UNIVERSIDADE FEDERAL DE SÃO CARLOS
CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO

Lucas Henrique Marchiori

Agrupamento de embeddings: Análise exploratória de datasets textuais

São Carlos
2025

Lucas Henrique Marchiori

Agrupamento de embeddings: Análise exploratória de datasets textuais

Trabalho de Conclusão de Curso de Graduação em Engenharia de Computação do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de São Carlos como requisito para a obtenção do título de Engenheira de Computação.
Orientador: Prof. Orientador, Dr. Murilo Coelho Naldi

São Carlos
2025

AGRADECIMENTOS

Agradeço aos meus pais por serem minha rede de apoio e me incentivarem ao longo da minha graduação, não desistindo de mim

Ao meu irmão por trazer leveza a vida em forma de risadas.

Às minhas Avós, por todo o amor.

Às minhas amizades que me acompanham desde minha infância, especialmente Vitória e Yasmim, por suportar todos meus momentos de estresse, ansiedade e esgotamento, e por todos os conselhos e ajudas.

Aos amigos que fiz durante a graduação, especialmente Isadora, Maria Rita, Carol, Bianca, Marcella, Jhon, Vinicius, Vitor, Caique, Juan e Kleber, vocês tornaram minha experiência universitária infinitamente mais leve e mais divertida.

Ao meu orientador professor Murilo, por me acompanhar neste trabalho.

E por fim, a Universidade Federal de São Carlos e todos seus colaboradores, por me assegurarem um ensino de altissima qualidade e experiências relevantes durante estes 7 anos.

“Não importa o que aconteça, continue em frente. Não pare. Mesmo quando se sentir perdido. Mesmo quando estiver com medo. Levante sua cabeça e continue em frente.”

— Impa

RESUMO

A seleção inadequada de algoritmos de agrupamento para embeddings — técnicas de representação vetorial de palavras que capturam relações semânticas em espaços multidimensionais — pode comprometer significativamente a qualidade da descoberta de padrões semânticos, resultando em agrupamentos com baixa coesão interna e separação deficiente entre grupos. A ausência de diretrizes claras força pesquisadores a extensivos processos de tentativa e erro, consumindo recursos computacionais e tempo, o que pode levar à perda de desempenho em tarefas subsequentes de Processamento de Linguagem Natural. Neste contexto, este trabalho investiga o desempenho de quatro algoritmos de agrupamento (K-Means, Self-Organizing Maps (SOM), HDBSCAN e Agrupamento Hierárquico Aglomerativo (HCA)) aplicados a word embeddings gerados pelos modelos Word2Vec e SBERT sobre dez conjuntos de dados de domínios distintos. A qualidade dos agrupamentos foi mensurada por métricas quantitativas, como o Índice de Silhueta, o Índice de Davies-Bouldin (DBI) , Density-Based Clustering Validation Index (DBCV) e o Adjusted Rand Index(ARI) . Os resultados revelaram que não existe uma combinação universalmente superior, mas sim padrões sistemáticos que permitem uma predição científica da metodologia ótima. A combinação HDBSCAN + Word2Vec emergiu como dominante em 66% dos casos, mostrando-se especialmente eficaz em domínios com vocabulário padronizado ou redundância informativa estruturada, como em Reuters-21578 e Steam Games. Em contrapartida, a combinação HDBSCAN + SBERT foi superior em domínios criativos e semanticamente heterogêneos, como Spotify e Amazon Reviews, que exigem maior compreensão contextual. O estudo estabelece que o tipo de especialização do domínio textual — criativa ou técnica — determina o embedding ótimo. Adicionalmente, o HDBSCAN consolidou-se como o algoritmo mais versátil, apresentando o melhor desempenho em 72% dos cenários analisados, notadamente por sua robustez em lidar com ruído e agrupamentos de densidade variável, características intrínsecas a dados textuais do mundo real.

Palavras-chave: Algoritmos de Agrupamento; Word Embeddings; Processamento de Linguagem Natural

ABSTRACT

The inadequate selection of clustering algorithms for word embeddings—vector representation techniques that capture semantic relationships in multidimensional spaces—can significantly compromise the quality of semantic pattern discovery, resulting in clusters with low internal cohesion and poor separation. The absence of clear guidelines forces researchers into extensive trial-and-error processes, consuming computational resources and time, which can lead to performance degradation in subsequent Natural Language Processing tasks. In this context, this work investigates the performance of four clustering algorithms (K-Means, Self-Organizing Maps (SOM), HDBSCAN, and Agglomerative Hierarchical Clustering (AHC)) applied to word embeddings generated by the Word2Vec and SBERT models across ten datasets from distinct domains. The quality of the clusters was measured by quantitative metrics, such as the Silhouette Score, the Davies-Bouldin Index (DBI),the Density-Based Clustering Validation Index (DBCV), and the Adjusted Rand Index(ARI). The results revealed that no universally superior combination exists, but rather systematic patterns that allow for a scientific prediction of the optimal methodology. The HDBSCAN + Word2Vec combination emerged as dominant in 66% of cases, proving especially effective in domains with standardized vocabulary or structured informational redundancy, such as in Reuters-21578 and Steam Games. Conversely, the HDBSCAN + SBERT combination was superior in creative and semantically heterogeneous domains, such as Spotify and Amazon Reviews, which require greater contextual understanding. The study establishes that the type of textual domain specialization—creative or technical—determines the optimal embedding. Additionally, HDBSCAN established itself as the most versatile algorithm, achieving the best performance in 72% of the analyzed scenarios, notably for its robustness in handling noise and variable-density clusters, which are intrinsic characteristics of real-world textual data.

Keywords: Clustering Algorithms; Word Embeddings; Natural Language Processing

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AMI	Adjusted Mutual Information (Informação Mútua Ajustada)
ARI	Adjusted Rand Index (Índice Rand Ajustado)
BERT	Bidirectional Encoder Representations from Transformers (Representações de Codificador Bidirecional de Transformers)
BERTimbau	BERT model for Brazilian Portuguese (Modelo BERT para Português Brasileiro)
BMU	Best Matching Unit (Unidade de Melhor Correspondência)
BOWE	Bag-of-Word-Embeddings
CBOW	Continuous Bag-of-Words
DBCV	Density-Based Clustering Validation (Validação de Agrupamento Baseada em Densidade)
DBI	Davies-Bouldin Index (Índice de Davies-Bouldin)
DBSCAN	Density-Based Spatial Clustering of Applications with Noise (Agrupamento Espacial Baseado em Densidade de Aplicações com Ruído)
ELMo	Embeddings from Language Model (Embeddings de Modelo de Linguagem)
EOM	Excess of Mass (Excesso de Massa)
HCA	Hierarchical Agglomerative Clustering (Agrupamento Hierárquico Aglomerativo)
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise (Agrupamento Espacial Hierárquico Baseado em Densidade de Aplicações com Ruído)
MLM	Masked Language Modeling (Modelagem de Linguagem Mascarada)
MPNet	Masked and Permuted Language Modeling (Modelagem de Linguagem Mascarada e Permutada)
MST	Minimum Spanning Tree (Árvore Mínima de Expansão)
NLTK	Natural Language Toolkit
NSP	Next Sentence Prediction (Predição de Próxima Sentença)
PLN	Processamento de Linguagem Natural
RoBERTa	Robustly Optimized BERT Pretraining Approach (Abordagem de Pré-treinamento BERT Robustamente Otimizada)
SBERT	Sentence-BERT (BERT para Sentenças)
SOM	Self-Organizing Maps (Mapas Auto-Organizáveis)
TF-IDF	Term Frequency-Inverse Document Frequency (Frequência de Termo - Frequência Inversa de Documento)
WCSS	Within-Cluster Sum of Squares (Soma de Quadrados Intra-Grupos)

SUMÁRIO

1 INTRODUÇÃO

As *embeddings* têm-se consolidado como uma abordagem eficaz para a representação de palavras em espaços vetoriais multidimensionais, possibilitando a identificação de semelhanças semânticas com base no contexto. Contudo, o processamento desses dados, especialmente em cenários com volumes elevados e pela alta dimensionalidade, impõe desafios relacionados à eficiência computacional. Uma estratégia para mitigar essa ineficiência é a aplicação de técnicas de agrupamento que segmentam grandes conjuntos de dados em subconjuntos menores e mais homogêneos, reduzindo o custo computacional e potencializando as análises.

Diversos estudos têm explorado o agrupamento das *embeddings* com o objetivo de melhorar o desempenho em tarefas de *data mining*, de classificação e de recomendação. Por exemplo, (MEHTA; BAWA; SINGH, 2021) demonstrou a eficácia do agrupamento na segmentação de artigos semelhantes; (SHAFFER, 2021) e (TAN *et al.*, 2019) evidenciaram os ganhos do agrupamento em cenários multilíngues, nos quais a redução do *corpus* facilita o processamento simultâneo de várias línguas. Em tarefas de classificação, o agrupamento também tem sido empregado para aprimorar o desempenho dos modelos, como discutido em (REIMERS *et al.*, 2019).

No campo de PLN (Processamento de Linguagem Natural), o agrupamento de palavras é aplicado em diversas frentes. Em trabalhos como o de (**tache-etal-2ladi-clustering**), modelos como o *Word2Vec*(MIKOLOV *et al.*, 2013) foram utilizados para construir uma BOWE (Bag-of-Word-Embeddings), na qual cada documento é representado por um histograma resultante da aplicação do *K-Means*(LLOYD, 1982) (MACQUEEN, 1967) e do SOM (Self-Organizing Maps (Mapas Auto-Organizáveis))(KOHONEN, 1990). Essa abordagem evidencia a necessidade de alternativas ao *K-Means* em cenários com grandes variações de densidade entre os grupos. Por sua vez, (REIMERS *et al.*, 2019) explorou o agrupamento hierárquico para segmentar textos argumentativos, demonstrando que a técnica pode facilitar a aplicação de métodos de classificação ao agrupar conteúdos semanticamente semelhantes.

Este trabalho propõe investigar o desempenho de diferentes algoritmos de agrupamento aplicados às *embeddings*, implementando as técnicas *K-Means*, SOM, HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise (Agrupamento Espacial Hierárquico Baseado em Densidade de Aplicações com Ruído)) e o HCA (Hierarchical Agglomerative Clustering (Agrupamento Hierárquico Aglomerativo)). A avaliação será realizada de forma quantitativa, utilizando métricas como o DBI (Davies-Bouldin Index (Índice de Davies-Bouldin)), o Índice de Silhueta, o DBCV (Density-Based Clustering Validation (Validação de Agrupamento Baseada em Densidade)) e o ARI (Adjusted Rand Index (Índice Rand Ajustado)) quando rótulos de referência estiverem disponíveis. O objetivo é identificar a técnica mais eficaz para representar semelhanças semânticas em dados vetoriais de palavras, contribuindo para o avanço das aplicações de agrupamento em PLN.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver, implementar e avaliar diferentes algoritmos de agrupamento aplicados às *embeddings*, visando identificar os métodos mais eficazes para agrupar palavras em tarefas relacionadas a PLN. O estudo busca explorar a capacidade dos algoritmos em capturar relações semânticas e contextuais entre palavras, considerando diferentes métricas de desempenho, cenários de dados e configurações de hiperparâmetros.

1.1.2 Objetivos Específicos

- Investigar diferentes modelos de geração de *embeddings*, como SBERT (Sentence-BERT (BERT para Sentenças))(REIMERS; GUREVYCH, 2019b) e *Word2Vec*(MIKOLOV *et al.*,

2013), avaliando suas características e adequação para tarefas de clusterização.

- Implementar e avaliar algoritmos de agrupamento, incluindo *K-Means*, SOM(KOHONEN, 1990), HDBSCAN(MCINNES; HEALY; ASTELS, 2017) e HCA)(SOKAL; SNEATH, 1963)
- Examinar o desempenho dos algoritmos de agrupamento utilizando métricas quantitativas, como o DBI, Índice de Silhueta, o DBCV e o ARI, para medir a qualidade dos agrupamentos gerados.
- Comparar os resultados em diferentes cenários de dados e configurações de hiper parâmetros, consolidando as análises em uma avaliação.
- Identificar os algoritmos e configurações que apresentam o melhor desempenho em termos de coesão interna e separação entre grupos para os algoritmos baseados em centroides e, para algoritmos baseados em densidade, identificar aqueles com melhor desempenho em termos de capacidade de detectar agrupamentos de formatos arbitrários e lidar com ruídos.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 WORD EMBEDDINGS E DOCUMENT EMBEDDINGS

As *word embeddings* são uma forma de representação de palavras em que os termos são mapeados para vetores numéricos densos de dimensionalidade fixa, capazes de capturar relações semânticas e sintáticas entre si. Esses vetores permitem que termos semanticamente semelhantes sejam representados por vetores próximos no espaço vetorial, possibilitando operações matemáticas que preservam relações semânticas.

Expandindo este conceito, *document embeddings* representam sequências completas de texto (frases, parágrafos ou documentos inteiros) como vetores únicos, capturando a semântica global do conteúdo textual. Esta abordagem é fundamental para tarefas que requerem a comparação ou agrupamento de textos completos, como exemplificado pela figura 1 ao representar vetores fictícios para documentos completos como *Catching Fire* e *The Catcher in the Rye*.

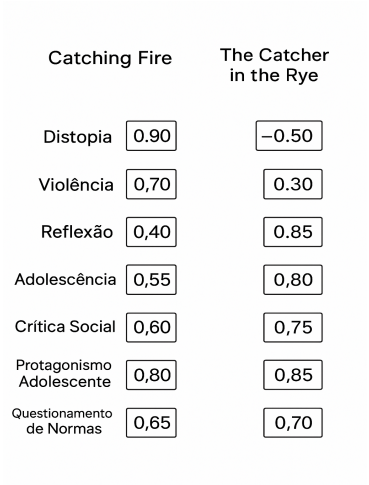


Figura 1 – Exemplo de vetores para documentos

Por meio dessas representações vetoriais, é possível encontrar textos semelhantes com base em seu conteúdo semântico, e atualmente existem diferentes abordagens para transformar textos em representações numéricas de maneira autônoma. Este trabalho explora duas estratégias principais: a geração de *word embeddings* seguida da agregação para formar *document embeddings*, utilizando o algoritmo *Word2Vec*(MIKOLOV *et al.*, 2013), e a geração direta de *document embeddings* utilizando modelos baseados em *Transformers* por meio de *Sentence-Transformers*(REIMERS; GUREVYCH, 2019b).

2.1.1 Word2Vec

Word2Vec(MIKOLOV *et al.*, 2013) consiste em treinar uma rede neural rasa, ou seja, de apenas uma camada, que utiliza duas arquiteturas distintas: o modelo CBOW e o modelo *Skip-Gram*.

