iguais a ou menores que a distância de agrupamento daquela aglomeração, pois o agrupamento é realizado em ordem crescente de distância entre os grupos.

Como exemplo, seja a distância de agrupamento, entre dois grupos que foram aglomerados, igual a 0,2. Devido à definição da ligação completa, essa é a distância máxima entre todos os pares de elementos pertencentes ao novo agrupamento. Assim, pela Equação 3, é possível definir que a similaridade mínima entre todos os pares de elementos pertencentes ao novo agrupamento é igual a 0,8. À média de similaridades de todas as combinações de elementos pertencentes ao novo agrupamento, tomados de dois em dois, definiu-se uma nova propriedade, denominada de similaridade interna. Evidentemente, nesse cenário, seu valor será maior que ou igual a 0,8. Além disso, devido à ordem seguida pelo método, todos os outros agrupamentos realizados anteriormente também apresentam os limites de distância e de similaridade do agrupamento sendo observado.

3.5.2 Filtragem dos Agrupamentos

A construção do dendrograma, no entanto, não finaliza a resolução do problema. Com exceção das folhas e da raiz, todos os grupos tanto fazem parte de um supergrupo maior como são compostos de dois subgrupos menores. Portanto, é necessário definir a aplicação de um filtro sobre um conjunto de características dos agrupamentos para que apenas alguns deles sejam observados. Tal conjunto foi definido por:

- Distância interna do agrupamento: representa a distância (calculada pela ligação completa) entre o par de grupos aglomerados que formam aquele agrupamento. Também pode ser interpretada como o afastamento máximo entre os vetores do agrupamento. Quanto maior for a distância interna, mais desiguais são os grupos que foram aglomerados, de modo que o filtro deve estabelecer um valor máximo à distância interna;

- Similaridade interna do agrupamento: representa a média de similaridades de todas as combinações, tomadas de dois em dois, de sentenças daquele agrupamento. Quanto menor for a similaridade interna, menor é a semelhança entre os elementos que compõem o agrupamento, de modo que o filtro deve estabelecer um valor mínimo à similaridade interna;

- Tamanho do agrupamento: representa a quantidade de elementos (número de sentenças) que compõem aquele agrupamento. Como é de interesse observar assuntos recorrentes entre os usuários, o filtro deve estabelecer um valor mínimo ao tamanho do agrupamento.

Vale notar que o filtro, embora possa, não precisa utilizar todas as características anteriormente listadas, sendo minimamente necessário que a distância interna ou a similaridade interna estejam definidas. Embora os parâmetros estejam de certa maneira relacionados, pois conforme se avança na árvore em direção à raiz, os tamanhos dos agrupamentos aumentam, as distâncias internas aumentam e as similaridades internas diminuem, não há uma relação direta e fixa que os interliguem. Um agrupamento, por exemplo, com vários elementos muito próximos terá uma alta similaridade interna e uma baixa distância interna. A inserção ao grupo de um elemento mais afastado aumenta a distância interna, mas pode não interferir significativamente na similaridade interna, pois ela é a média de todas as similaridades dos pares de elementos. Logo, não é possível afirmar que a distância interna e a similaridade interna estão relacionadas pela Equação 3, pois ela relaciona a distância interna e a similaridade mínima entre os pares de elementos daquele grupo.

Contudo, apenas aplicar o filtro sobre o conjunto de agrupamentos não é suficiente. Como exemplo, seja escolhido, para o dendrograma da Figura 5, que a distância interna máxima deve ser igual a 0,2. Logo, todos os agrupamentos localizados abaixo da linha tracejada A satisfazem a condição imposta. No entanto, não é de interesse que sejam apresentados todos os grupos e subgrupos. Dado que o grupo 23 satisfaz o filtro, não deve-se listar juntamente a ele os grupos 22 e 20 (e possivelmente os grupos unitários 2, 8, 16 e 19) e os elementos que os compõem, tanto por redundância da informação apresentada quanto por atrapalhar a visualização do resultado final pelo incremento considerável de grupos e elementos exibidos.

Assim, a linha tracejada A deve ser interpretada como uma linha de corte superior, tal que o resultado da filtragem seja composto apenas dos ramos (dos grupos) cruzados por ela, ou seja, das aglomerações (linhas horizontais) realizadas logo abaixo dela. Nesse

cenário, 9 ramos são interceptados, de modo que o resultado do agrupamento hierárquico aglomerativo do conjunto de 20 sentenças, após aplicado o filtro, é dado, portanto, pelos 9 agrupamentos a seguir: 23, 25, 10, 28, 29, 0, 27, 5 e 30. Dessa forma, garante-se que cada sentença aparece uma única vez em um único grupo.

Dessa definição, na qual apenas os supergrupos devem ser selecionados, fica evidente porque não é possível estabelecer apenas o tamanho mínimo do agrupamento, pois a aplicação dessa filtragem implicaria a seleção do agrupamento raiz. Seria possível também adicionar ao filtro um valor máximo ao tamanho do agrupamento. Contudo, esse novo parâmetro não é relevante a quem for aplicar a filtragem, pois a determinação da distância interna e/ou da similaridade interna já delimitam o dendrograma pelos ramos superiores.

Já para uma nova filtragem, definida para uma distância interna igual a 0,3 (linha tracejada B na Figura 5), 4 ramos são interceptados. Assim, o resultado do algoritmo de agrupamento, para esse cenário, é dado pelos seguintes grupos: 31, 35, 33 e 34. Tal como exposto anteriormente, todos esses agrupamentos foram aglomerados por uma distância menor que 0,3.

Dessas duas filtragens nota-se que quanto maior for a distância interna definida para linha de corte do filtro, menor será o número de agrupamentos obtidos, os quais, por consequência, apresentam, em média, maior número de elementos cada. De maneira paralela, a heterogeneidade entre os elementos dentro do grupo aumenta, diminuindo a similaridade interna, pois com o aumento da distância permitida ao agrupamento, menor a semelhança entre os elementos que o compõem.

Ademais, percebe-se que para ambos os cenários há desuniformidade no tamanho dos agrupamentos gerados. Para a linha de corte A, por exemplo, os agrupamentos 23 e 29 contêm 4 elementos cada, enquanto os agrupamentos 10, 0 e 5 possuem apenas 1. Tal comportamento já era esperado, pois naturalmente haverá várias mensagens no conjunto de dados que são muito similares, tratando sobre um assunto recorrente entre os usuário do *chatbot*, as quais geram grupos com muitos elementos e com baixa distância interna. Enquanto isso, outras mensagens, por tratarem sobre um assunto incomum ou mesmo por não fazerem sentido, são agrupadas tardiamente no dendrograma.

Como consequência, surge o questionamento sobre quais devem ser os valores definidos aos parâmetros do filtro, bem como quais deles devem ser utilizados no estabelecimento final dos agrupamentos encontrados pelo algoritmo. Contudo, não há uma definição clara e objetiva para tal. Dessa forma, permitiu-se, por meio da ferramenta de visualização desenvolvida (explorada na Seção 3.6), que o usuário tenha o poder de configurar o filtro à sua vontade, de acordo com os objetivos que ele deseja atingir na análise sendo empreendida.

3.6 Ferramenta de Visualização dos Agrupamentos

Com a execução do algoritmo de agrupamento das mensagens apresentado na Seção 3.5, já é possível extrair as informações necessárias para automatizar o processo de descoberta de novas intenções em um *chatbot* cognitivo. No entanto, para visualizar os agrupamentos, analisar as mensagens e suas relações de distâncias e similaridades e, efetivamente, tirar valor das informações geradas, até então, é necessário ter conhecimento de programação e de análise de dados para manipular e interpretar os resultados obtidos pelo processamento.

É importante ressaltar que times que trabalham no desenvolvimento de um *chatbot* cognitivo possuem membros diversos, com habilidades de diferentes áreas do conheci-

mento. Há equipes dedicadas: à experiência do usuário, para que o diálogo seja fluido e agrade o cliente; à manutenção do canal de comunicação entre cliente e *chatbot*; ao armazenamento, tratamento e análise dos dados gerados pelas interações, para gerar insumos à melhoria do sistema; às decisões no âmbito de negócio, para que a solução atinja os objetivos da organização; ao aperfeiçoamento do *chatbot*, criando novos fluxos de conversação e melhorando os já existentes; à orientação do time como um todo nas boas práticas de desenvolvimento de um sistema baseado em inteligência artificial.

Assim, desenvolveu-se uma ferramenta, como uma prova de conceito, a fim de tornar a informação acessível a membros da equipe com ou sem conhecimento técnico. Para tanto, uma aplicação *web* foi criada utilizando-se o Streamlit, um *framework open source* que permite o desenvolvimento utilizando-se apenas linguagem de programação Python (RAJAN, 2021), uma das mais utilizadas no campo de ciência de dados (HEBBAR, 2019). As três páginas que compõem a aplicação são exploradas nas seções seguintes.

A biblioteca abstrai muito do desenvolvimento tradicionalmente necessário na construção de aplicações para o *back end* e o *front end*. Com apenas um arquivo Python e utilizando-se a API disponibilizada pelo Streamlit, é possível criar uma página *web* com elementos interativos, os quais podem ser inseridos à página com apenas uma linha de código cada. Ademais, com foco ao desenvolvimento de aplicações voltadas a dados, a biblioteca é compatível com diversas das bibliotecas Python utilizadas para processamento e visualização de dados. Com essa simplicidade, em contrapartida, restringe-se a liberdade de personalização da página *web* e o controle de outros aspectos relativos seu desenvolvimento.