No CBOW, as palavras que aparecem no contexto ao redor de uma palavra central são combinadas por meio de uma média ou soma, e o modelo é treinado para prever a palavra central a partir dessa combinação.

Em contraste, o Skip-Gram inverte essa estratégia, utilizando cada ocorrência de uma palavra central para prever, de forma independente, cada uma das palavras que aparecem em uma janela de contexto definida.

A imagem 2 ilustra esta diferença entre as arquiteturas explicadas.

Uma vez obtidas as representações vetoriais individuais para cada palavra através do *Word2Vec*,

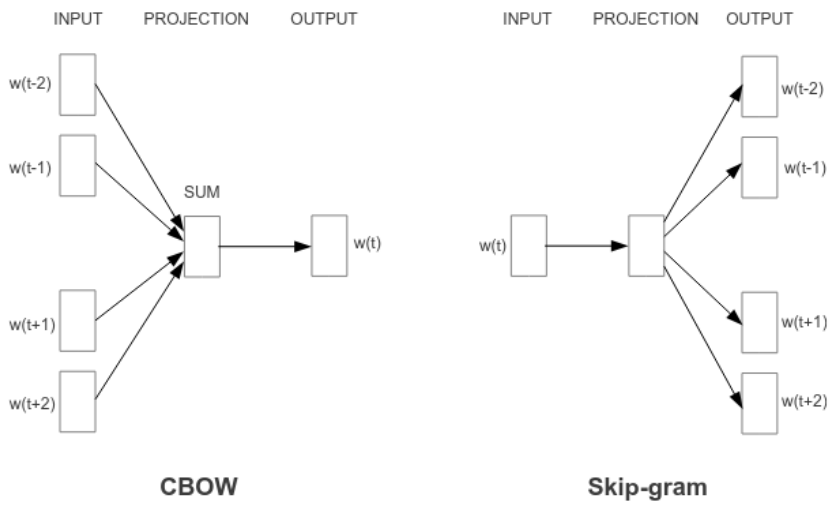


Figura 2 – Ilustração das arquiteturas word2vec(MIKOLOV *et al.*, 2013)

torna-se necessário agregar essas informações para representar documentos completos. A abordagem mais comum consiste no cálculo da média aritmética dos vetores de todas as palavras presentes no documento, resultando em um *document embedding* que preserva as características semânticas globais do texto. Esta estratégia de agregação, embora simples, tem-se mostrado eficaz na literatura por sua capacidade de capturar a semântica geral do documento.

2.1.2 BERT

Os modelos baseados em *Transformer* bidirecionais adotam uma arquitetura composta pelo *encoder*, como explicitado por (DEVLIN *et al.*, 2019). Nesse modelo, o *encoder* é responsável por extrair as características da entrada por meio de camadas de atenção multi-cabeças em paralelo, seguidas por uma rede neural *feed-forward*. A camada de atenção processa todos os *tokens* de uma sequência para gerar um vetor de contexto, permitindo que a representação de cada token incorpore informações dos demais e determinando quais *tokens* recebem maior peso na representação final. Esses escores de atenção são então combinados e encaminhados para a rede *feed-forward*, cujo resultado serve como entrada para a próxima camada de atenção.

Durante seu pré-treinamento, são realizadas duas tarefas semi-supervisionadas: (1) **MLM (Masked Language Modeling (Modelagem de Linguagem Mascarada))**, no qual alguns *tokens* da sequência são substituídos por um *token* especial (por exemplo, [MASK]) e o modelo é treinado para prever os *tokens* ausentes a partir do contexto bidirecional; e (2) **NSP (Next Sentence Prediction (Predição de Próxima Sentença))**, na qual o modelo recebe um par de sentenças e deve identificar se a segunda sentença sucede à primeira.

A entrada do BERT consiste em uma sequência de *tokens* para a qual são atribuídas três formas de *embeddings*: o *token embedding* (resultante da tokenização, incluindo *tokens* especiais como [CLS] e [SEP]), o *segment embedding* (utilizado para distinguir diferentes sentenças) e o *position embedding* (que indica a posição de cada *token* na sequência). A soma desses *embeddings* compõe a representação final de entrada do modelo, como ilustrado na figura 3.

Em contraste com métodos mais simples, como o *Word2Vec*, que utiliza uma rede neural rasa com uma única camada de projeção para gerar *embeddings* estáticos (um único vetor para cada palavra,

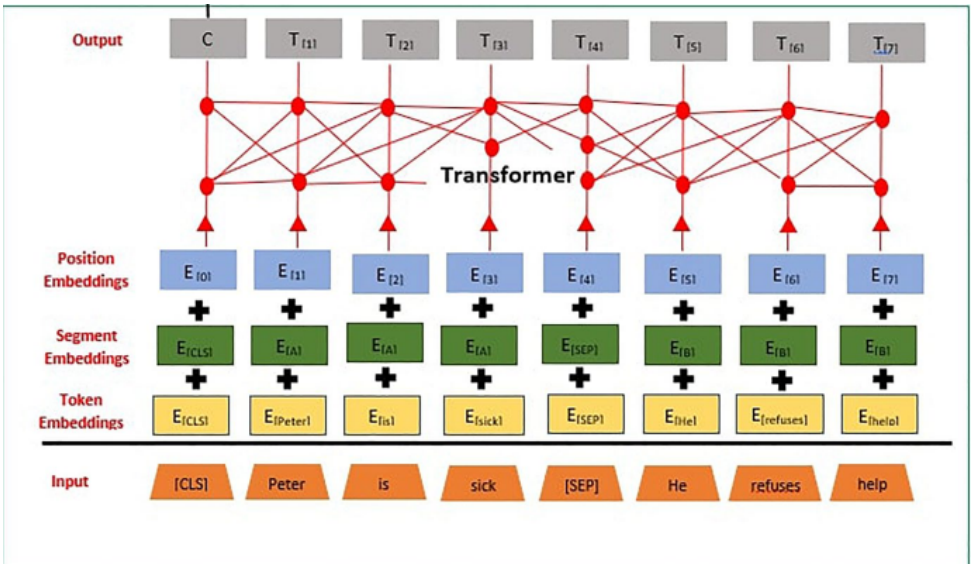


Figura 3 – Arquitetura do BERT(VITHANAGE *et al.*, 2024)

independentemente do contexto), o BERT produz representações dinâmicas que variam conforme o contexto de ocorrência do *token*. Enquanto o *Word2Vec* utiliza arquiteturas como o CBOW, que prevê a palavra central a partir das palavras do contexto, ou o *Skip-Gram*, que, dado um *token* central, prevê cada palavra do contexto de forma independente, o BERT emprega múltiplas camadas de autoatenção para processar toda a sequência de uma só vez. Essa característica permite que o BERT capture nuances contextuais mais refinadas.

2.1.3 Sentence-Transformers

Os *Sentence-Transformers*(REIMERS; GUREVYCH, 2019a) representam uma evolução dos modelos BERT, especificamente projetados para gerar representações semânticas de alta qualidade para sequências completas de texto. Enquanto o BERT tradicional é otimizado para tarefas de predição de *tokens* individuais, os *Sentence-Transformers* são treinados especificamente para produzir *embeddings* que preservam a similaridade semântica entre textos.

A arquitetura dos *Sentence-Transformers* baseia-se em modelos *Transformer* pré-treinados (como BERT), que são refinados utilizando técnicas de aprendizado siamês e de *triplet loss*. Durante o treinamento, o modelo aprende a mapear textos semanticamente semelhantes para regiões próximas no espaço vetorial, enquanto textos semanticamente dissimilares são mapeados para regiões distantes.

O processo de geração de *embeddings* com os *Sentence-Transformers* elimina a necessidade de estratégias de agregação complexas. O modelo processa toda a sequência textual e aplica *pooling* às representações contextuais dos *tokens*, produzindo um único vetor de dimensão fixa que representa o documento.

Conforme descrito em (REIMERS; GUREVYCH, 2019a), existem diferentes estratégias de *pooling* que podem ser aplicadas aos *embeddings* de *tokens* gerados pelo modelo *Transformer*:

Mean Pooling (Pooling por Média): A estratégia mais comumente utilizada, onde o *embedding* final do documento é calculado como a média aritmética de todos os *embeddings* de *tokens*. Matematicamente, para uma sequência de *tokens* $\{t_1, t_2, ..., t_n\}$ com suas respectivas representações vetoriais $\{h_1, h_2, ..., h_n\}$, o *embedding* final u é calculado como:

$$u = \frac{1}{n} \sum_{i=1}^{n} h_i \tag{1}$$

onde $h_i \in \mathbb{R}^d$ representa o vetor de *embedding* do *token* t_i e d é a dimensionalidade dos *embeddings*.

Max Pooling: Nesta estratégia, cada dimensão do vetor final é determinada pelo valor máximo encontrado naquela dimensão entre todos os *tokens*:

$$u_j = \max_{i=1}^{n} h_{i,j} \tag{2}$$

onde u_j é a j-ésima dimensão do vetor final e $h_{i,j}$ é a j-ésima dimensão do *embedding* do *token* t_i.

CLS Pooling: Utiliza especificamente o *embedding* do *token* especial `[CLS]` que é adicionado no início de cada sequência durante o pré-processamento do BERT. Este *token* é projetado para capturar informações globais da sequência:

$$u = h_{[CLS]} \tag{3}$$

Uma vantagem significativa dos *Sentence-Transformers* é sua capacidade multilíngue. Modelos como o *paraphrase-multilingual-MiniLM-L12-v2* são treinados em múltiplos idiomas simultaneamente, permitindo a geração de *embeddings* consistentes para textos em diferentes línguas. Isso é particularmente relevante para conjunto de dados internacionais que contêm conteúdo em múltiplos idiomas, como nomes de artistas, títulos de obras e descrições de produtos de diferentes regiões.

A figura 4 ilustra a arquitetura do *Sentence-Transformers*

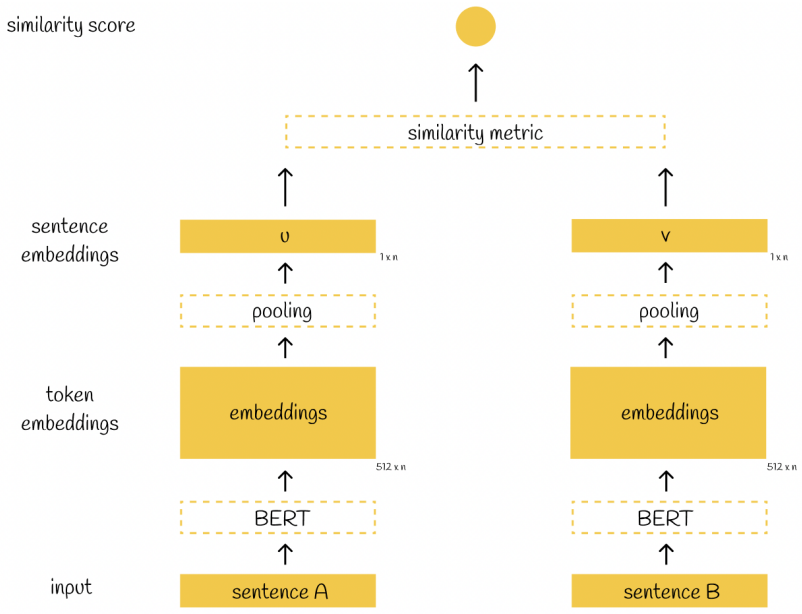


Figura 4 – Arquitetura *Sentence-Transformers*[1]

2.2 SISTEMAS DE AGRUPAMENTO

2.2.1 Fundamentação teórica K-means

O *k-means*(LLOYD, 1982)(MACQUEEN, 1967) é um método amplamente utilizado para agrupamento não supervisionado, cujo objetivo é particionar um conjunto de dados em k grupos de forma que cada dado pertença ao grupo com o centroide (centro) mais próximo, minimizando a soma das distâncias quadráticas entre os dados e os respectivos centroides. Formalmente, dado um conjunto de pontos $\{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^d$, o *k-means* procura encontrar k centroides $\{\mu_1, \mu_2, \ldots, \mu_k\}$ que minimizem a função objetivo

$$J = \sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \|x_i - \mu_j\|^2,$$

onde $\|x_i - \mu_j\|$ representa a distância euclidiana entre o ponto x_i e o centroide μ_j.

A distância euclidiana é a métrica mais utilizada no *k-means*, definida para dois pontos $x = (x_1, x_2, \ldots, x_d)$ e $\mu = (\mu_1, \mu_2, \ldots, \mu_d)$ em $\mathbb{R}^d$ por

$$\|x - \mu\| = \sqrt{\sum_{l=1}^{d} (x_l - \mu_l)^2}\,.$$

O *k-means* é normalmente implementado em duas etapas principais de forma iterativa:

1. **Atribuição:** Cada ponto x_i é atribuído ao grupo cujo centroide μ_j está mais próximo, ou seja, para o qual $\|x_i - \mu_j\|$ é mínimo.
2. **Atualização:** Para cada grupo, o novo centroide é calculado como a média de todos os pontos atribuídos a ele:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i\,,$$

onde C_j é o conjunto de pontos atribuídos ao grupo j e $|C_j|$ é o número de pontos nele contido.

Essas etapas são repetidas até que as atribuições não mudem mais ou que uma condição de parada seja atingida.

O método *k-means++*(ARTHUR; VASSILVITSKII, 2007) foi proposto para melhorar essa etapa de inicialização. Em vez de escolher os centroides de forma completamente aleatória, o *k-means++* seleciona o primeiro centroide aleatoriamente e, para cada centroide subsequente, escolhe um ponto x do conjunto de dados com probabilidade proporcional à sua distância quadrática mínima em relação aos centroides já escolhidos. Formalmente, após escolher os centroides iniciais, o próximo centroide μ é escolhido de modo que a probabilidade de selecionar um ponto x seja

$$P(x) = \frac{D(x')^2}{\sum_{x \in X} D(x)^2}\,,$$

onde $D(x)$ é a menor distância de x a qualquer centroide já selecionado. Essa estratégia tende a dispersar os centroides iniciais, contribuindo para uma convergência mais rápida e para a obtenção de melhores agrupamentos.

2.2.2 Fundamentação teórica HDBSCAN

HDBSCAN(CAMPELLO, Ricardo J. G. B.; MOULAVI; SANDER, 2013) é um algoritmo de agrupamento baseado em densidade que expande os métodos tradicionais, constrói uma hierarquia de grupos e, em seguida, extrai uma partição plana baseada na estabilidade dos grupos.

Ao contrário do *k-means*, que particiona os dados em k grupos pré-definidos e utiliza a distância euclidiana para definir a similaridade, o HDBSCAN não requer a especificação do número de grupos e é capaz de identificar grupos de forma arbitrária, além de detectar ruídos.

O algoritmo HDBSCAN funciona basicamente em duas etapas:

1. **Construção da Hierarquia:** Inicialmente, o HDBSCAN transforma as distâncias entre os pontos em uma medida de "distância de alcance"que incorpora a densidade local. Em seguida, constrói uma árvore mínima de expansão(MST (Minimum Spanning Tree (Árvore Mínima de Expansão))) sobre esses pontos e utiliza a variação dessa medida para construir uma hierarquia de grupos.
2. **Extração de grupos:** A partir da hierarquia, o HDBSCANseleciona os grupos mais estáveis — isto é, aqueles que persistem por um amplo intervalo de densidade — e designa os pontos que não pertencem a grupos estáveis como ruído.

E para a realização desta hierarquia e extração de grupos, o algoritmo utiliza-se da distância de alcance (*core distance*)definida como a distância entre dois pontos p e q em um espaço de alta dimensão, onde k é o número de vizinhos mais próximos de p, definido pelo tamanho mínimo do grupo. A distância de alcance é dada por:

$$d_{\text{core}}(p) = \text{distância de } p \text{ ao seu } k^{\text{th}} \text{ vizinho mais próximo}\,.$$

A partir dessa medida, define-se a **distância de alcance mútua** entre dois pontos p e q como:

$$d_{\text{mrd}}(p,q) = \max\{d_{\text{core}}(p),\, d_{\text{core}}(q),\, d(p,q)\}\,,$$

Utilizando os valores de d_{mrd}, constrói-se uma Árvore Geradora Mínima (MST) sobre o grafo dos pontos, onde cada aresta tem peso dado por $d_{\text{mrd}}(p,q)$. Em seguida, o algoritmo gera uma hierarquia de grupos, removendo progressivamente as arestas da MST em ordem crescente de peso, o que equivale a aumentar um limiar de densidade.

Após ter gerado a árvore hierárquica através da MST é aplicado o *Framework for Optimal Selection of Clusters*(CAMPELLO, R. J. G. B. *et al.*, 2013) para transformar a estrutura hierárquica em um conjunto plano de grupos que melhor representa os dados. o *Framework*(CAMPELLO, R. J. G. B. *et al.*, 2013) inicia-se esta conversão definindo uma variável de densidade inversa:

$$\lambda = \frac{1}{d}\,,$$

De forma que regiões de alta densidade correspondem a valores elevados de λ. Na hierarquia, cada grupo C surge em um determinado nível de densidade, denominado $\lambda_{\text{birth}}(C)$, e "morre"quando a densidade cai abaixo de um limiar $\lambda_{\text{death}}(C)$.

A **estabilidade** de um grupo é calculada somando-se, para cada ponto p pertencente ao grupo, a diferença entre o valor de λ no qual p deixa o grupo e o nível de nascimento do grupo:

$$\text{Stabilidade}(C) = \sum_{p \in C} \Big(\lambda(p) - \lambda_{\text{birth}}(C)\Big)\,,$$

onde $\lambda(p)$ representa o valor de λ em que o ponto p sai do grupo C. grupos com maior estabilidade são considerados mais robustos e são preservados na extração final da partição, enquanto pontos que não se enquadram em grupos estáveis são rotulados como ruído.

Diferente do *k-means*, que assume grupos convexos e de forma aproximadamente esférica, o HDBSCAN pode identificar grupos com formas arbitrárias e é menos sensível à inicialização, pois não depende de centroides pré-definidos.

2.2.3 Fundamentação teórica SOM

O SOM(KOHONEN, 1990), é uma rede neural não supervisionada utilizada para a redução de dimensionalidade e agrupamento, mantendo a topologia dos dados. Em contraste com algoritmos como *k-means*, que particionam os dados em grupos baseados em centroides sem levar em conta a organização espacial entre os grupos, e com o HDBSCAN, que identifica grupos de forma hierárquica a partir de medidas de densidade, o SOMorganiza os dados em uma grade bidimensional onde cada neurônio é associado a um vetor de pesos representativo de um protótipo dos dados.

No começo do processo, todos os pesos são inicializados com valores aleatórios; após isto, os pesos dos neurônios são ajustados durante o treinamento para refletir as características dos dados de entrada. Para cada entrada $x \in \mathbb{R}^d$, o SOM determina o BMU (Best Matching Unit (Unidade de Melhor Correspondência)) por meio da minimização da distância euclidiana:

$$\text{BMU} = \arg\min_i \|x - w_i\|,$$

onde w_i é o vetor de pesos associado ao neurônio i.

Após a determinação do BMU, os pesos dos neurônios vencedores e dos seus vizinhos são atualizados através da seguinte equação:

$$w_i(t+1) = w_i(t) + \eta(t)\, h_{ci}(t)\, \big(x(t) - w_i(t)\big),$$

onde:

- $\eta(t)$ é a taxa de aprendizagem que decresce com o tempo;
- $h_{ci}(t)$ é a função de vizinhança, tipicamente uma função gaussiana, definida por

$$h_{ci}(t) = \exp\left(-\frac{\|r_c - r_i\|^2}{2\sigma^2(t)}\right),$$

  Na qual r_c e r_i são as posições (em termos de coordenadas da grade) do BMUe do neurônio i, respectivamente, e $\sigma(t)$ é o raio de vizinhança que também decresce ao longo do treinamento, tendendo a zero quando umneurônioo estiver muito próximo do BMU.

O algoritmo termina quando o mapa atinge um estado estável, ou seja, quando as atualizações dos pesos não resultam em mudanças significativas nas posições dos neurônios.

2.2.4 Fundamentação Teórica de HCA

O HCA(SOKAL; SNEATH, 1963) é uma técnica de análise não supervisionada que constrói uma estrutura em árvore, representando a organização dos dados em diferentes níveis de granularidade. Esse método pode ser aplicado de forma aglomerativa (*bottom-up*), iniciando com cada ponto em seu próprio grupo e, iterativamente, fundindo os dois grupos mais próximos, ou de forma divisiva (*top-down*), partindo de um único grupo que contém todos os dados e dividindo-o sucessivamente.

Quando os dados correspondem a *word embeddings*, é comum utilizar a distância do cosseno como métrica de similaridade, pois essa medida é mais adequada para dados que possuem alta dimensionalidade e cujas magnitudes não são tão relevantes quanto a direção dos vetores. A distância do cosseno entre dois vetores x e y é definida por

$$d_{\cos}(x, y) = 1 - \frac{x \cdot y}{\|x\|\,\|y\|},$$

onde $x \cdot y$ representa o produto escalar e $\|x\|$ e $\|y\|$ as normas dos vetores x e y, respectivamente.

No contexto do agrupamento hierárquico aplicado a *word embeddings*, a distância do cosseno é utilizada para medir a similaridade entre os pontos. Para calcular a distância entre dois grupos, empregam-se critérios de ligação (*linkage*) adaptados, tais como:

- **Ligação simples (*single linkage*)**(SNEATH, 1957): A distância entre dois grupos A e B é a mínima distância do cosseno entre quaisquer dois pontos:

$$d_{\text{simples}}(A, B) = \min_{a \in A,\, b \in B} d_{\cos}(a, b)\,.$$

- **Ligação completa (*complete linkage*)**(SOKAL; SNEATH, 1963)(MCQUITTY, 1960):Define-se pela máxima distância do cosseno entre quaisquer dois pontos dos grupos:

$$d_{\text{completa}}(A, B) = \max_{a \in A,\, b \in B} d_{\cos}(a, b)\,.$$

- **Ligação média (*average linkage*)**(SOKAL; MICHENER, 1958): Calcula-se a média das distâncias do cosseno entre todos os pares de pontos, um em cada grupo:

$$d_{\text{media}}(A, B) = \frac{1}{|A||B|} \sum_{a \in A} \sum_{b \in B} d_{\cos}(a, b)\,.$$

A partir dessas medidas, o algoritmo aglomerativo combina os grupos conforme o critério de ligação escolhido, gerando um dendrograma que pode ser "cortado"em diferentes níveis para extrair o número desejado de agrupamentos.

Em comparação, o *k-means* depende de centroides e de uma partição rígida dos dados, o HDBSCAN explora a densidade local para identificar agrupamentos robustos e ruídos, e o SOM organiza os dados em uma grade preservando a topologia sem produzir uma estrutura hierárquica explícita.

2.3 MÉTRICAS DE AVALIAÇÃO

2.3.1 Fundamentação Teórica do índice de silhueta

O índice de Silhueta(ROUSSEEUW, 1987) é uma métrica para a interpretação e validação de resultados de algoritmos de agrupamento de dados. O método oferece uma medida da qualidade do agrupamento ao quantificar quão bem cada objeto se encaixa no seu próprio grupo em comparação com outros grupos. A análise avalia duas características fundamentais de um grupo bem definido: a sua **compactação** (ou coesão), que se refere a quão próximos os pontos estão entre si dentro de um mesmo grupo, e a sua **separação**, que mede o quão distintos ou distantes os grupos estão uns dos outros.

Para cada ponto de dados x_i, o índice de silhueta $s(i)$ é calculado com base em dois valores: $a(i)$ e $b(i)$.

1. **Coesão $a(i)$:** Representa a distância média do ponto x_i para todos os outros pontos no mesmo grupo. Se C_k é o grupo ao qual x_i pertence, então $a(i)$ é definido como:

$$a(i) = \frac{1}{|C_k| - 1} \sum_{x_j \in C_k, i \neq j} d(x_i, x_j)\,,$$

   onde $|C_k|$ é o número de pontos no grupo C_k e $d(x_i, x_j)$ é a distância entre os pontos x_i e x_j. Um valor baixo de $a(i)$ indica que o ponto está bem ajustado ao seu grupo.

2. **Separação $b(i)$:** Representa a menor distância média de x_i para todos os pontos de qualquer outro grupo do qual x_i não é membro. É calculada como:

$$b(i) = \min_{l \neq k} \left\{ \frac{1}{|C_l|} \sum_{x_j \in C_l} d(x_i, x_j) \right\},$$

   onde o mínimo é obtido sobre todos os grupos C_l dos quais x_i não faz parte. Um valor alto de $b(i)$ indica que o ponto está longe dos outros grupos.

Com base em $a(i)$ e $b(i)$, o índice de silhueta para o ponto x_i é formalmente definido pela equação:

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

O valor de $s(i)$ varia no intervalo de $[-1, 1]$, e sua interpretação é a seguinte:

- $s(i) \approx 1$**:** Indica que a distância intra-grupo $a(i)$ é muito menor que a distância do grupo vizinho mais próximo $b(i)$. Isso sugere que o ponto x_i foi corretamente atribuído ao seu grupo.
- $s(i) \approx 0$**:** Indica que $a(i)$ é muito semelhante a $b(i)$, o que significa que o ponto está na fronteira entre dois grupos.
- $s(i) \approx -1$**:** Indica que $a(i)$ é muito maior que $b(i)$, sugerindo que o ponto x_i foi mal classificado e estaria melhor alocado no grupo vizinho.

A silhueta média de todo o conjunto de dados fornece uma avaliação global da validade do agrupamento, e é definida por:

$$\bar{s} = \frac{1}{n} \sum_{i=1}^{n} s(i)\,,$$

onde n é o número total de instâncias no conjunto de dados.

2.3.2 Fundamentação Teórica do DBI

O DBI(DAVIES; BOULDIN, 1979) é uma métrica de validação interna para algoritmos de agrupamento. Sua função é avaliar a qualidade da partição dos dados medindo a relação entre a dispersão dentro dos grupos e a separação entre os grupos. A métrica se baseia na ideia de que um bom agrupamento contém grupos **compactos** (com baixa dispersão interna) e bem **separados** uns dos outros.

O índice calcula, para cada grupo, uma medida de similaridade com os demais, baseada em uma razão entre a dispersão interna e a separação externa. O objetivo é minimizar essa razão. Portanto, ao contrário de métricas como a Silhueta, um valor mais baixo no DBI indica um agrupamento melhor.

Para cada grupo C_k, o cálculo se baseia nos seguintes componentes:

1. **Dispersão Interna (S_k):** Representa a dispersão média dos pontos de um grupo em relação ao seu próprio centroide. É uma medida de quão compacto o grupo é. Para um grupo C_k com T_k pontos, a dispersão Interna S_k é formalmente definida como:

$$S_k = \frac{1}{T_k} \sum_{x \in C_k} ||x - c_k||_p$$

   onde c_k é o centroide do grupo C_k, e $||x - c_k||_p$ é a distância entre o ponto x e o centroide c_k, geralmente a distância Euclidiana (norma com $p = 2$). Um valor baixo de S_k indica uma maior compactação do grupo e consequentemente menor dispersão interna.

2. **Separação entre Grupos ($d(c_k, c_l)$):** Representa a distância entre os centroides de dois grupos distintos, C_k e C_l. Esta medida quantifica o quão separados os grupos estão no espaço de características.

$$d(c_k, c_l) = ||c_k - c_l||_p$$

   Um valor alto de $d(c_k, c_l)$ indica que os grupos estão bem separados.

A partir destes componentes, calcula-se um índice de similaridade R_{kl} entre um par de grupos C_k e C_l. Esta razão contrasta a dispersão interna combinada dos dois grupos com sua separação:

$$R_{kl} = \frac{S_k + S_l}{d(c_k, c_l)}$$

O passo seguinte é, para cada grupo C_k, encontrar o grupo vizinho C_l que maximiza essa razão de similaridade. Este valor é denotado por R_k:

$$R_k = \max_{l \neq k} \{R_{kl}\}$$

Um valor alto de R_k sugere que o grupo C_k é pouco distinto de seu vizinho mais similar, seja por ter uma alta dispersão interna ou por estar muito próximo dele.

DBI é definido como a média aritmética desses valores de "pior cenário"para todos os N grupos:

$$DBI = \frac{1}{N} \sum_{k=1}^{N} R_k$$

O valor de DBI varia no intervalo de $[0, +\infty]$, e sua interpretação é a seguinte:

- **DBI $\approx$ 0:** Indica um agrupamento excelente. Agrupamentos com baixas distâncias intra-grupo (alta dispersão Interna) e altas distâncias inter-grupo (alta separação) resultarão em um baixo DBI.
- **DBI elevado:** Sugere uma partição de baixa qualidade, na qual os grupos não são densos ou se sobrepõem, pois a dispersão interna é grande em relação à separação.