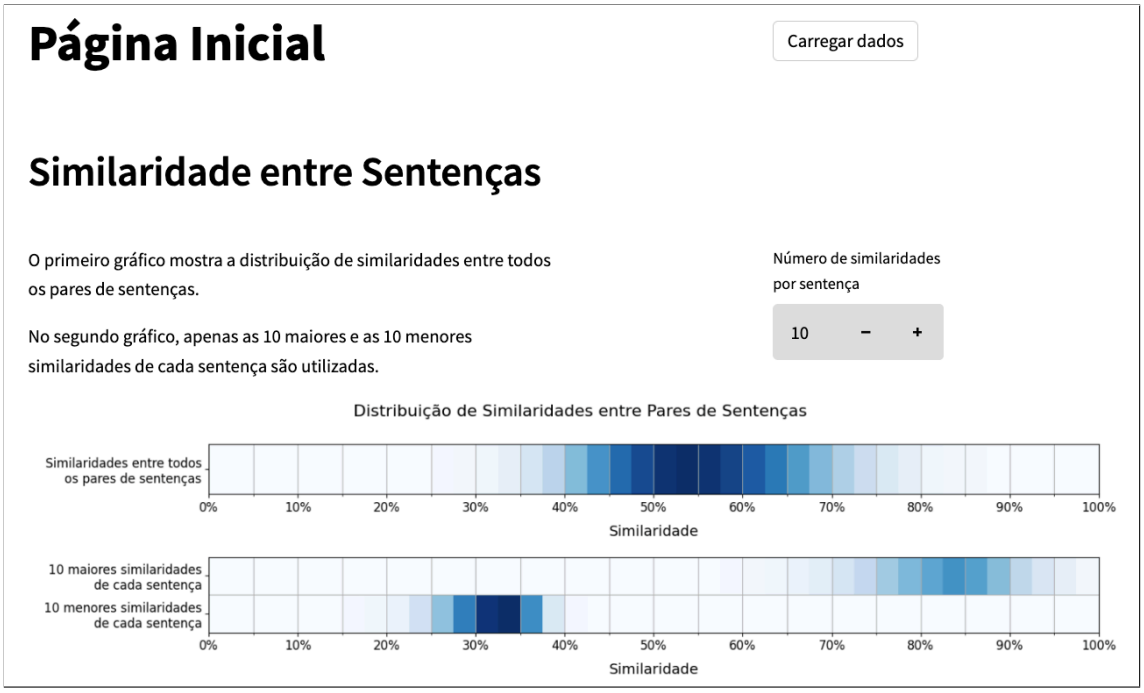
3.6.1 Página Inicial

Na página inicial, são expostos gráficos para que o usuário possa entender a distribuição de similaridades entre as sentenças do conjunto de dados e a distribuição de similaridades e de distâncias dos agrupamentos encontrados. Para tanto, uma versão simplificada do histograma da Figura 3 é exibida, na qual a altura das barras verticais é substituída pela intensidade da coloração azul no gráfico, conforme Figura 6.

Vale notar que as similaridades, até então tratadas como números inteiros pertencentes ao intervalo $[-1, 1]$, são representadas por uma notação de porcentagem para facilitar a leitura. Como não foram encontrados valores negativos, ela é tratada como um valor inteiro entre 0% e 100% (representando o intervalo $[0, 1]$). No entanto, de modo a manter o trabalho aplicável a um conjunto qualquer de mensagens, estipulou-se a aplicação de uma função para que valores negativos sejam substituídos por 0. Assim, embora não tenham ocorrido para o conjunto de dados utilizado, é importante observar na etapa de processamento a quantidade e a relevância desses elementos.

Na sequência, ainda na Figura 6, é apresentado um segundo gráfico contendo dois histogramas. O primeiro representa a distribuição de frequências dos valores de similaridade das n sentenças mais similares a cada uma das sentenças, enquanto o segundo utiliza as n sentenças menos similares. O valor de n, inicialmente definido em 10, pode ser alterado pelo usuário pelo campo de entrada numérica. Para melhor entender a construção desses histogramas, considere a matriz apresentada Figura 2 de maneira completa: com o triângulo superior preenchido de modo a ficar simétrica. Na sequência, os valores de similaridade de cada linha são ordenados de maneira crescente e a última coluna é removida, pois ela representa os elementos unitários da diagonal principal (similaridade da sentença com si mesma). Assim, o primeiro histograma é construído obtendo-se, para cada linha, os n últimos elementos (da última coluna em direção à primeira, da direita para a es-

Figura 6: *Distribuição de similaridades entre as sentenças.*



Fonte: **O Autor**

querda), enquanto o segundo histograma utiliza os n primeiros elementos de cada linha (da primeira coluna em direção à última, da esquerda para a direita).

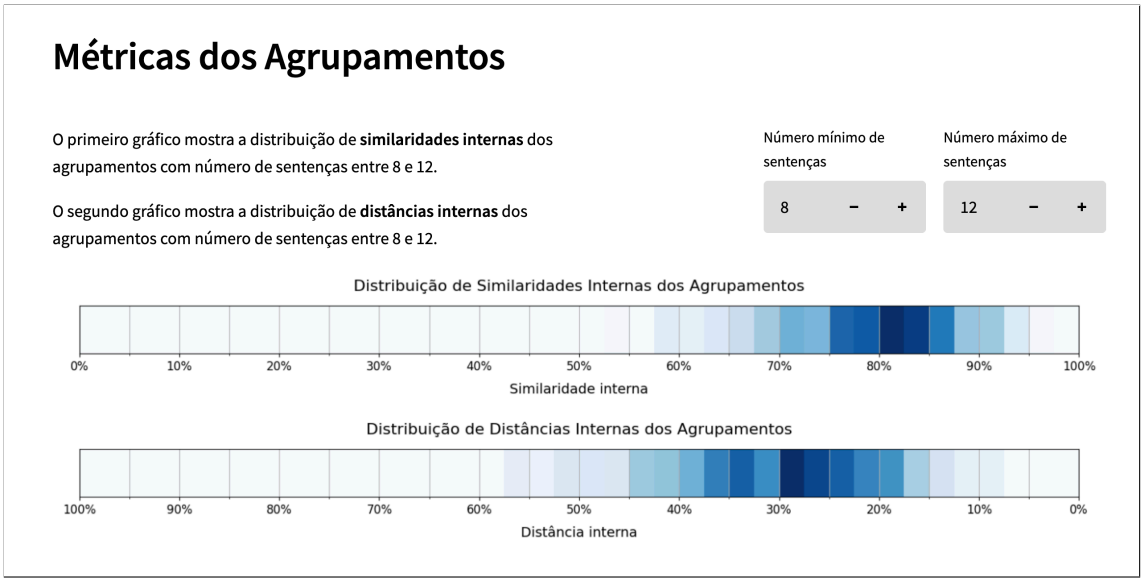
De tal forma, o primeiro histograma visa informar ao usuário qual é a similaridade ótima esperada a um agrupamento com n sentenças. O termo ótima, nesse caso, refere-se ao fato de que as escolhas das n similaridades de cada sentença foram feitas livremente, sem as restrições impostas pelo algoritmo de agrupamento. Assim, ao investigar os agrupamentos obtidos com número de elementos próximo a n, o valor de similaridades tende a ser menor. Embora não tão útil, o segundo histograma exibe os valores de similaridade mínimos, representando a dessemelhança entre as sentenças.

Já na Figura 7, são utilizadas as informações relativas aos agrupamentos para gerar dois histogramas. Neles, é possível observar, para um determinado intervalo de número de sentenças no agrupamento, a distribuição de frequências de similaridades internas e de distâncias internas dos agrupamentos (parâmetros importantes na definição do filtro a ser aplicado). Os gráficos utilizam um intervalo ao tamanho do grupo pois não há garantia de que agrupamentos sejam criados com um número específico de elementos.

Para permitir comparação com o histograma de cima, os valores de distâncias internas estão em ordem decrescente. Como a distância interna, dada pela ligação completa, representa a distância máxima entre os elementos que compõem o agrupamento, ela também pode ser interpretada, pela Equação 3, como a similaridade mínima entre os elementos do agrupamento. Assim, a observação dos dois histogramas permite conhecer a relação entre as similaridades internas (similaridades médias) e as similaridades mínimas dos agrupamentos.

Por fim, os gráficos da Figura 7 apresentam um contraponto aos gráficos da Figura 6. Enquanto a Figura 6 informa as similaridades ótimas, a Figura 7 apresenta as similaridades (e as distâncias) de fato encontradas pelo algoritmo de agrupamento.

Figura 7: *Distribuição de similaridades e de distâncias internas dos agrupamentos.*

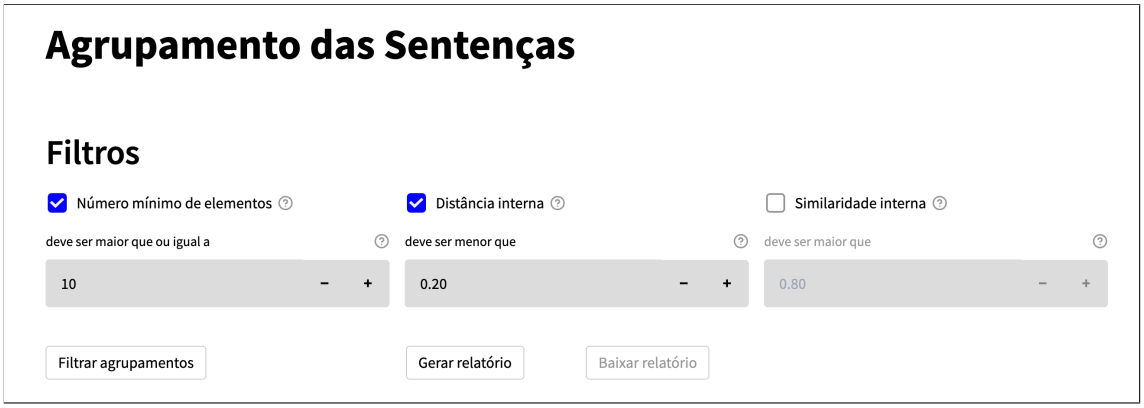


Fonte: **O Autor**

3.6.2 Página de Agrupamento das Sentenças

Na segunda página da aplicação *web*, os resultados gerados pelo algoritmo de agrupamento hierárquico aglomerativo são apresentados ao usuário. Para tanto, tal como visto na Seção 3.5.2, é necessário definir os parâmetros de filtragem dos agrupamentos, sendo tal processo realizado por meio da interface apresentada na Figura 8. As caixas de seleção definem quais dos três parâmetros devem ser utilizados na filtragem dos agrupamentos, enquanto os campos de entrada numérica localizados logo abaixo permitem que o usuário defina os valores a serem utilizados.