2.3.3 Fundamentação Teórica do DBCV

O DBCV(MOULAVI *et al.*, s.d.) é uma métrica de validação interna projetada especificamente para avaliar algoritmos de agrupamento que identificam grupos de formatos arbitrários e com densidades variáveis, como o o HDBSCAN. Diferentemente de métricas como o DBI, que se baseiam em centroides e assumem uma estrutura esférica para os grupos, o DBCV avalia a qualidade do agrupamento com base na noção de conectividade por densidade.

A construção do índice envolve os seguintes componentes principais:

1. ***Core distance* local (**$\text{core}(o)$**):** para cada objeto o define-se uma medida de escala local que captura a densidade em sua vizinhança. Uma formulação prática utilizada na literatura (e nas implementações dos autores) baseia-se em estatísticas das distâncias aos vizinhos mais próximos; seja $\text{KNN}(o, i)$ a distância do ponto o ao seu i-ésimo vizinho, então uma versão agregada da *core distance* (denotada aqui $\text{core}_{apts}(o)$) pode ser escrita simbolicamente como

$$\text{core}_{apts}(o) \;=\; \left(\frac{1}{T-1} \sum_{i=2}^{T} \left(\frac{1}{\text{KNN}(o,i)} \right)^{d} \right)^{-1/d},$$

   onde T é o número de pontos considerados na vizinhança e d é a dimensionalidade do espaço. Pontos em regiões de maior densidade tendem a apresentar *core distances* menores.

2. ***Mutual reachability distance*** **(**mrd**):** a partir das *core distances* define-se a distância de alcance mútuo entre dois pontos a e b como

$$\text{mrd}(a,b) \;=\; \max\{\, \text{core}(a),\ \text{core}(b),\ d(a,b) \,\},$$

   onde $d(a, b)$ é a distância geométrica, por exemplo, euclidiana). A mrd torna a conectividade sensível a variações locais de densidade, evitando conexões geométricas que atravessam regiões de baixa densidade.

3. **MST em** mrd **e *internal nodes*:** para cada grupo C_i constrói-se a *minimum spanning tree* (MST) usando as distâncias mrd restritas aos pontos de C_i. Identificam-se então os *internal nodes* — pontos do grupo com grau ≥ 2 na MST — que representam a estrutura interna do grupo (grupos triviais / singletons normalmente não contribuem como *internal nodes*).

4. **Densidade-*sparseness*** **(**DSC**) e densidade-separação (**DSPC**):**

   - A *density sparseness* de um grupo C_i, $\text{DSC}(C_i)$, é definida como o maior peso de aresta (i.e., maior mrd) na MST restrita aos *internal nodes* de C_i. Intuitivamente, quanto menor DSC, mais denso/compacto é o grupo em termos de conectividade de densidade.
   - A *density separation* entre dois grupos C_i e C_j, $\text{DSPC}(C_i, C_j)$, é definida como a menor mrd entre quaisquer *internal nodes* de C_i e C_j. Valores maiores de DSPC indicam maior separação por densidade entre as estruturas internas dos grupos.

5. **Validade de um grupo (**$VC(C_i)$**):** combinando DSC e DSPC define-se a validade de cada grupo como uma razão normalizada que compara a separação mínima para os outros grupos com a esparsidade interna:

$$VC(C_i) \;=\; \frac{\min_{j \neq i} \text{DSPC}(C_i, C_j) \;-\; \text{DSC}(C_i)}{\max\left\{ \min_{j \neq i} \text{DSPC}(C_i, C_j),\ \text{DSC}(C_i) \right\}}.$$

   Esta normalização assegura que $VC(C_i) \in [-1, 1]$: valores positivos indicam que C_i é mais denso internamente do que próximo a outros grupos; valores negativos indicam o contrário.

6. **Índice global (**DBCV**):** o DBCV da partição $\mathcal{C} = \{C_1, \ldots, C_\ell\}$ é uma média ponderada das validades por grupo, ponderada pela fração de pontos em cada grupo:

$$\text{DBCV}(\mathcal{C}) \;=\; \sum_{i=1}^{\ell} \frac{|C_i|}{|O|}\, VC(C_i),$$

onde $|O|$ é o número total de objetos (pontos rotulados como *noise* são tratados conforme a implementação: grupos muito pequenos tipicamente não contribuem como *internal nodes* e o índice penaliza partições com elevada proporção de *noise* através da ponderação).

O valor de DBCV varia no intervalo de $[-1, 1]$, e sua interpretação é a seguinte:

- DBCV ≈ 1 indicam partições com grupos bem definidos por densidade (estruturas internas densas e claramente separadas por regiões de baixa densidade).
- DBCV ≈ 0 indica solução indefinida quanto ao critério de densidade (esparsidade interna e separação equilibradas).
- DBCV ≈ -1 indica partição ruim segundo o critério de densidade.

2.3.4 Fundamentação Teórica do ARI

O ARI (Adjusted Rand Index)(HUBERT; ARABIE, 1985) é uma métrica de validação externa para algoritmos de agrupamento que quantifica a concordância entre duas partições dos mesmos dados, ajustada para concordância aleatória. Desenvolvido como uma correção ao Índice de Rand original(RAND, 1971), o ARI resolve o problema de que partições aleatórias não resultam necessariamente em um valor constante no índice original, fornecendo assim uma medida mais robusta e interpretável.

A construção do índice baseia-se na análise combinatória de pares de objetos e sua classificação em relação a duas partições. Para duas partições $\mathcal{U} = \{U_1, U_2, \ldots, U_R\}$ e $\mathcal{V} = \{V_1, V_2, \ldots, V_C\}$ de um conjunto de n objetos, o cálculo envolve os seguintes componentes:

1. **Tabela de Contingência:** Constrói-se uma matriz n_{ij} onde cada elemento representa o número de objetos em comum entre o grupo U_i da primeira partição e o grupo V_j da segunda partição:

$$n_{ij} = |U_i \cap V_j|$$

As somas marginais são definidas como:

$$a_i = \sum_{j=1}^{C} n_{ij} \quad \text{e} \quad b_j = \sum_{i=1}^{R} n_{ij}$$

2. **Contagem de Pares Concordantes e Discordantes:** Para todos os $\binom{n}{2}$ pares possíveis de objetos, classifica-se cada par como:

   - **Concordante em ambas as partições:** dois objetos estão no mesmo grupo em $\mathcal{U}$ e no mesmo grupo em $\mathcal{V}$, ou em grupos diferentes em ambas as partições
   - **Discordante:** dois objetos estão no mesmo grupo em uma partição mas em grupos diferentes na outra

O número de pares concordantes pode ser expresso em termos da tabela de contingência como:

$$a = \sum_{i=1}^{R} \sum_{j=1}^{C} \binom{n_{ij}}{2}$$

3. **Correção para Concordância Aleatória:** O ARI ajusta o índice de Rand original subtraindo o valor esperado sob a hipótese de independência entre as partições. O valor esperado de pares concordantes sob aleatoriedade é:

$$E[a] = \frac{\sum_{i=1}^{R} \binom{a_i}{2} \cdot \sum_{j=1}^{C} \binom{b_j}{2}}{\binom{n}{2}}$$

4. **Normalização:** O índice é normalizado para ter valor máximo 1, resultando na fórmula geral:

$$\text{ARI} = \frac{\text{Índice} - \text{Valor Esperado}}{\text{Valor Máximo} - \text{Valor Esperado}}$$

A formulação completa do ARI é dada por:

$$\text{ARI} = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \cdot \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2} \cdot \sum_j \binom{b_j}{2}\right] / \binom{n}{2}}$$

onde $\binom{x}{2} = \frac{x(x-1)}{2}$ representa o número de combinações de 2 elementos escolhidos de um conjunto de x elementos.

O valor do ARI varia no intervalo $[-1, 1]$, com a seguinte interpretação:

- ARI ≈ 1**:** Indica alta similaridade entre as partições, indicando que os agrupamentos capturam estruturas similares nos dados.
- ARI $= 0$**:** Sugere que a concordância entre as partições não é melhor do que o esperado ao acaso.
- ARI < 0**:** Indica concordância inferior ao esperado por aleatoriedade, sugerindo que as partições são sistematicamente diferentes.

3 TRABALHOS RELACIONADOS

A influência das *word embeddings* em tarefas de agrupamento já vem sendo explorada, conforme demonstrado por (SAHA, 2023). Neste estudo, os autores utilizaram um *corpus* de 1.177 avaliações de produtos da *Amazon* para avaliar o desempenho de algoritmos como *K-means*(LLOYD, 1982), DBSCAN (Density-Based Spatial Clustering of Applications with Noise (Agrupamento Espacial Baseado em Densidade de Aplicações com Ruído))(ESTER *et al.*, 1996), HDBSCAN(CAMPELLO, Ricardo J. G. B.; MOULAVI; SANDER, 2013) e agrupamento hierárquico. O foco principal da metodologia foi comparar, como representações estáticas (*Word2Vec*) e contextuais (nas variações *BERT-CLS* e *BERT-Average*) impactam a formação dos grupos.

Em termos de resultados quantitativos, (SAHA, 2023) demonstraram que o desempenho é altamente dependente da combinação específica entre algoritmo e representação. Para métodos de partição, o ***BERT-CLS* mostrou-se superior**, sendo a única representação a alcançar um índice de silhueta de aproximadamente 0,22 no *K-means* (com $k = 3$), enquanto o *Word2Vec* e *BERT-Average* apresentaram resultados inferiores. Curiosamente, no agrupamento hierárquico, o ***Word2Vec*** superou as representações do BERT em coesão interna à medida que o número de grupos aumentava, sugerindo que sua distribuição espacial favorece a separação em cadeias (*single-linkage*).

No entanto, a sensibilidade foi observada nos algoritmos de densidade, onde as diferenças entre as *embeddings* se tornaram drásticas em relação à detecção de ruído. No DBSCAN, parâmetros restritivos ($\epsilon = 0.5$) geraram silhuetas próximas a 1 para todas as *embeddings*, mas de forma artificial, pois a grande maioria dos dados foi descartada. Já no HDBSCAN ($min\_cluster\_size = 10$), a comparação revelou que *embeddings* baseadas em média colapsaram: tanto o ***Word2Vec*** quanto o ***BERT-Average*** classificaram cerca de 89% dos dados (mais de 1.040 instâncias) como ruído. O ***BERT-CLS***, por sua vez, demonstrou um comportamento ligeiramente mais robusto, retendo mais dados, embora ainda tenha descartado 80% do *corpus* (945 instâncias) como ruído. Esses números evidenciam que, embora os autores sugiram que representações médias (*Average*) possam ter silhuetas teóricas melhores no HDBSCAN, na prática, todas as técnicas falharam em estruturar a maioria dos dados neste domínio.

Embora o trabalho de (SAHA, 2023) seja fundamental para compreender essas nuances e a dificuldade dos algoritmos de densidade em lidar com diferentes espaços vetoriais, ele diverge da presente pesquisa em dois pontos. Primeiramente, as métricas de avaliação limitaram-se a silhueta, ARI e pureza, sem explorar a topologia dos dados. Em segundo lugar, o estudo não incluiu o SOM(KOHONEN, 1990) em seus testes, lacuna que este trabalho visa preencher ao analisar a preservação topológica dos grupos.

Em um estudo de grande escala visando unificar a avaliação de representações textuais, (MUENNIGHOFF *et al.*, 2023) introduziram o *Massive Text Embedding Benchmark* (MTEB), analisando 33 modelos distintos em 8 tarefas e 58 *datasets*. Especificamente para a tarefa de agrupamento, foram utilizadas 11 bases de dados (incluindo *subsets* do *Arxiv*, *Biorxiv*, *Reddit* e *StackExchange*) para testar a capacidade de generalização de modelos que variam desde o clássico *GloVe*(PENNINGTON; SOCHER; MANNING, 2014) até arquiteturas massivas baseadas em *Transformers*, como o MPNet (Masked and Permuted Language Modeling (Modelagem de Linguagem Mascarada e Permutada))(SONG *et al.*, 2020). Uma característica metodológica crucial deste *benchmark* foi a padronização do algoritmo de agrupamento: em todos os cenários, utilizou-se exclusivamente o *K-means* (com k igual ao número de rótulos reais) avaliado pela métrica de *V-Measure*.

Os resultados quantitativos do MTEB revelaram que, para tarefas de agrupamento, a arquitetura e o treinamento do modelo superam a simples escala de parâmetros. Enquanto o *GloVe* obteve uma pontuação média de *V-Measure* de apenas 27,73 e o *BERT* original alcançou 30,12, os modelos modernos treinados especificamente para codificação de sentenças dominaram o *ranking*. O **MPNet**, que alcançou um desempenho de 43,69. Os autores atribuem o sucesso do MPNet à diversidade de seus dados de ajuste

fino.

Apesar da abrangência do MTEB em termos de *datasets*, a metodologia de agrupamento adotada por (MUENNIGHOFF *et al.*, 2023) apresenta limitações que este trabalho busca superar. Ao restringir a avaliação ao *K-means* o estudo pressupõe grupos com geometria esférica e ignora topologias complexas ou densidades variáveis. Diferentemente dessa abordagem, a presente pesquisa não se limita a avaliar as *embeddings*, mas foca no comportamento dos algoritmos de agrupamento em si, implementando além do *K-means*, métodos baseados em densidade (HDBSCAN), hierarquia (HCA) e redes neurais (SOM), permitindo uma análise mais profunda sobre como diferentes espaços vetoriais interagem com a estrutura topológica dos dados.

De forma semelhante, os autores de (WALKOWIAK; GNIEWKOWSKI, 2019) investigaram a sinergia entre *word embeddings* e algoritmos de agrupamento com foco no **idioma polonês**, utilizando dois *corpora* distintos: um de notícias de imprensa (*Press*, com 6.500 documentos) e outro de resenhas científicas (*Reviews*, com 10.500 documentos). O estudo avaliou seis modelos de representação , contrastando abordagens estáticas como o *word2vec* e o *KGR10* (KOCOŃ; GAWOR, 2018) (uma implementação de *FastText*(BOJANOWSKI *et al.*, 2017) treinada em *Skipgram* para o idioma polonês) contra modelos dinâmicos profundos como ELMo (Embeddings from Language Model (Embeddings de Modelo de Linguagem)) e BERT. Além disso, a metodologia testou o impacto da padronização de dados, concluindo que a mesma é essencial para melhorar a performance de medidas de distância baseadas em cosseno

Os resultados quantitativos medidos pelo AMI (Adjusted Mutual Information (Informação Mútua Ajustada)) revelaram que modelos mais complexos não necessariamente geram melhores agrupamentos neste domínio. A combinação do modelo ***KGR10*** com o agrupamento hierárquico atingiu os melhores resultados globais, superando 0,60 de AMI no *dataset* de imprensa. Em contrapartida, o BERT apresentou um desempenho notavelmente inferior, com índices de AMI muitas vezes abaixo de 0,40 nos mesmos testes. Os autores atribuem essa falha ao fato de terem utilizado modelos BERT multilíngues pré-treinados, que não capturaram as nuances morfológicas do polonês tão bem quanto o *KGR10*.