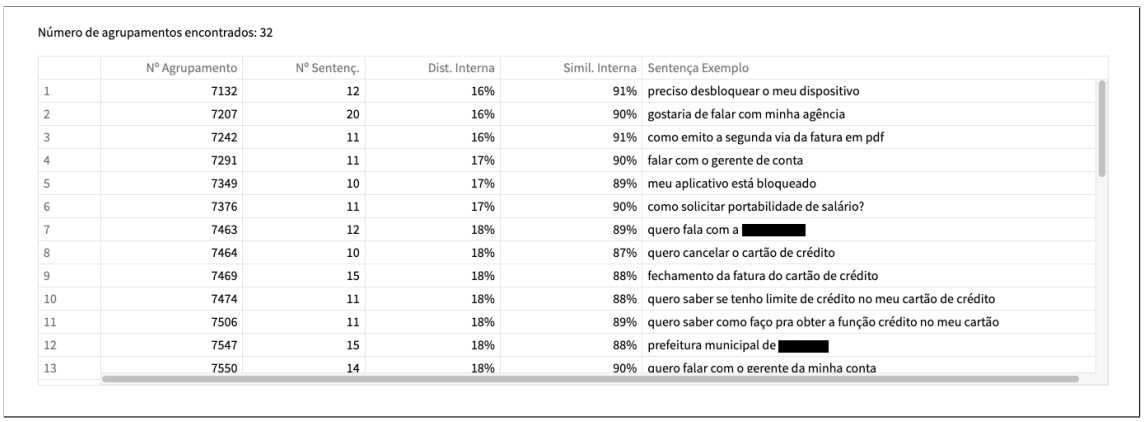
Figura 8: *Definição dos parâmetros de filtragem.*



Fonte: **O Autor**

Ao clicar no botão "*Filtrar agrupamentos*", são apresentados os agrupamentos obtidos da aplicação do filtro por meio de uma tabela, conforme Figura 9. Cada linha da tabela representa um agrupamento que passou pelo filtro, excluindo-se, como já elucidado, os subgrupos dos grupos que satisfazem as condições impostas.

Figura 9: *Agrupamentos resultantes da aplicação do filtro.*



Número de agrupamentos encontrados: 32

	Nº Agrupamento	Nº Sentenç.	Dist. Interna	Simil. Interna	Sentença Exemplo
1	7132	12	16%	91%	preciso desbloquear o meu dispositivo
2	7207	20	16%	90%	gostaria de falar com minha agência
3	7242	11	16%	91%	como emito a segunda via da fatura em pdf
4	7291	11	17%	90%	falar com o gerente de conta
5	7349	10	17%	89%	meu aplicativo está bloqueado
6	7376	11	17%	90%	como solicitar portabilidade de salário?
7	7463	12	18%	89%	quero fala com a [redacted]
8	7464	10	18%	87%	quero cancelar o cartão de crédito
9	7469	15	18%	88%	fechamento da fatura do cartão de crédito
10	7474	11	18%	88%	quero saber se tenho limite de crédito no meu cartão de crédito
11	7506	11	18%	89%	quero saber como faço pra obter a função crédito no meu cartão
12	7547	15	18%	88%	prefeitura municipal de [redacted]
13	7550	14	18%	90%	quero falar com o gerente da minha conta

Fonte: **O Autor**

A tabela pode ser ordenada de acordo com os valores de qualquer coluna. A primeira, sem título, simplesmente ajuda o usuário na navegação da tabela ao indexar cada linha. As demais, cujos nomes são abreviados para melhor exibição na tela, são descritas a seguir na ordem em que aparecem da esquerda para a direita:

1. Número do Agrupamento: identifica o agrupamento, sendo gerado tal como na construção do dendrograma da Figura 5;

2. Número de Sentenças: quantidade de sentenças que compõem o agrupamento;

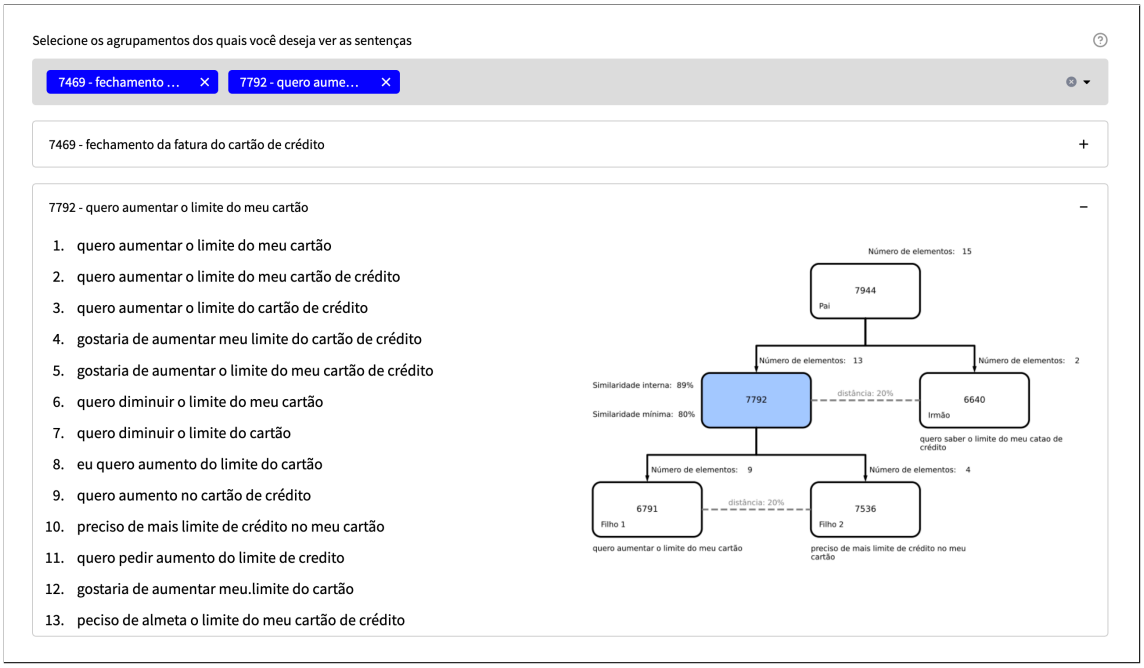
3. Distância Interna: valor da distância, dada pela ligação completa, entre os subgrupos que foram aglomerados na criação do agrupamento;

4. Similaridade Interna: média de similaridades de todos os pares de sentenças do agrupamento;

5. Sentença Exemplo: sentença com maior similaridade média com todas as outras sentenças do agrupamento, representando o elemento mais próximo ao centroide do grupo. Ela é apresentada ao usuário para que ele possa identificar o assunto central das sentenças pertencentes ao agrupamento.

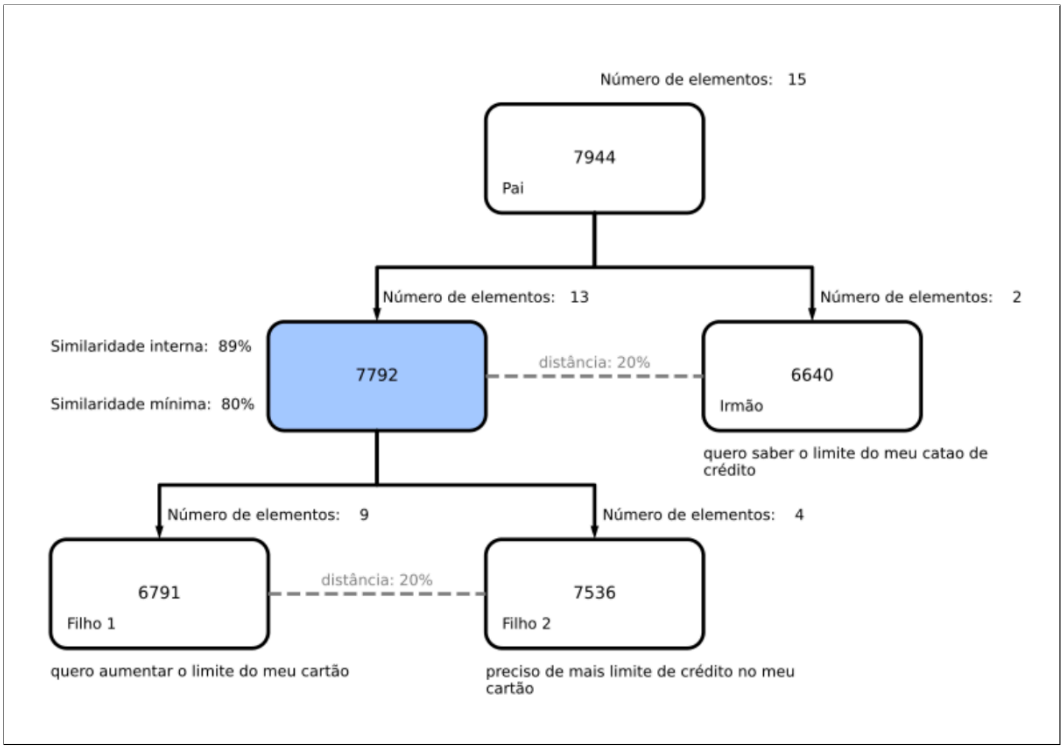
Na sequência, é apresentada uma interface, exposta na Figura 10, na qual o usuário pode escolher os agrupamentos que ele deseja explorar por meio de um campo de entrada de seleção múltipla. Cada grupo é unicamente identificado pelo Número do Agrupamento, o qual é acompanhado, por conveniência, da Sentença Exemplo. Os agrupamentos escolhidos são então expostos na sequência, em seções individuais que podem ser expandidas e retraídas. À esquerda de cada seção são listadas as sentenças daquele agrupamento, em ordem decrescente de similaridade média com as demais sentenças do agrupamento, ou seja, em ordem decrescente de proximidade ao centroide do agrupamento. Já à direita é apresentado um recorte do dendrograma com os agrupamentos vizinhos ao escolhido, juntamente de algumas métricas de interesse, o qual pode ser melhor observado na Figura 11. Com essa informação, o usuário pode explorar tanto os subgrupos daquele agrupamento (denominados de grupos filhos na aplicação) quanto os grupos envolvidos na sua aglomeração (os grupos irmão e pai).

Figura 10: *Agrupamentos selecionados pelo usuário.*



Fonte: **O Autor**

Figura 11: *Detalhe da Figura 10 mostrando o recorte no dendrograma da vizinhança do agrupamento selecionado.*



Fonte: **O Autor**

Para um conjunto de n sentenças, o número de grupos gerados pelo algoritmo de agrupamento hierárquico aglomerativo é igual a $n-1$ (excluindo-se os grupos unitários).

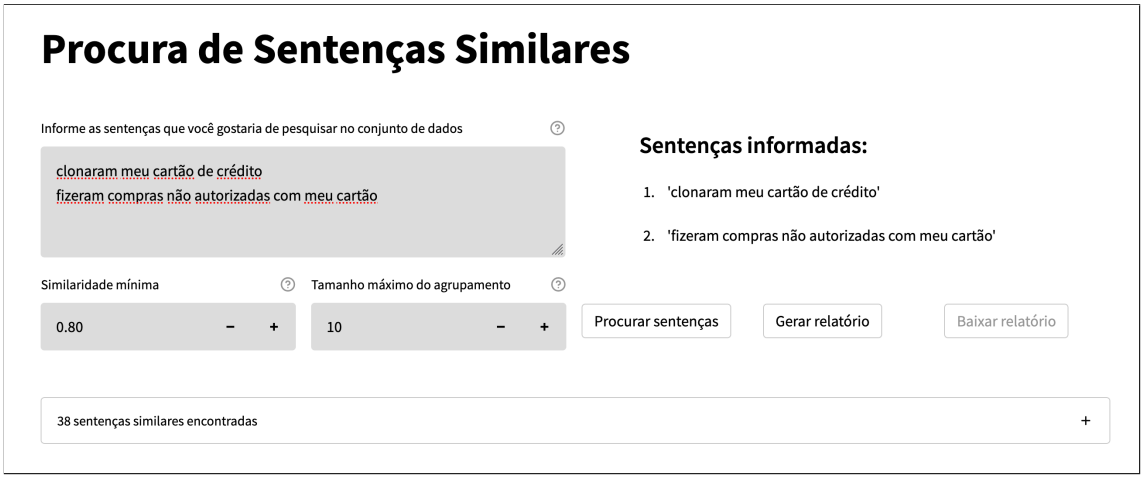
Assim, para o conjunto explorado de 5.170 sentenças, 5.169 grupos seriam expostos na tela, além do eixo horizontal precisar comportar os 5.170 elementos. De tal forma, por ser de difícil visualização e não trazer informação significativa ao usuário da aplicação, o dendrograma completo não é apresentado.

Muitas vezes, no entanto, a visualização da informação pela aplicação *web* não é suficiente. Seja para compartilhar o resultado com outras pessoas, armazenar o material ou realimentar o modelo do *chatbot*, a informação é comumente manipulada por meio de planilhas. De tal forma, desenvolveu-se funcionalidade para que o usuário possa baixar um relatório como um arquivo Excel com todos os resultados apresentados na página. Devido às limitações do Streamlit, foi necessário, vide Figura 8, fazer uso de dois botões: o botão "*Gerar relatório*" cria o arquivo e o salva em memória, enquanto o botão "*Baixar relatório*" (habilitado ao clique após criado o arquivo) faz a sua transferência para a máquina do usuário.

3.6.3 Página para Procura de Sentenças Similares

Na última página da aplicação *web*, desenvolveu-se uma funcionalidade adicional para que o usuário possa procurar por sentenças similares a um conjunto de sentenças por ele informadas no campo de entrada textual da Figura 12, as quais devem ser digitadas uma por linha. Ao clicar no botão "*Procurar sentenças*", um retorno visual das sentenças digitadas é apresentado logo à direita e elas são então transformadas em representações vetoriais da mesma forma como foi estabelecido ao conjunto de 5.170 mensagens. Na sequência, para cada sentença do conjunto inicial de mensagens, obtém-se o máximo valor de similaridade por cosseno entre ela e as sentenças informadas pelo usuário. Assim, basta que a mensagem seja similar a uma das sentenças informadas para que ela seja considerada relevante.

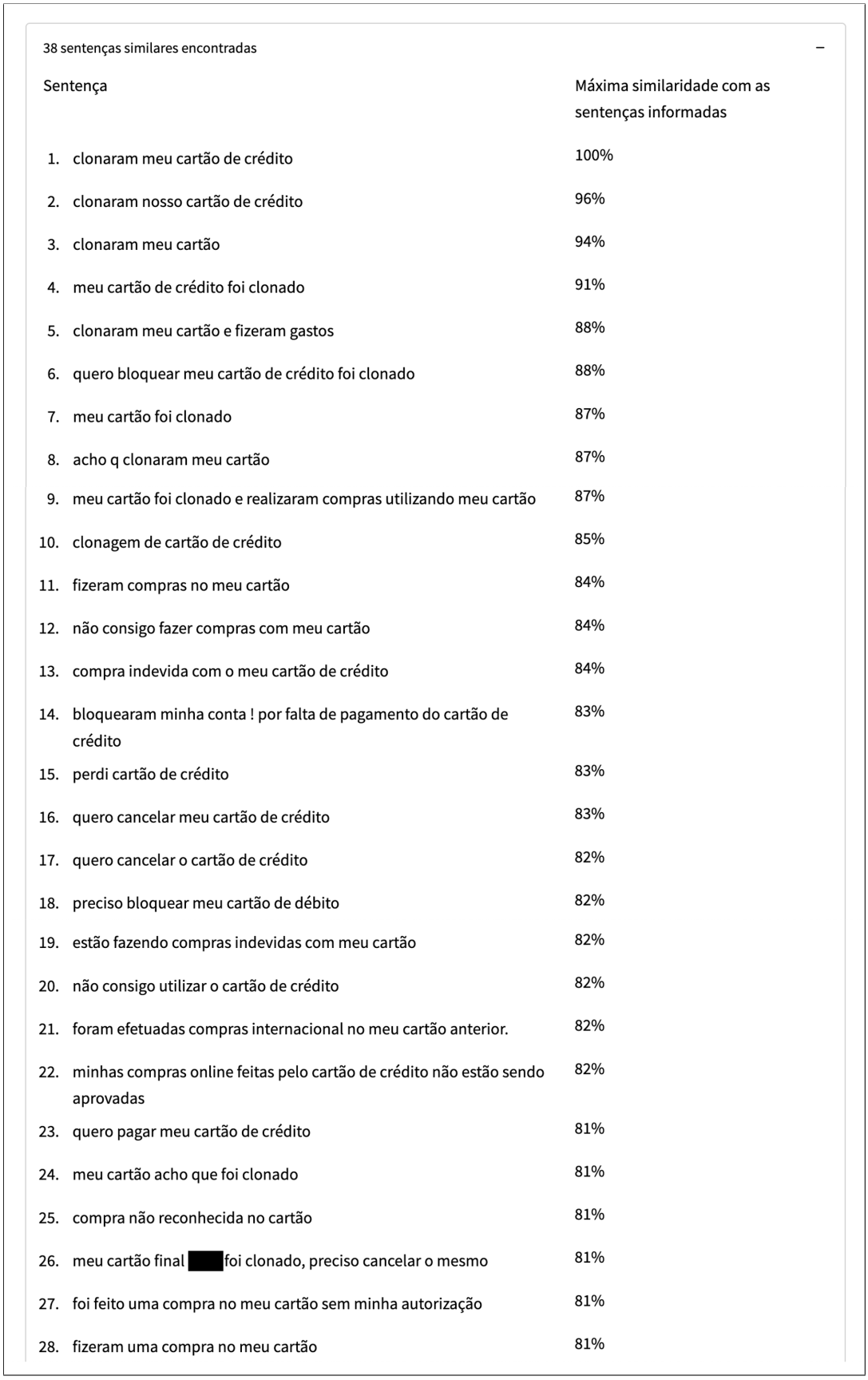
Figura 12: *Definição dos parâmetros de pesquisa.*



Fonte: **O Autor**

Dessa operação, são selecionadas as sentenças com similaridade (máxima) maior que um determinado valor informado pelo usuário em um campo de entrada numérica, conforme observado na Figura 12. Numa seção logo abaixo, que pode ser expandida (vide Figura 13) ou retraída (vide Figura 12), essas sentenças são apresentadas em ordem decrescente de similaridade máxima com as sentenças informadas. Em uma coluna à direita, tal valor é apresentado em formato percentual.

Figura 13: *Sentenças similares encontradas.*



38 sentenças similares encontradas

Sentença	Máxima similaridade com as sentenças informadas
1. clonaram meu cartão de crédito	100%
2. clonaram nosso cartão de crédito	96%
3. clonaram meu cartão	94%
4. meu cartão de crédito foi clonado	91%
5. clonaram meu cartão e fizeram gastos	88%
6. quero bloquear meu cartão de crédito foi clonado	88%
7. meu cartão foi clonado	87%
8. acho q clonaram meu cartão	87%
9. meu cartão foi clonado e realizaram compras utilizando meu cartão	87%
10. clonagem de cartão de crédito	85%
11. fizeram compras no meu cartão	84%
12. não consigo fazer compras com meu cartão	84%
13. compra indevida com o meu cartão de crédito	84%
14. bloquearam minha conta ! por falta de pagamento do cartão de crédito	83%
15. perdi cartão de crédito	83%
16. quero cancelar meu cartão de crédito	83%
17. quero cancelar o cartão de crédito	82%
18. preciso bloquear meu cartão de débito	82%
19. estão fazendo compras indevidas com meu cartão	82%
20. não consigo utilizar o cartão de crédito	82%
21. foram efetuadas compras internacional no meu cartão anterior.	82%
22. minhas compras online feitas pelo cartão de crédito não estão sendo aprovadas	82%
23. quero pagar meu cartão de crédito	81%
24. meu cartão acho que foi clonado	81%
25. compra não reconhecida no cartão	81%
26. meu cartão final [redacted] foi clonado, preciso cancelar o mesmo	81%
27. foi feito uma compra no meu cartão sem minha autorização	81%
28. fizeram uma compra no meu cartão	81%

Fonte: **O Autor**

Na sequência, é exposta a tabela vista na Figura 14, na qual são descritos os agrupamentos que contêm pelo menos uma das sentenças consideradas similares. Evidentemente, o agrupamento raiz contém essas sentenças, além de todas as outras. Dessa forma, é necessário definir um filtro (um corte superior) a ser aplicado ao dendrograma que representa a estrutura de agrupamento. Embora qualquer combinação de um ou mais parâmetros pudesse ser utilizada, optou-se por empregar, por simplicidade, apenas um, de modo que o usuário também deve informar o tamanho máximo do agrupamento na interface da Figura 12. Vale notar que até então o tamanho do agrupamento era utilizado como um valor mínimo, pois ele não era utilizado para definir uma linha de corte superior ao dendrograma, servindo apenas para suprimir agrupamentos com poucos elementos.