Essa análise diferencia-se do presente trabalho em dois aspectos fundamentais. Primeiro, na metodologia de avaliação: enquanto (WALKOWIAK; GNIEWKOWSKI, 2019) dependem de uma métrica externa (AMI) que exige rótulos verdadeiros, este trabalho emprega métricas de validação interna, como o Índice de Silhueta e o DBI, para avaliar a estrutura intrínseca dos grupos. Segundo, no escopo algorítmico: ao invés de limitar-se a métodos clássicos de partição e hierarquia, esta pesquisa explora técnicas capazes de mapear topologias complexas e densidades variáveis, especificamente através do uso de SOM e HDBSCAN.

4 MATERIAIS E MÉTODOS

Neste capítulo, são apresentados os materiais e a metodologia adotados para a implementação dos agrupamentos e a avaliação dos mesmos. O objetivo é oferecer uma visão clara dos recursos, dos dados e dos procedimentos que possibilitam a realização deste trabalho. Para tanto, o capítulo está organizado da seguinte forma:

1. **Ferramentas e Bibliotecas:** São descritas as tecnologias e os *frameworks* que sustentam o desenvolvimento do projeto, incluindo modelos de agrupamento, modelos de *embeddings*, bibliotecas de aprendizado de máquina e de processamento de linguagem natural (PLN).
2. **Base de dados:** Detalha-se a origem dos dados e a composição das fontes que irão gerar nosso *corpus*.
3. **Geração do *corpus*:** Explicam-se os processos de pré-processamento realizados na base de dados para a geração do *corpus*.
4. **Geração das *embeddings*:** São descritas as etapas de extração de representações (*embeddings*) das palavras, utilizando *word2vec* e SBERT (Sentence-BERT (BERT para Sentenças)).
5. **Geração dos agrupamentos:** São descritas as etapas e estratégias para o treinamento e validação dos algoritmos de agrupamento.

4.1 FERRAMENTAS

4.1.1 Modelos de embeddings

- *word2vec*(MIKOLOV *et al.*, 2013), com o auxílio da biblioteca *Gensim*[1]. Esse algoritmo cria representações vetoriais densas para palavras com base em seu contexto.
- *Transformers*: Biblioteca em *Python* da *Hugging Face* que oferece implementações de modelos de linguagem pré-treinados. Neste trabalho, utilizaremos o SBERT Multilingual[2](REIMERS; GUREVYCH, 2019b).

4.1.2 Bibliotecas de PLN e agrupamento

- Pandas: Biblioteca em *Python* para manipulação e análise de dados. Será usada para carregar, processar e explorar os conjuntos de dados.[3]
- NumPy: Biblioteca fundamental para computação numérica em *Python*. O NumPy fornece suporte a *arrays* e matrizes, além de funções matemáticas de alto desempenho, essenciais para operações com grandes volumes de dados.[4]
- Scikit-Learn: Biblioteca em *Python* para aprendizado de máquina que fornece ferramentas eficientes para mineração e análise de dados; inclui implementações de algoritmos para redução de dimensionalidade.[5]
- NLTK (Natural Language Toolkit): O *Natural Language Toolkit* é uma biblioteca para processamento de linguagem natural (PLN) que fornece ferramentas para *tokenização*, *lemmatização*, análise de sintaxe e outras tarefas de pré-processamento.[6]

[1] https://radimrehurek.com/gensim/models/word2vec.html
[2] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2
[3] https://pandas.pydata.org/
[4] https://numpy.org/
[5] https://scikit-learn.org/stable/
[6] https://www.nltk.org/

4.2 BASE DE DADOS

Para a realização deste trabalho, selecionaram-se dez bases de dados de diferentes domínios, o que permite uma análise abrangente da eficácia dos algoritmos de agrupamento em diversos tipos de *corpora* textuais; a seguir, detalham-se as bases de dados selecionadas.

4.2.1 Goodbooks10k

A base de dados *Goodbooks10k*, criada por *Zygmunt Zając*[7], é amplamente reconhecida em estudos e experimentos voltados a sistemas de recomendação de livros (RAHUTOMO *et al.*, 2019)(BERBATOVA, 2019)(SLOKOM, 2018)(YANG *et al.*, 2024)(KULA, 2018)(PAUDEL; LUCK; BERNSTEIN, 2018). Sua popularidade decorre da riqueza e diversidade dos dados, que abrangem milhares de obras literárias avaliadas por usuários ao longo do tempo.

Este conjunto de dados possibilita a exploração de padrões de comportamento e preferências de leitura e é também utilizado em estudos sobre a aplicação de técnicas de agrupamento (ZHU; LI; WU, 2024); sua relevância é particularmente evidente em análises que buscam analogias com plataformas digitais de recomendação de livros, uma vez que os dados capturam interações reais entre leitores e obras.

O conjunto de dados está dividido em três fontes principais — uma sobre livros, outra sobre tags e uma terceira que associa tags a livros — com as seguintes informações:

- **Livros(*books.csv*):** registros onde cada um representa um livro cadastrado no *Good Books*, com as seguintes colunas mais relevantes:
  - ***goodreads_book_id*:** Identificador do livro no *Good Books*
  - ***books_count*:** Quantos deste livro estão cadastrados
  - ***authors*:** Autores do livro
  - ***title*:** Titulo do livro
- **Tags dos livros (*book_tags.csv*):** registros onde cada um representa uma *tag* que está associada a um livro, com as seguintes colunas:
  - ***goodreads_book_id*:** Identificador do livro no *Good Books*
  - ***tag_id*:** Identificador da *tag*
- **Tags (tags.csv):** registros onde cada um representa uma *tag* diferente, com as seguintes colunas:
  - ***tag_id*:** Identificador da *tag*
  - ***tag_name*:** Nome daquela *Tag*

Para este trabalho, o *corpus* textual será gerado a partir do arquivo ***books.csv***. Cada documento do *corpus* será obtido concatenando os campos ***title*** e ***authors***, separados por um delimitador, criando assim uma representação textual que une a identidade da obra à de seu(s) autor(es) para enriquecer o contexto semântico.

4.2.2 Spotify Music Dataset

O *Spotify Music Dataset*, disponibilizado na plataforma *Kaggle*[8], é uma coleção de dados sobre faixas musicais do *Spotify*. Este conjunto é frequentemente empregado em tarefas de análise de tendências musicais, em sistemas de recomendação e em classificação de gêneros.

Sua aplicação neste estudo justifica-se pela natureza do seu conteúdo textual, que, embora conciso, é semanticamente rico. O nome de uma faixa e o nome do artista, em conjunto, funcionam

[7] https://github.com/zygmuntz/goodbooks-10k
[8] https://www.kaggle.com/datasets/solomonameh/spotify-music-dataset

como identificadores culturais que podem ser explorados por algoritmos de agrupamento para revelar comunidades de artistas ou temas musicais latentes.

O conjunto de dados é composto por dois arquivos — *high_popularity_spotify_data.csv* e *low_popularity_spotify_data.csv* — que contêm, respectivamente, registros de faixas musicais mais e menos populares; ambos apresentam as seguintes colunas relevantes para a análise:

- ***track_id*:** Identificador único da faixa.
- ***track_name*:** Título da música.
- ***track_artist*:** Nome do artista.
- ***track_album_name*:** Nome do álbum.
- ***playlist_genre*:** Gênero principal (*rock, pop, etc.*).
- ***playlist_subgenre*:** Sub Gênero (*Pop Rock, punk rock, etc.*).
- ***popularity*:** Índice de popularidade (0–100).
- ***danceability*, *energy*, *valence*, *tempo*, *loudness*, *liveness*, *speechiness*, *instrumentalness*, *mode*, *duration_ms*, *acousticness*, *key*:** Atributos de áudio.

Para este conjunto de dados, o *corpus* textual será construído a partir da concatenação das colunas ***track_name*** e ***track_artist***, separadas por um delimitador . Os atributos numéricos de áudio foram desconsiderados para garantir que a análise se concentre exclusivamente na capacidade dos algoritmos de agrupar com base na semântica extraída das *embeddings*, segundo os objetivos deste trabalho. A coluna ***playlist_genre*** será utilizada como referência para avaliação por métricas externas.

4.2.3 MovieLens Latest Datasets

O *MovieLens*[9], mantido pelo *GroupLens*(laboratório de pesquisa da Universidade de Minnesota) é um dos conjuntos de dados mais icônicos e amplamente utilizados para pesquisa em sistemas de recomendação. As diversas versões do conjunto incluem avaliações de filmes, metadados e *tags* atribuídas por usuários.

O conjunto de dados é composto por múltiplos arquivos, sendo eles:

- **Avaliações (*ratings.csv*):** registros com:
  - ***userId*:** Identificador do usuário.
  - ***movieId*:** Identificador do filme.
  - ***rating*:** Avaliação numérica (por exemplo, 0.5–5.0).
  - ***timestamp*:** Data e hora da avaliação.
- **Filmes (*movies.csv*):** Cada linha representa um filme, com as seguintes colunas:
  - ***movieId*:** Identificador único do filme.
  - ***title*:** Título do filme, incluindo o ano de lançamento.
  - ***genres*:** Lista de gêneros associados ao filme, separados por ".
- **Tags (*tags.csv*):** Contém *tags* aplicadas por usuários aos filmes:
  - ***userId*:** Identificador do usuário que aplicou a *tag*.
  - ***movieId*:** Identificador do filme.
  - ***tag*:** O texto da *tag* aplicada.

[9] https://grouplens.org/datasets/movielens/

Para este conjunto de dados, será utilizado o arquivo ***movies.csv***; o *corpus* textual será composto exclusivamente pelo conteúdo da coluna ***title***. Embora o conjunto de dados contenha também um arquivo de *tags* geradas por usuários (*tags.csv*), optou-se por não incluí-las, pois são dados brutos que exigiriam um processo complexo de agregação e limpeza e apresentam alto grau de inconsistência e ruído subjetivo. Focar apenas nos títulos permite avaliar de forma mais clara a capacidade dos algoritmos de extrair sinal semântico; o campo ***genres*** será utilizado como referência para avaliação por métricas externas.

4.2.4 Steam Games Dataset

Este conjunto de dados, disponível no *Kaggle*[10], oferece uma visão detalhada do catálogo de jogos da plataforma *Steam*. Inclui informações sobre desenvolvedores, preços, gêneros e descrições, constituindo uma fonte rica para análises de mercado e de tendências na indústria de jogos.

A relevância deste conjunto de dados para o presente estudo reside na possibilidade de agrupar jogos com base em seus atributos textuais. O título, a descrição curta e os gêneros de um jogo fornecem material suficiente para que os modelos de *embeddings* capturem suas características essenciais, permitindo a identificação de nichos e categorias de jogos de forma não supervisionada.

O conjunto de dados principal é um arquivo CSV com as seguintes colunas:

- ***AppID*:** Identificador único do jogo na Steam.
- ***Name*:** Nome do jogo.
- ***Release date*:** Data de lançamento do jogo.
- ***Estimated owners*:** Estimativa do número de proprietários do jogo.
- ***Peak CCU*:** Pico de usuários simultâneos conectados.
- ***Required age*:** Classificação etária mínima requerida.
- ***Price*:** Preço do jogo.
- ***DLC count*:** Número de conteúdos para download (*DLC*) disponíveis.
- ***About the game*:** Descrição detalhada sobre o jogo.
- ***Supported languages*:** Idiomas suportados pelo jogo.
- ***Full audio languages*:** Idiomas com áudio completo disponível.
- ***Reviews*:** Avaliações dos usuários.
- ***Header image*:** *URL* da imagem de cabeçalho do jogo.
- ***Website*:** *Website* oficial do jogo.
- ***Support url*:** *URL* de suporte técnico.
- ***Support email*:** *Email* de suporte técnico.
- ***Windows*:** Compatibilidade com sistema *Windows*.
- ***Mac*:** Compatibilidade com sistema *macOS*.
- ***Linux*:** Compatibilidade com sistema *Linux*.
- ***Metacritic score*:** Pontuação no *Metacritic*.
- ***Metacritic url*:** *URL* da página do jogo no *Metacritic*.
- ***User score*:** Pontuação atribuída pelos usuários.
- ***Positive*:** Número de avaliações positivas.

[10] https://www.kaggle.com/datasets/fronkongames/steam-games-dataset

- ***Negative*:** Número de avaliações negativas.
- ***Score rank*:** Classificação baseada na pontuação.
- ***Achievements*:** Número de conquistas disponíveis no jogo.
- ***Recommendations*:** Número de recomendações recebidas.
- ***Notes*:** Observações adicionais sobre o jogo.
- ***Average playtime forever*:** Tempo médio de jogo total dos usuários.
- ***Average playtime two weeks*:** Tempo médio de jogo nas últimas duas semanas.
- ***Median playtime forever*:** Mediana do tempo de jogo total dos usuários.
- ***Median playtime two weeks*:** Mediana do tempo de jogo nas últimas duas semanas.
- ***Developers*:** Desenvolvedores do jogo.
- ***Publishers*:** Publicadoras do jogo.
- ***Categories*:** Categorias associadas ao jogo.
- ***Genres*:** Gêneros associados ao jogo.
- ***Tags*:** *Tags* ou marcadores atribuídos ao jogo.
- ***Screenshots*:** *URLs* das capturas de tela do jogo.
- ***Movies*:** *URLs* dos vídeos promocionais do jogo.

O *corpus* será construído a partir da combinação de três campos textuais: ***Name***, ***Developers*** e ***Tags***. Essa escolha visa produzir um documento semanticamente rico e denso. Apesar da aparente riqueza de informação, o campo *About the game* foi deliberadamente omitido, pois descrições extensas frequentemente incorporam linguagem de *marketing* genérica e detalhes secundários que tendem a diluir a identidade semântica do jogo. Em contrapartida, a combinação selecionada funciona como um resumo de alta densidade baseado em metadados curados. O campo ***Genre*** será utilizado como referência para a avaliação das métricas externas.

4.2.5 RecipeNLG

O conjunto de dados *RecipeNLG*[11] é um extenso compilado de receitas culinárias, contendo títulos, ingredientes e instruções, originalmente projetado para tarefas de Geração de Linguagem Natural (*Natural Language Generation*), cuja estrutura detalhada o torna ideal para estudos de extração de informação e agrupamento.