Figura 14: *Agrupamentos que contêm as sentenças similares.*

Número de agrupamentos encontrados: 20

	Nº Agrupamento	Nº Sentenç. Simil.	Nº Sentenç.	% Sentenç. Simil.	Dist. Interna	Simil. Interna	Sentença Exemplo
1	6678	2	4	50%	13%	90%	não consigo fazer compras ..
2	7171	3	9	33%	16%	89%	só quero desbloquear meu ..
3	7253	1	7	14%	16%	91%	quero parcelar fatura do m..
4	7323	1	4	25%	17%	88%	meu cartão ainda não chego
5	7391	1	6	17%	17%	86%	foi realizada uma compra n..
6	7464	2	10	20%	18%	87%	quero cancelar o cartão de ..
7	7570	1	4	25%	18%	85%	parcelamento do meu cart...
8	7867	3	6	50%	20%	85%	fraude no cartão de crédito
9	7900	1	8	12%	20%	87%	esqueci a senha do cartão ...
10	8079	1	9	11%	21%	86%	quero minha fatura do cartã
11	8094	2	9	22%	21%	85%	liberar limite do meu cartã...
12	8241	1	9	11%	22%	83%	abri a conta mas não recebi..
13	8275	1	6	17%	23%	82%	não consigo usar o crédito ..

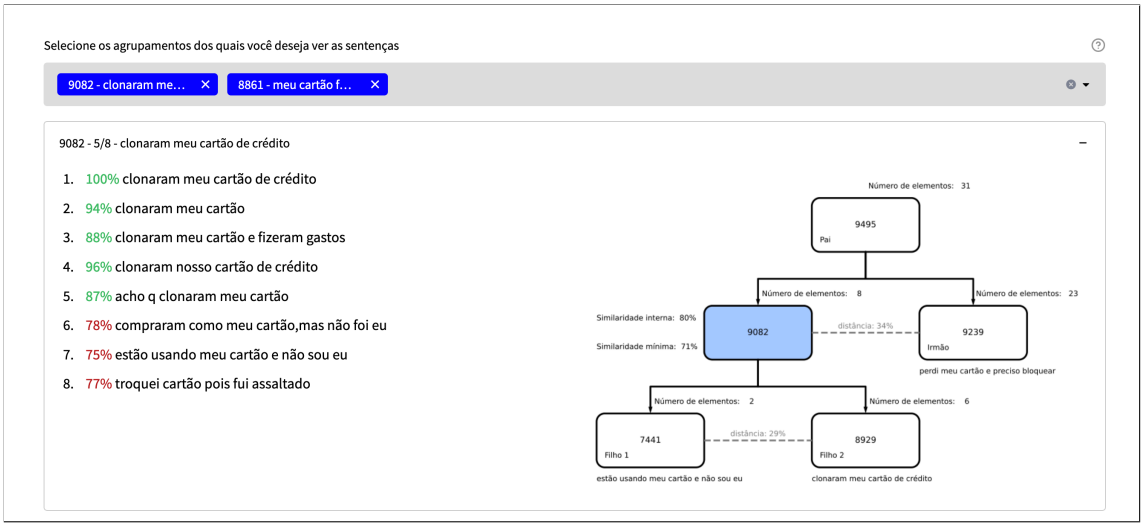
Fonte: **O Autor**

Essa tabela adiciona à estrutura daquela apresentada da página anterior, descrita na Seção 3.6.2, duas novas colunas. A primeira informa o número de sentenças pertencentes ao agrupamento que são similares às sentenças digitadas pelo usuário da aplicação. Já a segunda apresenta esse valor em formato percentual, o qual é obtido da razão entre esse número de sentenças similares e a quantidade total de sentenças do agrupamento.

Ao final da página, de maneira similar à anterior, é apresentada a interface da Figura 15 para que o usuário possa explorar os agrupamentos de interesse. Nas seções dedicadas a cada grupo, além de serem exibidas as sentenças e o recorte do dendrograma, é indicado, ao lado esquerdo da sentença, a sua máxima similaridade com as sentenças informadas pelo usuário. Tal informação recebe coloração verde quando for maior ou igual ao valor de similaridade mínima definida pelo usuário, ou vermelho quando for menor. Ademais, para cada agrupamento, destaca-se a razão entre o número de sentenças consideradas similares e o número total de sentenças.

Por fim, a página também conta com a funcionalidade de extração de relatório por meio de arquivo Excel. Para tanto, tal como na página anterior, o botão "*Gerar relatório*" cria o arquivo com os resultados apresentados na página e o salva em memória, enquanto o botão "*Baixar relatório*" o transfere para a máquina do usuário, vide Figura 12.

Figura 15: *Agrupamentos selecionados pelo usuário.*



Fonte: **O Autor**

4 RESULTADOS E ANÁLISE

Não é fácil avaliar a qualidade dos resultados de um algoritmo de agrupamento (PALACIO-NIÑO; BERZAL, 2019). Por trabalhar de maneira não supervisionada com dados não rotulados, não é possível comparar os agrupamentos obtidos contra um determinado conjunto de elementos já conhecido e tomado como uma verdade absoluta. De tal forma, diferentes técnicas de validação foram propostas, as quais dependem das características do problema de agrupamento e da técnica empregada pelo algoritmo.

É importante notar que, embora tenha sido empregado um método de agrupamento hierárquico, o resultado final da solução desenvolvida não é a hierarquia entre os elementos. O processo de filtragem, apresentado na Seção 3.5.2, obtém os supergrupos que satisfazem as condições impostas, juntamente dos seus elementos, descartando-se, para aquele determinado cenário, as relações de hierarquia com os demais grupos. Assim, a avaliação deve fazer uso dos métodos destinados a algoritmos particionais, os quais, em oposição aos hierárquicos, separam os elementos em agrupamentos independentes, sem que haja uma relação de hierarquia entre eles.

Na validação interna, em geral, duas métricas são utilizadas: a coesão, que mede quão próximos os elementos de um mesmo agrupamento estão entre si, e a separação, que mede quão afastados os distintos agrupamentos estão (PALACIO-NIÑO; BERZAL, 2019). Um bom agrupamento deve apresentar altas coesão e separação, as quais podem ser combinadas de diferentes formas a fim de gerar uma única métrica, sendo a mais comum a *silhouette score* (do inglês, pontuação de silhueta).

O valor de *silhouette score* relativo a uma determinada configuração de agrupamentos é igual à média dos valores de *silhouette score* para cada elemento, cujo cálculo é apresentado na Equação 4. Nela, o termo a representa a distância média entre o elemento e todos os outros elementos do mesmo agrupamento, enquanto o termo b representa a distância média entre o elemento e todos os elementos do agrupamento mais próximo ao elemento considerado (KUMAR, 2020). Sendo limitado ao intervalo $[-1,1]$, pontuações próximas: a 1 indicam que os agrupamentos são densos e bem separados; a 0, que os agrupamentos se sobrepõem, com elementos muito próximos às fronteiras dos agrupamentos vizinhos; a -1, que os elementos foram atribuídos ao agrupamento errado.

$$s = \frac{b-a}{max(a,b)} \tag{4}$$

Na Tabela 3 é possível observar os valores de *silhouette score* para os agrupamentos encontrados, no estudo de caso abordado, para algumas condições de filtragem distintas. Nela, percebe-se que quanto mais restrito for o filtro (quanto menor for a distância interna máxima e quanto maiores forem a similaridade interna mínima e o tamanho mínimo do agrupamento), maior tende a ser o *silhouette score*, pois a filtragem remove agrupamentos

de baixa qualidade, ou seja, com baixas coesão e separação. No entanto, como o filtro exclui alguns grupos, deve-se observar, além da pontuação, a quantidade de agrupamentos obtidos, pois uma alta pontuação com apenas 2 agrupamentos não é satisfatória para um conjunto de 5.170 sentenças. Assim, alguns resultados podem ser observados na Tabela 3 com pontuações perto de 0,5 e com número de agrupamentos na ordem de poucas dezenas.

Tabela 3: *Silhouette score e número de agrupamentos encontrados (em parênteses) para diferentes parâmetros de filtragem.*

Distância interna máxima	Similaridade interna mínima	*Silhouette score* (número de grupos)
0,10	0,80	0,56 (27)
0,10	0,90	0,56 (27)
0,10	0,95	0,57 (16)
0,20	0,80	0,27 (202)
0,20	0,90	0,40 (86)
0,20	0,95	0,58 (17)
0,30	0,80	0,17 (275)
0,30	0,90	0,41 (85)
0,30	0,95	0,58 (17)

(a) *Tamanho mínimo do agrupamento igual a 5.*

Distância interna máxima	Similaridade interna mínima	*Silhouette score* (número de grupos)
0,10	0,80	0,84 (2)
0,10	0,90	0,84 (2)
0,10	0,95	— (1)
0,20	0,80	0,38 (32)
0,20	0,90	0,46 (13)
0,20	0,95	— (1)
0,30	0,80	0,17 (115)
0,30	0,90	0,46 (14)
0,30	0,95	— (1)

(b) *Tamanho mínimo do agrupamento igual a 10.*

Fonte: **O Autor**

Nota: O cálculo de *silhouette score* é definido apenas quando o número de agrupamentos pertence ao intervalo $[2, n_e - 1]$, no qual n_e é a quantidade total de elementos agrupados. Caso o número esteja fora desse intervalo, o resultado é omitido pelo símbolo "—".