A utilização deste conjunto de dados permite avaliar como algoritmos de agrupamento se comportam em um domínio com vocabulário altamente específico e textos estruturados, tendo como objetivo identificar receitas que compartilham ingredientes similares com base exclusivamente na representação vetorial de seu conteúdo textual.

O conjunto de dados principal, *RecipeNLG_dataset.csv*, possui as seguintes colunas relevantes:

- ***title*:** Título da receita.
- ***ingredients*:** Lista de ingredientes necessários para a receita.
- ***directions*:** Instruções passo a passo para o preparo da receita.
- ***link*:** *URL* original da receita.
- ***source*:** Fonte ou *website* de onde a receita foi extraída.
- ***NER*:** Entidades nomeadas extraídas da receita (*Named Entity Recognition*).

[11] https://www.kaggle.com/datasets/paultimothymooney/recipenlg

Neste estudo, o *corpus* textual de cada receita será formado pela concatenação de seu **titulo(*title*)** com a lista de **ingredientes*(ingredients)***. A coluna *directions* foi intencionalmente omitida para focar nos componentes essenciais que definem a receita, osem vez de seu passo a passo, reduzindo assim o ruído textual.

4.2.6 BBC News Summary

O *dataset BBC News Summary*[12] é um compilado de artigos de notícias da BBC, originalmente coletado e utilizado em pesquisas de categorização de documentos e posteriormente adaptado para tarefas de sumarização de texto, abrangendo notícias publicadas entre 2004 e 2005 distribuídas em cinco categorias temáticas distintas. Sua relevância para este estudo decorre de sua estrutura de alta qualidade, com artigos previamente categorizados, o que o torna um excelente *benchmark* para avaliar o desempenho de algoritmos de agrupamento.

O conjunto de dados é organizado em uma estrutura de diretórios que separa os artigos completos de seus respectivos resumos:

- **Artigos (*News Articles*):** Um diretório principal contendo cinco subdiretórios, um para cada categoria (*business, entertainment, politics, sport, tech*). Cada arquivo dentro desses subdiretórios contém o texto integral de uma notícia, onde a primeira frase geralmente corresponde ao título.
- **Resumos (*Summaries*):** Um diretório com estrutura idêntica ao de artigos, mas contendo resumos criados por humanos para cada notícia correspondente.

Para este trabalho, o *corpus* será formado pelo conteúdo textual resumido dos arquivos localizados na pasta ***Summaries***, sendo cada artigo tratado como um documento individual, enquanto as categorias correspondentes serão utilizadas como referência para a avaliação das métricas externas.

4.2.7 Amazon US Customer Reviews Dataset

O *Amazon US Customer Reviews Dataset*, disponibilizado na plataforma *Kaggle*[13], constitui um dos maiores compilados de avaliações de produtos do comércio eletrônico mundial, abrangendo milhões de avaliações de clientes reais coletadas ao longo de diversos anos e representando uma ampla variedade de categorias de produtos disponíveis na plataforma *Amazon*

Sua aplicação neste estudo justifica-se pela riqueza semântica das avaliações textuais, que mesclam opiniões subjetivas e descrições objetivas dos produtos, proporcionando uma oportunidade robusta para avaliar a capacidade dos algoritmos de agrupamento de identificar padrões em textos que combinam linguagem comercial, técnica e opinativa — característica recorrente em plataformas de *e-commerce*.

O conjunto de dados apresenta as seguintes colunas principais:

- ***marketplace*:** Plataforma onde a avaliação foi publicada.
- ***customer_id*:** Identificador único do cliente que realizou a avaliação.
- ***review_id*:** Identificador único da avaliação.
- ***product_id*:** Identificador único do produto avaliado.
- ***product_parent*:** Identificador do produto pai (quando aplicável).
- ***product_title*:** Título/nome do produto.
- ***product_category*:** Categoria principal do produto.
- ***star_rating*:** Avaliação numérica (1-5 estrelas).

[12] https://www.kaggle.com/datasets/pariza/bbc-news-summary
[13] https://www.kaggle.com/datasets/cynthiarempel/amazon-us-customer-reviews-dataset

- ***helpful_votes*:** Número de votos indicando que a avaliação foi útil.
- ***total_votes*:** Número total de votos recebidos pela avaliação.
- ***vine*:** Indicador se a avaliação faz parte do programa *Amazon Vine*.
- ***verified_purchase*:** Indicador se a compra foi verificada.
- ***review_headline*:** Título da avaliação.
- ***review_body*:** Texto completo da avaliação.
- ***review_date*:** Data da publicação da avaliação.

Para a execução dos experimentos, o *corpus* textual será construído pela concatenação das colunas ***product_title***, ***review_headline*** e ***review_body***, formando um documento que une a identidade do produto às percepções e experiências expressas pelos consumidores; a coluna ***product_category*** será utilizada como referência para a avaliação das métricas externas.

4.2.8 Reuters-21578 Text Categorization Collection

O conjunto de dados *Reuters-21578*[14], mantido pelo *UCI Machine Learning Repository*, representa um dos mais importantes *benchmarks* históricos para tarefas de categorização de textos. Essa coleção contém documentos publicados pelo serviço de notícias da *Reuters* em 1987, posteriormente indexados e categorizados por especialistas da *Reuters Ltd.* e do *Carnegie Group Inc.*

Sua relevância para o presente estudo decorre de sua consolidação como padrão de referência (*gold standard*) em pesquisas de mineração de texto e agrupamento de documentos. Os textos apresentam características do jornalismo profissional da década de 1980, oferecendo vocabulário formal e estrutura narrativa consistente, o que permite avaliar a eficácia dos algoritmos em identificar temas jornalísticos distintos baseando-se exclusivamente no conteúdo semântico.

O conjunto de dados é organizado em arquivos ?? contendo aproximadamente 21.578 documentos, com as seguintes informações estruturais:

- ***TITLE*:** Título da notícia.
- ***DATELINE*:** Linha de data e local de origem da notícia.
- ***BODY*:** Corpo principal do artigo jornalístico.
- ***TOPICS*:** Categorias temáticas atribuídas ao documento (*earnings, acquisitions, money-fx, etc.*).
- ***PLACES*:** Localização geográfica relevante à notícia.
- ***PEOPLE*:** Pessoas mencionadas no artigo.
- ***ORGS*:** Organizações citadas na reportagem.
- ***EXCHANGES*:** Bolsas de valores mencionadas.
- ***COMPANIES*:** Empresas referenciadas no texto.

Para este trabalho, o corpus será formado pela concatenação dos campos ***TITLE*** e ***BODY***, criando um documento que combina o resumo editorial (título) com o conteúdo informativo detalhado. Os demais campos de metadados (*PLACES*, *PEOPLE*, *ORGS*) foram intencionalmente excluídos para focar na capacidade dos algoritmos de extrair categorização semântica diretamente do texto narrativo, sem depender de entidades nomeadas pré-processadas.

[14] https://archive.ics.uci.edu/dataset/137/reuters+21578+text+categorization+collection

4.2.9 20 Newsgroups Dataset

O *20 Newsgroups Dataset* é um conjunto clássico amplamente utilizado em pesquisas de aprendizado de máquina e processamento de linguagem natural, disponível através do repositório *scikit-learn*[15]. Este conjunto de dados consiste em aproximadamente 20.000 mensagens de *newsgroups* coletadas de 20 grupos de discussão diferentes da *Usenet*, representando uma diversidade temática que abrange tecnologia, política, religião, esportes e *hobbies*.

A aplicação deste conjunto de dados é particularmente relevante para avaliar o desempenho dos algoritmos de agrupamento em textos de linguagem informal e conversacional. Diferentemente de textos jornalísticos ou acadêmicos, as mensagens de *newsgroups* apresentam características linguísticas únicas, incluindo gírias, abreviações, erros de digitação e estilos de escrita pessoais, oferecendo um desafio distinto para os modelos de *embedding*.

O conjunto de dados está estruturado em 20 categorias temáticas distintas:

- ***comp.graphics*:** Gráficos computacionais
- ***comp.os.ms-windows.misc*:** Sistema operacional *Microsoft Windows*
- ***comp.sys.ibm.pc.hardware*:** *Hardware IBM PC*
- ***comp.sys.mac.hardware*:** *Hardware Macintosh*
- ***comp.windows.x*:** Sistema de janelas *X Window*
- ***misc.forsale*:** Itens à venda
- ***rec.autos*:** Automóveis
- ***rec.motorcycles*:** Motocicletas
- ***rec.sport.baseball*:** *Baseball*
- ***rec.sport.hockey*:** *Hockey*
- ***sci.crypt*:** Criptografia
- ***sci.electronics*:** Eletrônica
- ***sci.med*:** Medicina
- ***sci.space*:** Espaço e astronomia
- ***soc.religion.christian*:** Cristianismo
- ***talk.politics.guns*:** Política sobre armas
- ***talk.politics.mideast*:** Política do Oriente Médio
- ***talk.politics.misc*:** Política geral
- ***talk.religion.misc*:** Religião geral
- ***alt.atheism*:** Ateísmo

Cada documento no conjunto de dados contém campos padrão de *email*/*newsgroup*, incluindo *From*, *Subject*, *Organization*, *Lines* e *Body*. Para a construção do corpus textual, será utilizada exclusivamente a concatenação dos campos ***Subject*** e ***Body***, focando no conteúdo semântico principal da mensagem. Os cabeçalhos de metadados foram deliberadamente excluídos para evitar que informações como endereços de *email* ou organizações influenciem artificialmente o agrupamento. As categorias temáticas serão utilizadas como referência para a avaliação utilizando métricas externas

[15] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html

4.2.10 Stack Overflow Questions Dataset

O *Stack Overflow Questions Dataset*, disponibilizado através do *Stack Exchange Data Dump*[16], constitui uma extensa coleção de perguntas técnicas, respostas e metadados da maior plataforma de perguntas e respostas para desenvolvedores no mundo. Este conjunto de dados oferece uma janela única para o vocabulário técnico especializado e padrões de comunicação da comunidade de desenvolvimento de *software*.

A relevância deste conjunto de dados para o estudo de agrupamento de *embeddings* está na sua natureza altamente especializada e estruturada. As perguntas no *Stack Overflow* seguem convenções específicas, combinam linguagem técnica formal com explicações didáticas. Essas características permitem avaliar como os algoritmos se comportam em domínios com terminologia técnica densa e conceitos abstratos de programação.

As principais estruturas de dados relevantes incluem:

- ***Id*:** Identificador único da pergunta.
- ***PostTypeId*:** Tipo de postagem (1 = pergunta, 2 = resposta).
- ***CreationDate*:** Data de criação da postagem.
- ***Score*:** Pontuação atribuída pela comunidade.
- ***ViewCount*:** Número de visualizações da pergunta.
- ***Body*:** Conteúdo *HTML* da pergunta ou resposta.
- ***OwnerUserId*:** Identificador do usuário que fez a pergunta.
- ***LastEditorUserId*:** Identificador do último usuário a editar.
- ***LastEditDate*:** Data da última edição.
- ***LastActivityDate*:** Data da última atividade.
- ***Title*:** Título da pergunta.
- ***Tags*:** Lista de *tags* associadas, separadas por delimitadores.
- ***AnswerCount*:** Número de respostas recebidas.
- ***CommentCount*:** Número de comentários.
- ***FavoriteCount*:** Número de usuários que favoritaram.
- ***AcceptedAnswerId*:** *ID* da resposta aceita como solução.

Para a construção do corpus textual, será utilizada a concatenação dos campos ***Title*** e ***Body*** (após remoção de marcação *HTML*), criando documentos que combinam o resumo do problema técnico com sua descrição detalhada.

4.3 PRÉ-PROCESSAMENTO DO CORPUS

O próximo passo, após a criação do *corpus*, consiste no pré-processamento do mesmo. Dada a diversidade dos domínios textuais contemplados neste estudo - abrangendo desde títulos literários até descrições culinárias - torna-se necessário implementar um pipeline de pré-processamento que seja tanto robusto quanto adaptável às especificidades de cada conjunto de dados. É importante ressaltar que a estratégia de pré-processamento será diferenciada de acordo com o modelo de geração de *embeddings* a ser empregado, considerando as particularidades e requisitos específicos de cada abordagem.

[16] https://archive.org/details/stackexchange

4.3.1 Pré-processamento para Word2Vec

Para a geração de *embeddings* utilizando o algoritmo *Word2Vec*, será aplicado um pré-processamento tradicional e mais rigoroso, uma vez que este modelo se beneficia significativamente da redução de ruído e normalização textual para aprender representações de qualidade.

O primeiro passo do pré-processamento será a normalização específica por domínio, etapa que precede as técnicas tradicionais de limpeza textual. Esta fase contempla o tratamento de elementos característicos de cada tipo de corpus: nos dados musicais, serão removidas expressões comuns como *"feat.", "ft."e "featuring"* que podem introduzir ruído na análise semântica; nos títulos de filmes, será realizada a normalização de anos e caracteres especiais; nas receitas culinárias, será implementada a padronização de medidas e quantidades; e nos dados de jogos, será efetuado o tratamento adequado de *tags* e categorias separadas por delimitadores específicos.

Após este processamento, será realizado a *lematização*, utilizando a biblioteca NLTK[17]. Essa técnica reduz palavras à sua forma básica (*lemas*), contribuindo para a redução da dimensionalidade do vocabulário e para o agrupamento semântico de formas flexionadas. Por exemplo, as palavras *catching* e *caught* são transformadas em *catch*, como ilustrado na Figura 5. Este processo é particularmente relevante, considerando a natureza heterogênea dos dados, onde diferentes domínios podem apresentar variações morfológicas distintas.

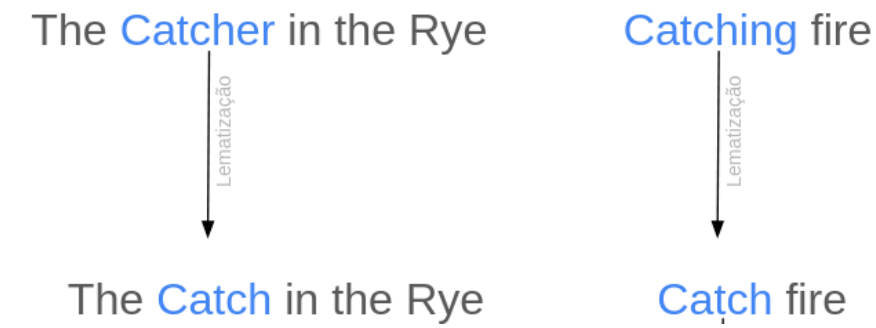


Figura 5 – Títulos antes e depois da lematização

O próximo passo do pré-processamento será a *tokenização* do *corpus*, utilizando também a biblioteca NLTK[18]. A *tokenização* consiste em dividir o texto em unidades menores, denominadas *tokens* (como palavras ou subpalavras). Esse processo é fundamental, pois garante que o texto esteja devidamente estruturado para a criação de *embeddings*, permitindo que cada unidade textual seja representada como um vetor numérico. Para os dados que envolvem concatenação de campos - como título e autor em livros, ou nome da música e artista - será implementado um tratamento específico com separadores semânticos, permitindo que o modelo preserve a distinção entre os diferentes componentes informativos.

A Figura 6 ilustra a representação de exemplos de títulos de livros após passarem pelo processo de *tokenização*, destacando como os textos são segmentados em suas respectivas unidades.

[17] https://www.nltk.org/api/nltk.stem.WordNetLemmatizer.html?highlight=wordnet
[18] https://www.nltk.org/api/nltk.tokenize.html

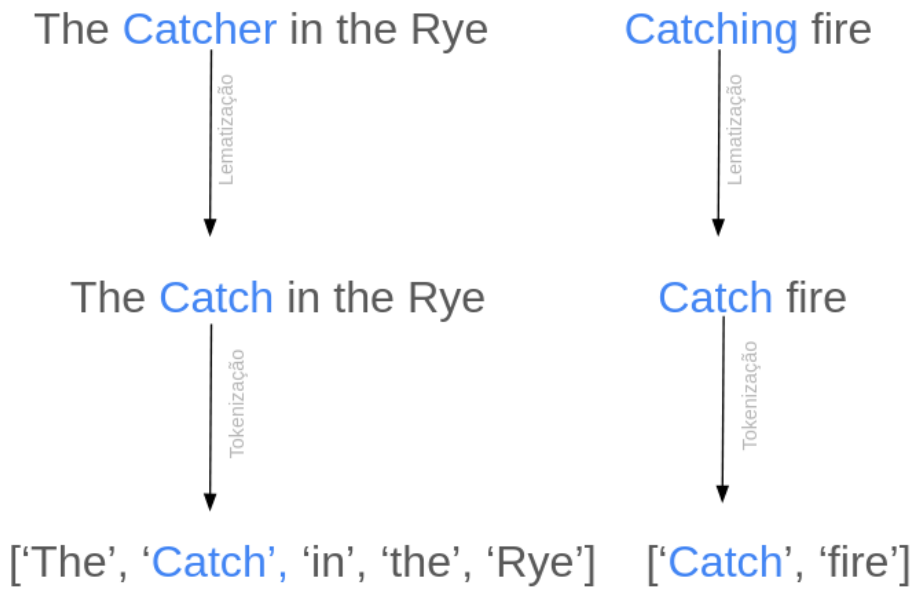


Figura 6 – *Tokenização* dos títulos.

Por fim, o último passo do pré-processamento para o *Word2Vec* será a remoção das *stopwords*, utilizando a biblioteca NLTK[19]. Esse processo eliminará palavras comuns e pouco informativas, como *the* e *in*, reduzindo assim os ruídos no *corpus* e preservando apenas as palavras de maior relevância semântica para o contexto.

Na Figura 7, é possível observar um exemplo de como algumas palavras do *corpus* ficarão após a conclusão das etapas de pré-processamento para o *Word2Vec*.

[19] https://www.nltk.org

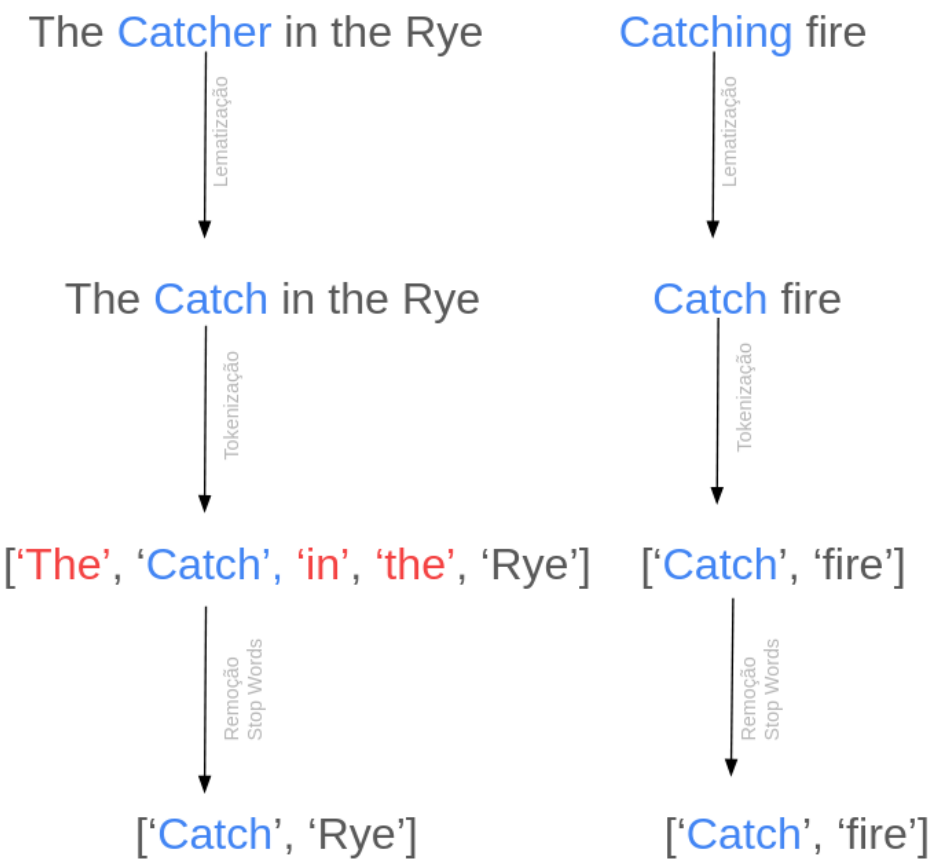


Figura 7 – Resultado do pré-processamento para Word2Vec

4.3.2 Pré-processamento para modelos baseados em Transformers

Para a geração de *embeddings* utilizando modelos baseados em *Sentence-Transformers*, será adotada uma abordagem de pré-processamento mais conservadora. Estes modelos foram treinados em linguagem natural e preservam informações contextuais cruciais que seriam perdidas com um pré-processamento tradicional rigoroso.

O pré-processamento para estes modelos consistirá apenas na normalização específica por domínio, conforme descrito anteriormente, e em uma limpeza textual básica que inclui a remoção de caracteres especiais desnecessários, normalização de espaços em branco e correção de codificação de caracteres.

4.4 GERAÇÃO DOS EMBEDDINGS

Uma vez finalizado o pré-processamento, a próxima etapa será a geração das representações vetoriais dos documentos. É fundamental destacar que o objetivo deste trabalho é agrupar documentos completos (livros, músicas, filmes, etc.) e não palavras individuais, portanto, cada documento deve ser representado por um único vetor que capture sua semântica global.

4.4.1 Geração de embeddings com Word2Vec

Iniciaremos esse processo gerando as *word embeddings* utilizando o algoritmo *word2vec*, com o auxílio da biblioteca *Gensim*[20]. Esse algoritmo cria representações vetoriais densas para palavras com base em seu contexto local. O modelo processa sequências de texto pré-processadas para identificar relações semânticas entre as palavras, levando em consideração o entorno de cada termo no vocabulário. Essa

[20] https://radimrehurek.com/gensim/models/word2vec.html

abordagem possibilita a captura de associações significativas, inclusive para palavras raras ou específicas de cada domínio estudado, desde terminologias culinárias especializadas até jargões específicos da indústria de jogos eletrônicos.

O processo de geração das *word embeddings* após o pré-processamento será realizado nas seguintes etapas:

1. **Configuração adaptativa do modelo:** Esta etapa consistirá nos ajustes dos parâmetros do algoritmo de acordo com as características específicas de cada conjunto de dados. Considerando que textos curtos (como títulos de músicas) demandam janelas de contexto menores, enquanto textos mais extensos (como descrições de receitas) beneficiam-se de janelas maiores, os parâmetros como tamanho do vetor de palavras, tamanho da janela de contexto e quantidade mínima de ocorrências serão calibrados especificamente para cada domínio textual.
2. **Treinamento do modelo:** Esta etapa consistirá em treinar o *word2vec* utilizando cada conjunto de dados já processado, permitindo que o modelo aprenda as representações vetoriais específicas de cada domínio, preservando tanto as particularidades semânticas quanto as possíveis inter-relações entre diferentes áreas do conhecimento.
3. **Extração das *word embeddings*:** Após o treinamento, as representações vetoriais são extraídas para cada palavra do vocabulário presente nos documentos.
4. **Agregação para *document embeddings*:** Uma vez obtidas as representações vetoriais de cada palavra, é necessário agregar essas informações para criar uma única representação vetorial por documento. Para cada documento, será calculada a média aritmética dos vetores de todas as palavras que o compõem, resultando em um *document embedding* que preserva as características semânticas globais do texto. Esta abordagem de agregação por média tem se mostrado eficaz na literatura por sua simplicidade e capacidade de capturar a semântica geral do documento, permitindo que documentos com vocabulários similares apresentem representações próximas no espaço vetorial.

A Figura 8 exemplifica como é a representação em *embeddings* das saídas do pré-processamento realizado anteriormente, ou seja, palavras individuais como aquelas presentes em *Catch Rye* e *Catch fire*, que posteriormente serão agregadas para formar a representação do documento completo.

Figura 8 – Exemplificação *Word2Vec* - representações de palavras individuais

4.4.2 Geração de embeddings com Sentence-Transformers

Além da geração de *embeddings* com o *Word2Vec*, que eficazmente captura associações locais entre palavras, também serão gerados *document embeddings* utilizando modelos baseados em *Transformers* através da biblioteca *Sentence-Transformers*[21]. Esta abordagem representa o estado da arte em geração de

[21] https://www.SBERT.net/

representações textuais, sendo especificamente projetada para criar *embeddings* de sequências completas de texto (frases, parágrafos, documentos) ao invés de palavras individuais.

Diferentemente do *Word2Vec*, que requer a etapa adicional de agregação para formar representações de documentos, os modelos *Sentence-Transformers* geram diretamente um único vetor para cada sequência textual, incorporando naturalmente informações contextuais e relacionais complexas. Esta característica os torna particularmente adequados para capturar nuances semânticas em textos de diferentes extensões e complexidades, aspecto especialmente relevante considerando a diversidade semântica dos domínios estudados.

Para este estudo, será utilizado o modelo *paraphrase-multilingual-MiniLM-L12-v2*[22], que oferece suporte multilíngue robusto e é otimizado para tarefas de similaridade semântica. A escolha deste modelo justifica-se pela natureza internacional dos base de dados utilizados, que contêm nomes de artistas, títulos de obras e conteúdos em múltiplos idiomas, desde português e inglês até outras línguas presentes nos dados de música, filmes e jogos.

O processo de geração dos *document embeddings* com *Sentence-Transformers* será realizado nas seguintes etapas:

1. **Carregamento do modelo:** O modelo *paraphrase-multilingual-MiniLM-L12-v2* será carregado da biblioteca *Sentence-Transformers*, juntamente com seu tokenizador específico, que já inclui o tratamento adequado para múltiplos idiomas e a *tokenização* otimizada.
2. **Geração direta de *document embeddings*:** Para cada documento do *corpus*, o modelo gerará diretamente um vetor de dimensão fixa que representa sua semântica global. Este processo elimina a necessidade de estratégias de agregação, uma vez que o modelo foi especificamente treinado para produzir representações coerentes de sequências textuais completas, independentemente de sua extensão ou estrutura.

A Figura 9 ilustra o comportamento dos modelos baseados em *Transformers* ao gerar *embeddings* contextuais, demonstrando como diferentes contextos podem influenciar a representação de elementos textuais similares, resultando em representações vetoriais mais refinadas e adaptadas ao contexto específico de cada documento.

[22] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

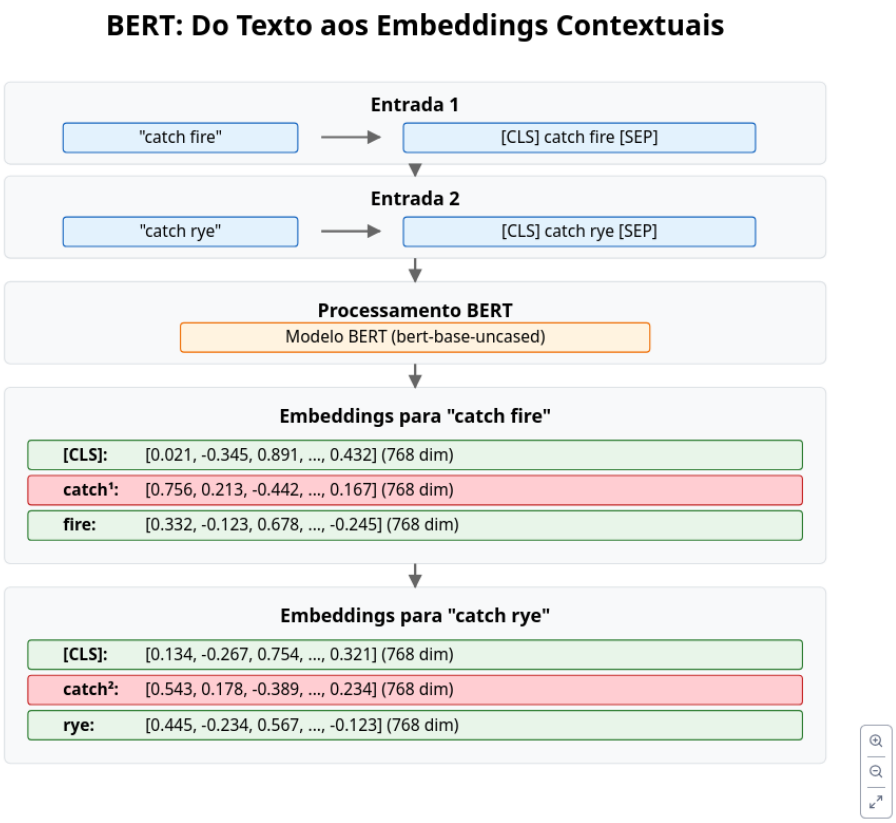


Figura 9 – Exemplificação de geração contextual com modelos *Transformer*

4.5 IMPLEMENTAÇÃO DOS ALGORITMOS DE AGRUPAMENTO

4.5.1 *K-means*

Após a geração das *embeddings*, será iniciada a aplicação de algoritmos de agrupamento, começando pela implementação do *K-Means*

4.5.1.1 Pré-processamento dos dados

Antes da aplicação do algoritmo *K-means*, os dados passam por uma etapa de pré-processamento para garantir a qualidade e adequação das *embeddings*. Inicialmente, são identificadas e removidas as linhas que contêm apenas valores zero, uma vez que esses vetores não contribuem significativamente para o processo de agrupamento. Em seguida, é aplicada a normalização *L2* aos vetores válidos, processo que padroniza a magnitude das *embeddings* e permite que o algoritmo foque nas relações angulares entre os vetores.