É muito importante observar, no entanto, que o algoritmo de agrupamento opera sobre o conjunto de vetores. Assim, não se está avaliando, diretamente, o agrupamento das sentenças, mas sim o agrupamento das suas representações vetoriais. E, como não há uma metodologia correta e definitiva para as suas obtenções, diferentes métodos resultam em diferentes vetores para uma mesma sentença, sem que haja uma representação completamente fiel. Logo, os diferentes vetores obtidos de diferentes métodos impactam nos valores de *silhouette score* sem necessariamente refletir na qualidade do agrupamento final dado às sentenças. Por conseguinte, não é apropriado se valer unicamente da análise de *silhouette score* para avaliar o resultado completo da solução desenvolvida.

Dessa forma, os agrupamentos obtidos também foram manualmente investigados, de modo que analisou-se, diretamente, o conteúdo das sentenças, desconsiderando-se as suas representações vetoriais. Embora tal avaliação seja subjetiva, sendo impossível executar ensaios programáticos e comparar métricas, esse é o tipo de análise que melhor se adapta ao cenário do trabalho, pois o sentido das sentenças é observado por um avaliador humano sobre as mensagens originais em linguagem natural.

A Tabela 4 apresenta um dos agrupamentos gerados pela solução para as mensagens do estudo de caso considerado. Nela, observa-se que sentenças com palavras distintas, mas que transmitem uma mensagem similar, foram, com sucesso, agrupadas. Salvo, contudo, a última sentença, que apresenta sentido mais afastado das demais, tal conjunto

poderia ser utilizado na construção de uma intenção que identificaria mensagens relativas a suspeitas de clonagem de cartão de crédito.

Tabela 4: *Agrupamento 7867. Número de sentenças: 6. Distância interna: 20%. Similaridade interna: 85%.*

Sentenças
Fraude no cartão de crédito
Cartão de crédito com cobrança indevida
Clonagem de cartão de crédito
Compra indevida com o meu cartão de crédito
Uso indevido de cartão de crédito
Problema com cartão de crédito

Fonte: **O Autor**

Embora todas as sentenças da Tabela 4 apresentem o termo "cartão de crédito" em comum, é importante ressaltar que, apesar de ele ser importante à construção do sentido da sentença, a sua presença não é o fator determinante à formação do agrupamento. Do conjunto de mensagens estudado, um total de 261 sentenças contêm esse termo com essa exata grafia, as quais tratam sobre diversos assuntos. Um deles pode ser observado na Tabela 5, na qual "cartão de crédito" é utilizado em outro contexto. Nela, nota-se, novamente, que sentenças construídas com palavras distintas, mas que tratam sobre um mesmo assunto, foram agrupadas com sucesso.

Tabela 5: *Agrupamento 7792. Número de sentenças: 13. Distância interna: 20%. Similaridade interna: 89%.*

Sentenças
Quero aumentar o limite do meu cartão
Quero aumentar o limite do meu cartão de crédito
Quero aumentar o limite do cartão de crédito
Gostaria de aumentar meu limite do cartão de crédito
Gostaria de aumentar o limite do meu cartão de crédito
Quero diminuir o limite do meu cartão
Quero diminuir o limite do cartão
Eu quero aumento do limite do cartão
Quero aumento no cartão de crédito
Preciso de mais limite de crédito no meu cartão
Quero pedir aumento do limite de credito
Gostaria de aumentar meu.limite do cartão
Peciso de almeta o limite do meu cartão de crédito

Fonte: **O Autor**

É bem verdade, todavia, que as palavras "aumentar" e "diminuir", na Tabela 5, configuram solicitações opostas às mensagens. Contudo, ambas possuem um sentido comum relativo a fazer uma mudança, seja ela positiva ou negativa, em determinada propriedade. Portanto, segundo a hipótese distribucional, elas são semelhantes porque apresentam palavras vizinhas em comum no conjunto de dados de treinamento, de modo que podem

ser substituídas em uma frase sem que haja estranhamento na nova construção. Assim, verifica-se que a similaridade por cosseno entre "quero aumentar o limite do meu cartão" e "quero diminuir o limite do meu cartão", utilizando-se as configurações estabelecidas ao longo do trabalho, é igual a 0,98.

No Apêndice A são exibidos outros agrupamentos encontrados pela aplicação da solução sobre o conjunto de mensagens do estudo de caso. Evidentemente, os exemplos listados representam apenas algumas amostras do resultado obtido.

Ainda, é possível perceber que várias sentenças dos agrupamentos expostos possuem erros ortográficos. No entanto, isso não foi um impeditivo para que o sentido da sentença fosse obtido e ela fosse agrupada com outras sentenças com sentidos semelhantes.

Por fim, em complemento à validação interna, é também possível realizar uma validação externa dos resultados (PALACIO-NIÑO; BERZAL, 2019). Nesse cenário, um conjunto de dados externo, conhecido e rotulado, deve ser utilizado para avaliar o algoritmo de agrupamento. Assim, desenvolveria-se um ensaio no qual seriam agrupadas as sentenças de um conjunto de intenções reais do *chatbot*. O resultado obtido seria então comparado aos agrupamentos esperados, de modo que sentenças de uma mesma intenção deveriam pertencer a um mesmo grupo. No entanto, como tais dados não estavam disponíveis, esse tipo de validação não foi executada.

5 CONCLUSÕES

Com a construção da solução, três etapas foram determinadas a fim de se atingir o objetivo de agrupamento das mensagens com sentido semelhante, as quais envolvem a obtenção, o processamento e a visualização dos dados. Para cada etapa, um conjunto de códigos independentes foi desenvolvido, todos escritos em linguagem de programação Python, que podem ser facilmente aplicados a um cenário real de desenvolvimento de *chatbots* cognitivos.

Os dados utilizados, contendo as mensagens dos usuários, foram obtidos de um arquivo texto, o qual foi disponibilizado por uma instituição financeira para fins de desenvolvimento deste trabalho. Entretanto, para uma melhor automação do processo, a leitura desse arquivo deve ser substituída pela consulta ao banco de dados que armazena os registros de interação entre usuários e *chatbot*, aplicando-se a filtragem quanto ao tamanho das sentenças que devem ser descartadas.

A partir do conjunto de instruções definidas no código de processamento, são então obtidas as representações vetoriais das mensagens, a matriz de similaridades entre todos os pares de sentenças e o agrupamento das mensagens por meio do método não supervisionado de agrupamento hierárquico aglomerativo. Além dessas, outras informações são geradas para desonerar a ferramenta de visualização de alguns processamentos no momento da sua utilização. Vale notar que esses resultados, até agora armazenados localmente como arquivos binários do tipo *pickle*, devem ser enviados a um banco de dados para posterior consumo pela ferramenta de visualização.

A última etapa, de visualização dos dados, faz uso de uma aplicação *web*, desenvolvida como uma prova de conceito utilizando a biblioteca Streamlit, para que a informação gerada seja acessível a todo o time que trabalha no desenvolvimento do *chatbot* cognitivo. Sendo composta por três páginas, a primeira informa ao usuário a distribuição de similaridades e de distâncias entre as sentenças no conjunto de dados e entre os agrupamentos gerados, a segunda permite explorar os agrupamentos encontrados e a terceira habilita o usuário a procurar por sentenças similares às por ele informadas.

Com a execução de todas as atividades, foi possível atingir o objetivo proposto, agrupando-se as mensagens dos usuários, de acordo com a similaridade no seu conteúdo, a fim de automatizar o processo de descoberta de novos assuntos a serem tratados pelo *chatbot* cognitivo, culminando na criação de novas intenções compostas pelas sentenças agrupadas. Ademais, o resultado também pode ser aproveitado para retroalimentar o modelo de NLP do *chatbot*, melhorando a composição das intenções já existentes com a inserção de novas sentenças.

Embora tenham sido utilizadas mensagens enviadas por usuários ao *chatbot*, a solução apresentada é independente quanto à origem dos textos. Por conseguinte, também podem ser utilizadas mensagens enviadas por usuários a agentes humanos, de modo a se conhecer

quais são os tópicos abordados em outros canais. Assim, o *chatbot* pode ser munido com as intenções e os fluxos de conversação necessários para automatizar tal atendimento.

Quanto à metodologia empregada, notou-se que um modelo de linguagem portuguesa brasileira genérico como o BERTimbau-Base foi capaz de extrair o sentido das sentenças para posterior agrupamento. Assim, a obtenção da representação vetorial das sentenças a partir da média entre os vetores, da última camada, relativos a cada *token* e o emprego da métrica de similaridade por cosseno foram capazes de identificar que sentenças, mesmo quando construídas com palavras distintas e/ou com erros ortográficos, transmitem uma mensagem similar.

O método de agrupamento hierárquico aglomerativo mostrou-se efetivo ao cumprimento do objetivo do trabalho. Como as sentenças são agrupadas gradativamente, garante-se que os grupos são inicialmente formados por sentenças com alta semelhança entre si, a qual diminui com o avanço na execução do algoritmo. Dessa forma, pela filtragem do resultado final obtido, é possível observar grupos com a similaridade desejada e remover sentenças e agrupamentos que não são relevantes à análise, proporcionando-se flexibilidade àquele interessado na investigação dos resultados.

Uma vantagem desse método de agrupamento é que cada configuração de parâmetros à filtragem não exige uma nova execução do algoritmo, pois basta investigar a estrutura hierárquica resultante. Não avaliou-se, no entanto, a sua performance quanto ao tempo de processamento e aos recursos computacionais exigidos frente a outros métodos, os quais podem ser abordados em trabalhos futuros.

Caso necessário, nota-se que o número de etapas processadas pelo algoritmo de agrupamento pode ser reduzido ao remover-se do conjunto as sentenças que apresentarem baixa similaridade máxima com as demais, as quais podem ser consideradas *outliers*, para os quais não há outras sentenças com sentido semelhante. Ademais, nota-se que o algoritmo de agrupamento hierárquico aglomerativo é executado até que todas as sentenças pertençam a um único grupo. No entanto, a partir de certa distância entre os grupos, os agrupamentos não fazem mais sentido ao objetivo almejado, pois são agrupadas sentenças muito distintas entre si. De tal forma, o algoritmo pode ser otimizado a fim de reduzir o tempo de processamento evitando-se o cálculo do dendrograma completo, o qual deve ser construído até que determinada distância de agrupamento seja atingida.

Para trabalhos futuros, sugere-se a investigação dos resultados obtidos por meio da aplicação da variante Large do BERTimbau. Como a sua rede neural é composta por pouco mais que o triplo de parâmetros utilizados na versão Base e seus vetores possuem um terço a mais de elementos, espera-se que as representações vetoriais resultantes da sua utilização sejam mais ricas de informação, contemplando maior conhecimento a respeito das sentenças. Além disso, também podem ser inseridos à analise os modelos multilíngue do BERT e do SBERT, a fim de avaliar se tais modelos geram representações vetoriais mais interessantes que as obtidas pelo modelo específico à língua portuguesa brasileira. Por fim, também é possível investigar os resultados obtidos para diferentes abordagens de obtenção da representação vetorial da sentença (como pela utilização de outras camadas da rede neural além da última) e de definição da distância entre grupos no algoritmo de agrupamento.

No entanto, mantém-se a dificuldade em avaliar a qualidade dos agrupamentos obtidos. Para um conjunto de dados não anotados, a tarefa é melhor executada pela investigação manual dos resultados, apesar da subjetividade inerente ao processo. Assim, caso possível, recomenda-se também fazer uso de um conjunto anotado já existente de intenções conhecidas pelo *chatbot* a fim de validar os agrupamentos encontrados.

REFERÊNCIAS

BERETTA, L. *Lu, do Magalu, se aproxima dos clientes e registra 8,5 milhões de interações ao mês com inteligência artificial de IBM Watson*. IBM. 2020. Disponível em: <https://www.ibm.com/blogs/ibm-comunica/lu-do-magalu-se-aproxima-dos-clientes-e-registra-85-milhoes-de-interacoes-ao-mes-com-inteligencia-artificial-de-ibm-watson/>. Acesso em: 20 fev. 2022.

BOJANOWSKI, P. et al. *Enriching Word Vectors with Subword Information*. [S.l.]: arXiv, 2016. DOI: 10.48550/ARXIV.1607.04606. Disponível em: <https://arxiv.org/abs/1607.04606>.

BROWNLEE, J. *A Gentle Introduction to the Bag-of-Words Model*. Machine Learning Mastery. 2017a. Disponível em: <https://machinelearningmastery.com/gentle-introduction-bag-words-model/>. Acesso em: 26 fev. 2022.

BROWNLEE, J. *How to Develop Word Embeddings in Python with Gensim*. Machine Learning Mastery. 2017b. Disponível em: <https://machinelearningmastery.com/develop-word-embeddings-python-gensim/>. Acesso em: 27 fev. 2022.

BROWNLEE, J. *What Is Natural Language Processing?* Machine Learning Mastery. 2017c. Disponível em: <https://machinelearningmastery.com/natural-language-processing/>. Acesso em: 20 fev. 2022.

CODEEMPORIUM. *BERT Neural Network - EXPLAINED!* YouTube. 2020. Disponível em: <https://www.youtube.com/watch?v=xI0HHN5XKDo>. Acesso em: 1 mar. 2022.

DEVLIN, J.; CHANG, M.-W. *Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing*. Google AI Blog. 2018. Disponível em: <https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html>. Acesso em: 1 mar. 2022.

DEVLIN, J.; CHANG, M.-W. et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. [S.l.: s.n.], 2019. arXiv: 1810.04805 [cs.CL].

GOMEZ-PEREZ, J. M.; DENAUX, R.; GARCIA-SILVA, A. Understanding Word Embeddings and Language Models. In: A Practical Guide to Hybrid Natural Language Processing: Combining Neural Models and Knowledge Graphs for NLP. Cham: Springer International Publishing, 2020. p. 17–31. ISBN 978-3-030-44830-1. DOI: 10.1007/978-3-030-44830-1_3. Disponível em: <https://doi.org/10.1007/978-3-030-44830-1_3>.

CLOUD TPU: Treine e execute modelos de machine learning mais rápido do que nunca. Google Cloud. Disponível em: <https://cloud.google.com/tpu>. Acesso em: 5 mar. 2022.

HAJJAR, A. J. *In-Depth Guide Into Chatbots Intent Recognition in 2022*. AIMultiple. 2022. Disponível em: <https://research.aimultiple.com/chatbot-intent/>. Acesso em: 7 mai. 2022.

HEBBAR, P. *Why Should You Learn Python For Data Science?* Analytics India Magazine. 2019. Disponível em: <https://analyticsindiamag.com/why-should-you-learn-python-for-data-science/>. Acesso em: 22 mar. 2022.

BERT. Hugging Face. Disponível em: <https://huggingface.co/docs/transformers/model_doc/bert>. Acesso em: 7 mai. 2022.

THE AI community building the future. Build, train and deploy state of the art models powered by the reference open source in machine learning. Hugging Face. Disponível em: <https://huggingface.co/>. Acesso em: 5 mar. 2022.

TRANSFORMERS. Hugging Face. Disponível em: <https://huggingface.co/docs/transformers/index>. Acesso em: 5 mar. 2022.

VOLKSWAGEN Virtus será o primeiro carro no Brasil a usar inteligência artificial para ajudar motoristas. IBM. 2017. Disponível em: <https://www.ibm.com/blogs/ibm-comunica/volkswagen-virtus-usa-inteligencia-artificial/>. Acesso em: 7 mai. 2022.

JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing*. Stanford. Disponível em: <https://web.stanford.edu/~jurafsky/slp3/>. Acesso em: 26 fev. 2022.

KASSAMBARA, A. *Agglomerative Hierarchical Clustering*. Datanovia. Disponível em: <https://www.datanovia.com/en/lessons/agglomerative-hierarchical-clustering/>. Acesso em: 21 mar. 2022.

KUMAR, A. *KMeans Silhouette Score Explained with Python Example*. Vitalflux. 2020. Disponível em: <https://vitalflux.com/kmeans-silhouette-score-explained-with-python-example/>. Acesso em: 1 abr. 2022.

LEE, J. et al. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Edição: Jonathan Wren. *Bioinformatics*, Oxford University Press (OUP), set. 2019. ISSN 1460-2059. DOI: 10.1093/bioinformatics/btz682. Disponível em: <http://dx.doi.org/10.1093/bioinformatics/btz682>.

LIU, Q.; KUSNER, M. J.; BLUNSOM, P. *A Survey on Contextual Embeddings*. [S.l.: s.n.], 2020. arXiv: 2003.07278 [cs.CL].

MCCORMICK, C. *BERT Word Embeddings Tutorial*. 2019. Disponível em: <https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/>. Acesso em: 6 mar. 2022.

MIKOLOV, T. et al. *Efficient Estimation of Word Representations in Vector Space*. [S.l.]: arXiv, 2013. DOI: 10.48550/ARXIV.1301.3781. Disponível em: <https://arxiv.org/abs/1301.3781>.

MORENO, C. *diacríticos*. Disponível em: <https://sualingua.com.br/diacriticos/>. Acesso em: 12 mar. 2022.

NAYAK, P. *Understanding searches better than ever before*. Google. 2019. Disponível em: <https://blog.google/products/search/search-language-understanding-bert/>. Acesso em: 1 mar. 2022.

NICKERSON, S. *Chatbot Intents — Simple Explanation & Fundamentals*. The Chatbot Business Framework. 2020. Disponível em: <https://chatbotbusinessframework.com/chatbot-intents/>. Acesso em: 7 mai. 2022.

PAI, A. *What is Tokenization in NLP? Here's All You Need To Know*. Analytics Vidhya. 2020. Disponível em: <https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/>. Acesso em: 1 mar. 2022.

PAIVA, F. *Pesquisa Panorama Mobile Time - Mapa do Ecossistema Brasileiro de Bots - Agosto de 2021*. Mobile Time. 2021.

PALACIO-NIÑO, J.-O.; BERZAL, F. *Evaluation Metrics for Unsupervised Learning Algorithms*. [S.l.]: arXiv, 2019. DOI: 10.48550/ARXIV.1905.05667. Disponível em: <https://arxiv.org/abs/1905.05667>.

PENG, T.; SARAZEN, M. *The Staggering Cost of Training SOTA AI Models*: While it is exhilarating to see AI researchers pushing the performance of cutting-edge models to new heights, the costs of such processes are also rising at a dizzying rate. Synced. 2019. Disponível em: <https://syncedreview.com/2019/06/27/the-staggering-cost-of-training-sota-ai-models/>. Acesso em: 5 mar. 2022.

PENNINGTON, J.; SOCHER, R.; MANNING, C. GloVe: Global Vectors for Word Representation. In: PROCEEDINGS of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, out. 2014. p. 1532–1543. DOI: 10.3115/v1/D14-1162. Disponível em: <https://aclanthology.org/D14-1162>.

PERES, S. M.; LIMA, C. A. M. *Técnicas de Agrupamento (Clustering)*. 2015. Disponível em: <http://each.uspnet.usp.br/sarajane/wp-content/uploads/2015/09/clustering1.pdf>. Acesso em: 21 mar. 2022.

PETERS, M. E. et al. *Deep contextualized word representations*. [S.l.: s.n.], 2018. arXiv: 1802.05365 [cs.CL].

PIETRO, M. D. *Text Classification with NLP: Tf-Idf vs Word2Vec vs BERT*: Preprocessing, Model Design, Evaluation, Explainability for Bag-of-Words, Word Embedding, Language models. Towards Data Science. 2020. Disponível em: <https://towardsdatascience.com/text-classification-with-nlp-tf-idf-vs-word2vec-vs-bert-41ff868d1794>. Acesso em: 26 fev. 2022.

POGIATZIS, A. *NLP: Contextualized word embeddings from BERT*: Extract contextualized word embeddings from BERT using Keras and TF. Towards Data Science. 2019. Disponível em: <https://towardsdatascience.com/nlp-extract-contextualized-word-embeddings-from-bert-keras-tf-67ef29f60a7b>. Acesso em: 1 mar. 2022.

RADFORD, A.; NARASIMHAN, K. Improving Language Understanding by Generative Pre-Training. In.

RAJAN, S. *Build Web App instantly for Machine Learning using Streamlit*. Analytics Vidhya. 2021. Disponível em: <https://www.analyticsvidhya.com/blog/2021/06/build-web-app-instantly-for-machine-learning-using-streamlit/>. Acesso em: 22 mar. 2022.

RAMOS, L. *Should I Choose a Rule-Based or an AI Hotel Chatbot? Here's the Difference*. ReviewPro. 2020. Disponível em: <https://www.reviewpro.com/blog/should-i-choose-a-rule-based-or-an-ai-chatbot/>. Acesso em: 7 mai. 2022.

REIMERS, N.; GUREVYCH, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In: PROCEEDINGS of the 2019 Conference on Empirical Methods in Natural Language Processing. [S.l.]: Association for Computational Linguistics, nov. 2019. Disponível em: <https://arxiv.org/abs/1908.10084>.

RÖSCH, D. *Confidence Score / Confidence Level*. BOTfriends. Disponível em: <https://botfriends.de/en/blog/botwiki/confidence-score/>. Acesso em: 7 mai. 2022.

RUDER, S. et al. Transfer Learning in Natural Language Processing. In: PROCEEDINGS of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials. Minneapolis, Minnesota: Association for Computational Linguistics, jun. 2019. p. 15–18. DOI: 10.18653/v1/N19-5004. Disponível em: <https://aclanthology.org/N19-5004>.

SIEG, A. *FROM Pre-trained Word Embeddings TO Pre-trained Language Models Focus on BERT*: FROM Static Word Embedding TO Dynamic (Contextualized) Word Embedding. Towards Data Science. 2019. Disponível em: <https://towardsdatascience.com/from-pre-trained-word-embeddings-to-pre-trained-language-models-focus-on-bert-343815627598>. Acesso em: 27 fev. 2022.

SOUZA, F.; NOGUEIRA, R.; LOTUFO, R. BERTimbau: Pretrained BERT Models for Brazilian Portuguese. In: [s.l.: s.n.], out. 2020. p. 403–417. ISBN 978-3-030-61376-1. DOI: 10.1007/978-3-030-61377-8_28.

VALLERIAN, A. *What Is Metric?* Understanding metric for data scientist. Towards Data Science. 2021. Disponível em: <https://towardsdatascience.com/what-is-metric-74b0bf6e862>. Acesso em: 19 mar. 2022.

VASWANI, A. et al. *Attention Is All You Need*. [S.l.]: arXiv, 2017. DOI: 10.48550/ARXIV.1706.03762. Disponível em: <https://arxiv.org/abs/1706.03762>.

WU, Y. et al. *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. [S.l.: s.n.], 2016. arXiv: 1609.08144 [cs.CL].

APÊNDICE A - AGRUPAMENTOS OBTIDOS

As Tabelas 6 a 13 exibem alguns dos agrupamentos obtidos pela solução desenvolvida. O nome da instituição financeira que disponibilizou os dados e outras informações que poderiam ser específicas a determinados grupos de usuários foram mascarados pelo termo "ABC".

Tabela 6: *Agrupamento 8219. Número de sentenças: 6. Distância interna: 22%. Similaridade interna: 86%.*

Sentenças
Gostaria de financiar um carro
Quero financiar um carro
Gostaria de fazer um financiamento de automóvel
Eu quero fazer o financiamento de um veículo
Quero financiar meu próprio carro
Já tenho financiamento de veículos

Fonte: **O Autor**

Tabela 7: *Agrupamento 7391. Número de sentenças: 6. Distância interna: 17%. Similaridade interna: 86%.*

Sentenças
Foi realizada uma compra no meu cartão de credito e nao fui eu
Fizeram uma compra em meu cartão de crédito mas não fui eu
Foi realizado uma compra no meu cartão de credito e nao fui eu quem comprou
Foi feito uma compra no meu cartão sem minha autorização
A compra não foi feita por mim, mas foi feita no meu cartão
Estou com compras no meu cartão de crédito que não foram feitas por mim

Fonte: **O Autor**

Tabela 8: *Agrupamento 7637. Número de sentenças: 12. Distância interna: 19%. Similaridade interna: 88%.*

Sentenças
Quero adiantar o pagamento da fatura do cartão de crédito
Gostaria de antecipar pagamento da fatura do cartão de crédito
Quero adiantar o pagamento da fatura de cartão de crédito
Eu quero a fatura do cartão de crédito
Quero antecipar o pagamento do cartão de credito
Eu preciso da fatura do cartão de crédito atual
Eu preciso que faz o pagamento do cartão de credito antecipado
Oi fazer o pagamento antecipado do cartão de crédito
Precisava do boleto da fatura do meu cartão de crédito
Geralmente a fatura do meu cartão de crédito faz débito automático
Manda pra mim a fatura atual do cartão de crédito
Preciso antecipar a fatura do meu cartão de crédito para o dia ABC

Fonte: **O Autor**

Tabela 9: *Agrupamento 8897. Número de sentenças: 11. Distância interna: 27%. Similaridade interna: 82%.*

Sentenças
Queria saber o saldo para quitação do meu consignado
Tô devendo o cheque especial e quero saber o valor para poder pagar
Estou devendo o limite do cheque especial quero saber a possibilidade de tar pagando
Quero saber meu saldo devedor
Tô devendo o limite do cheque especial aí queria saber se a possibilidade de tar quitando essa dívida
Sobre o valor do empréstimo que eu tô devendo gostaria de saber qual o valor
Eu queria sabe quantas parcela estou devendo do empréstimo do mês de ABC
Eu quero o saldo devedor do cartão
Bom dia é gostaria de quitar meu empréstimo ABC feito no aplicativo
Quero saber quanto falta pra quitação do meu conseguinado
Eu quero quitar emprestimo feito no aplicativo

Fonte: **O Autor**

Tabela 10: *Agrupamento 8036. Número de sentenças: 15. Distância interna: 21%. Similaridade interna: 86%.*

Sentenças
Gostaria saber como que eu faço pra adquirir um cartão de crédito
Quero ver como faço pra pedir um cartão de crédito
Quero saber como faço pra obter a função crédito no meu cartão.
Quero saber como faço pra obter a função crédito no meu cartão
Como faço pra pedir um cartão de crédito
Eu queria saber como que eu faço para abrir um cartão de crédito
Simples.. quero saber como faço pra obter a função crédito no meu cartão?
Como faço para ter um cartão de crédito
Como é que faço pra obter a função de crédito no meu cartão da ABC .?
Gostaria de saber como que eu pego a senha do cartão pelo aplicativo ABC
Eu queria ver como fasso para ter um cartão de crédito
Bom dia queria ver se tenho crédito aprovado com vcs pra cartão de crédito
Quero saber si comsigo um cartão de crédito
Gostaria de saber o melhor dia de compra no meu cartao de crédito
Eu quero saber se ja çhegou meu cartao de credito

Fonte: **O Autor**

Tabela 11: *Agrupamento 8585. Número de sentenças: 16. Distância interna: 25%. Similaridade interna: 84%.*

Sentenças
Troquei de celular e estou com dificuldade de pagamentos pelo aplicativo
Mudei de celular nao consigo paga minhas contas pelo aplicativo
Troquei de celular e não consigo mais fazer transferências pelo aplicativo
Troquei de aparelho não consigo pagar pelo aplicativo
Troquei de aparelho de telefone. e gostaria que liberasse meu aplicativo
Mudei de chip não consigo paga minhas conta pelo aplicativo
Eu utilizo e não troquei de celular
Desbloqueio do meu dispositivo pois troquei de celular
Já uso o aplicativo e não troquei de celular
Meu dispositivo está bloqueado para usuário ,mas nao troquei de aparelho
Quero recarregar o meu telefone mas não consigo eu troquei de aparelho
Troquei de aparelho de celular e está bloqueado para pagamento e transferências como faço
Mudei de chip no meu celular preciso do código pra paga minhas conta no aplicativo
Troquei de celular é gostaria de configurar pra fazer transferência
Já utilizo o aplicativo ABC, mas troquei de celular.
Troquei de aparelho e preciso código pra fazer pagamentos

Fonte: **O Autor**

Tabela 12: *Agrupamento 7791. Número de sentenças: 11. Distância interna: 20%. Similaridade interna: 88%.*

Sentenças
Preciso bloquear meu cartão
Quero bloquear meu cartão
Perdi meu cartão e preciso bloquear
Perdi meu cartão preciso bloquear
Gostaria de bloquear meu cartão
Gostaria de bloquear meu cartão por fraude
Preciso bloquear meu cartão por perca ou roubo
Como bloquear meu cartão?
Quero bloquear meu cartão ...perdi ele
Preciso bloquear meu cartão e acionar um novo cartão.
Preciso bloquear o cartão por perca oy roubo

Fonte: **O Autor**

Tabela 13: *Agrupamento 8961. Número de sentenças: 16. Distância interna: 28%. Similaridade interna: 82%.*

Sentenças
Recebi um pix mas não entrou na minha conta
Fiz um pix e não entrou na conta da pessoa
Foi feito pix na minha conta e não caiu
Foi feito um pix na minha conta e não caiu o valor
Fizeram um pix para mim só que não consta na minha conta
Foi feito pix na minha conta e não foi compensada
Recebi uma transferência via pix mas não entrou o valor na conta
Eu mandei um pix para conta do meu avô aki do ABC e o valor não caiu na conta
Não caiu o pix na minha conta
Fiz um pix e o dinheiro não entrou na conta do destinatário
É não chegou na conta que eu passei o pix
Fiz um pix outro pra outro banco. foi descontado mas não caiu na conta
Porque não entrou na minha conta uma transferência pelo pix?
O valor de ABC reais não entrou na outra conta, e não voltou
Meu cartão que passei na máquina ABC ainda n caiu na minha conta
Se entrou ABC na minha conta?

Fonte: **O Autor**