A normalização *L2*, também conhecida como normalização euclidiana, é definida matematicamente pela fórmula:

$$\hat{x} = \frac{x}{\|x\|_2} = \frac{x}{\sqrt{\sum_{i=1}^{n} x_i^2}} \tag{4}$$

Onde x representa o vetor original, $\hat{x}$ é o vetor normalizado, $\|x\|_2$ denota a norma *L2* (ou norma euclidiana) do vetor, e n é a dimensionalidade do vetor. Este processo de normalização transforma os vetores em vetores unitários, ou seja, vetores com magnitude igual a 1, preservando apenas sua direção no espaço multidimensional.

4.5.1.2 Definição do número de grupos

A escolha do número ideal de grupos foi realizada utilizando uma versão robusta do Método do Cotovelo. Diferentemente da implementação tradicional, esta abordagem executa múltiplas iterações do algoritmo *K-means* para cada valor de k testado (configurado para 8 execuções por valor de k), permitindo uma análise estatística mais confiável dos resultados. A partir dessas execuções, a inércia (ou WCSS (Within-Cluster Sum of Squares (Soma de Quadrados Intra-Grupos))) mínima foi selecionada para cada k, mitigando o risco de uma inicialização desfavorável influenciar o resultado.

A inércia, é uma métrica que quantifica a compacidade dos agrupamentos formados, sendo definida matematicamente pela fórmula:

$$WCSS = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2 \tag{5}$$

onde k representa o número de grupos, C_i denota o conjunto de pontos pertencentes ao grupo i, x são os pontos de dados individuais, e μ_i é o centroide do grupo i. Esta métrica calcula a soma das distâncias euclidianas quadráticas entre cada ponto de dados e o centroide de seu respectivo grupo. O algoritmo k-means minimiza as variâncias intra-grupos (distâncias euclidianas quadráticas), tornando a inercia uma medida direta da qualidade dos agrupamentos formados. O objetivo é encontrar o valor de k que representa um ponto de equilíbrio, onde um aumento no número de grupos não resulta em um ganho substancial na homogeneidade intra-grupo

A robustez do método é aprimorada através da possibilidade de aplicação de suavização por média móvel na curva resultante, reduzindo flutuações que poderiam interferir na identificação do ponto ótimo. Para determinação do "cotovelo", utiliza-se a métrica de distância perpendicular, onde traça-se uma reta entre os pontos extremos da curva e identifica-se o ponto que apresenta a maior distância em relação a essa reta. Essa distância é calculada através da fórmula abaixo, que mensura a distância de um ponto qualquer à reta definida pelos pontos p_0 e p_1, utilizando as coordenadas (x, y) dos pontos da curva:

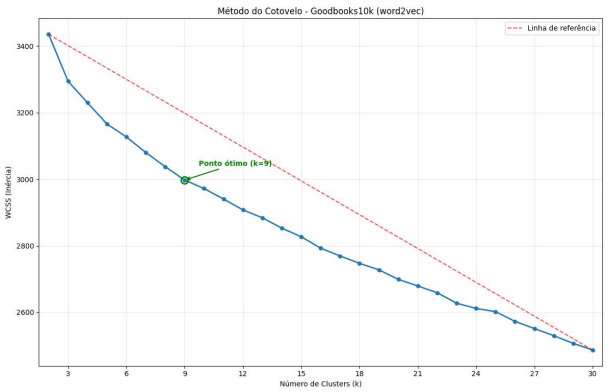
$$distancia(p_0, p_1, (x, y)) = \frac{|(y_1 - y_0)x - (x_1 - x_0)y + x_1 y_0 - y_1 x_0|}{\sqrt{(y_1 - y_0)^2 + (x_1 - x_0)^2}} \tag{6}$$

A implementação permite a seleção entre utilizar a média ou o valor mínimo das inércias obtidas nas múltiplas execuções para construção da curva de análise. O método testa valores de k em um intervalo configurável, sendo padrão de 2 a 30 grupos, e para cada k são realizadas 8 execuções independentes com diferentes sementes aleatórias para garantir a estabilidade dos resultados.

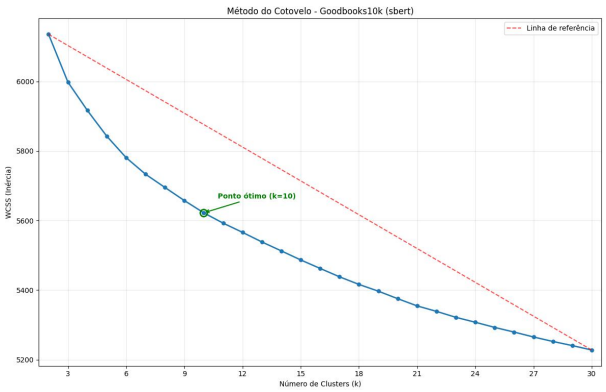
Para determinar o número ideal de grupos aplicamos o método do cotovelo em cada conjunto de dados utilizando tanto *embeddings* com *Word2Vec* quanto com SBERT. Os resultados obtidos estão sumarizados na Tabela 1 e os gráficos de cotovelo correspondentes estão nas Figuras 10a–14c (para *Word2Vec*) e 10b–14d (para SBERT). Cada conjunto de figuras segue a mesma ordem dos base de dados listados na tabela, de cima para baixo, o que facilita a comparação direta entre os métodos.

Tabela 1 – Número ótimo de grupos por conjunto de dados e método de *embedding*

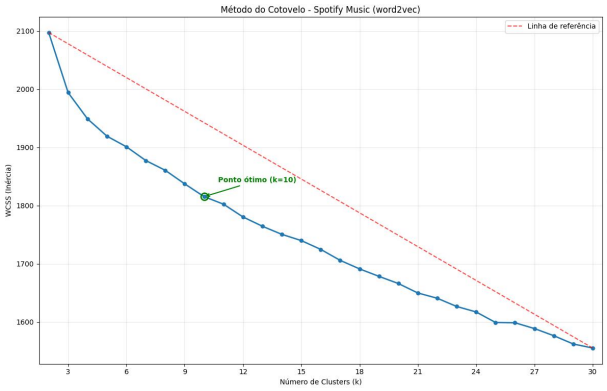
conjunto de dados	***Word2Vec***	**SBERT**
Goodbooks10k	9	10
Spotify Music Dataset	10	10
MovieLens Latest Datasets	8	12
Steam Games Dataset	10	9
RecipeNLG	8	11
BBC News Summary	8	9
Amazon US Customer Reviews Dataset	9	7
Reuters-21578 Text Categorization Collection	8	9
20 Newsgroups Dataset	10	9
Stack Overflow Questions Dataset	10	10



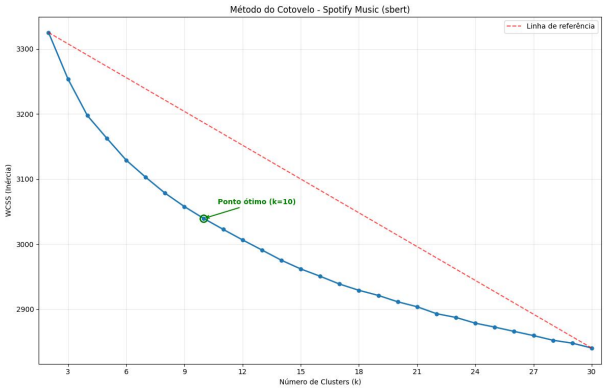
(a) *Goodbooks10k* — *Word2Vec*



(b) *Goodbooks10k* — SBERT

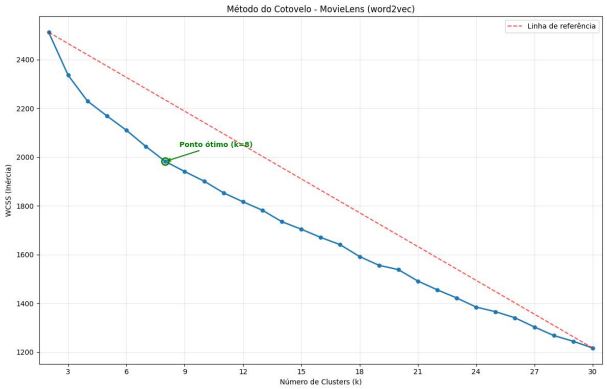


(c) *Spotify Music Dataset* — *Word2Vec*

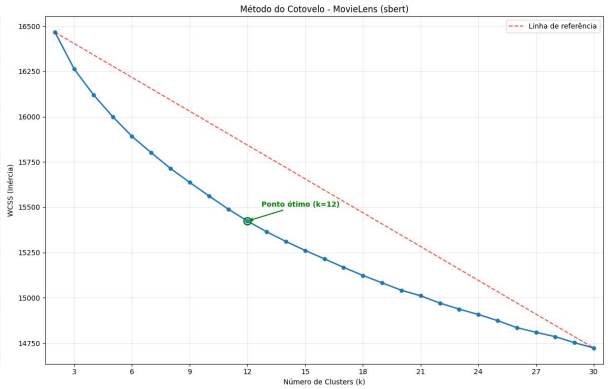


(d) *Spotify Music Dataset* — SBERT

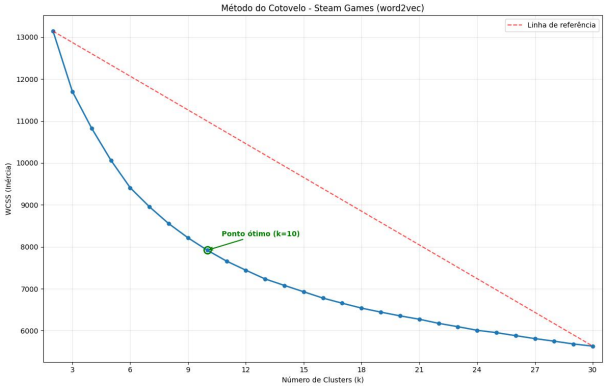
Figura 10 – Gráficos de cotovelo do *K-means* (parte 1). À esquerda: resultados para *Word2Vec*; à direita: resultados para SBERT.



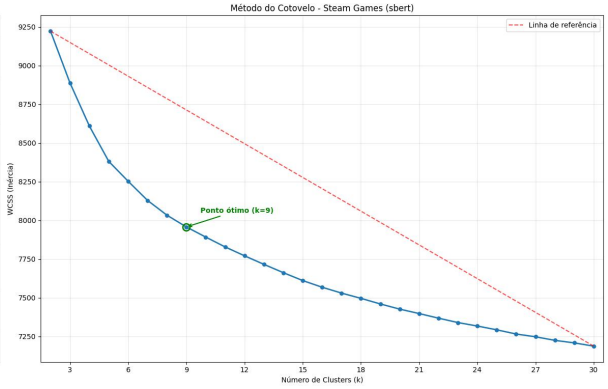
(a) *MovieLens Latest Datasets* — *Word2Vec*



(b) *MovieLens Latest Datasets* — SBERT

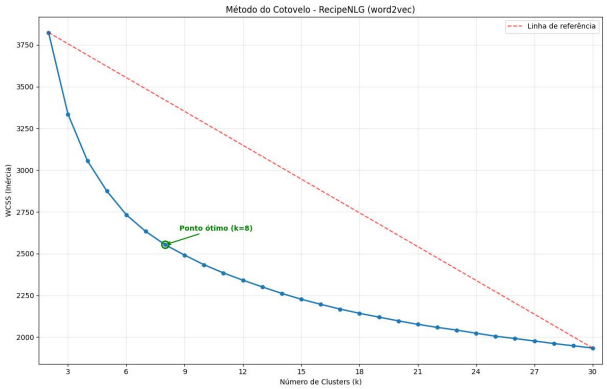


(c) *Steam Games Dataset* — *Word2Vec*

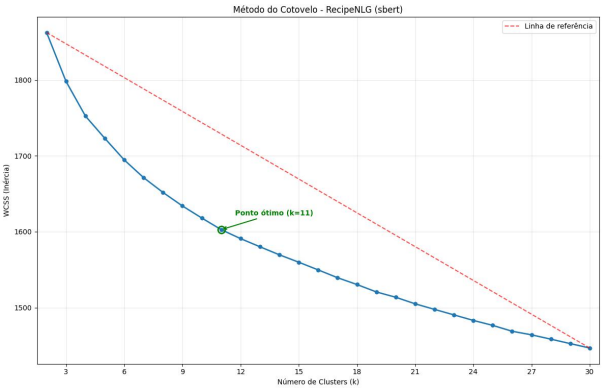


(d) *Steam Games Dataset* — SBERT

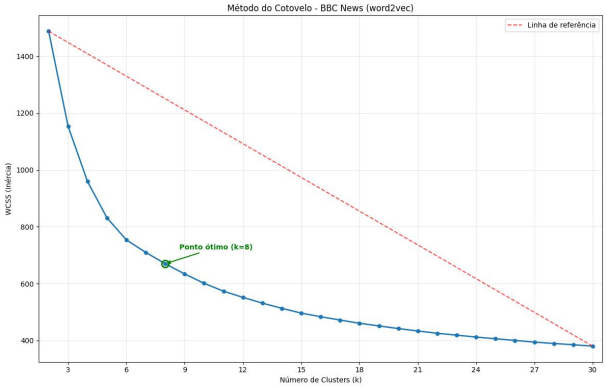
Figura 11 – Gráficos de cotovelo do *K-means* (parte 2). À esquerda: resultados para *Word2Vec*; à direita: resultados para SBERT.



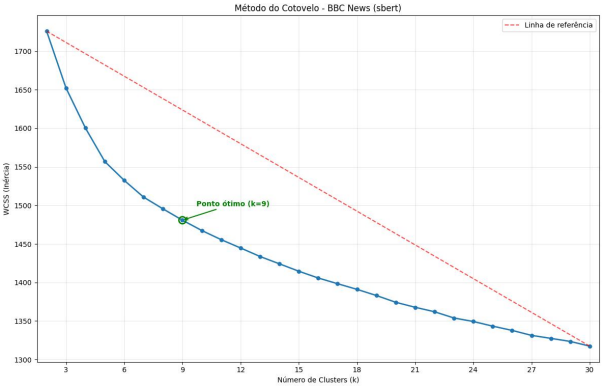
(a) *RecipeNLG — Word2Vec*



(b) *RecipeNLG* — SBERT

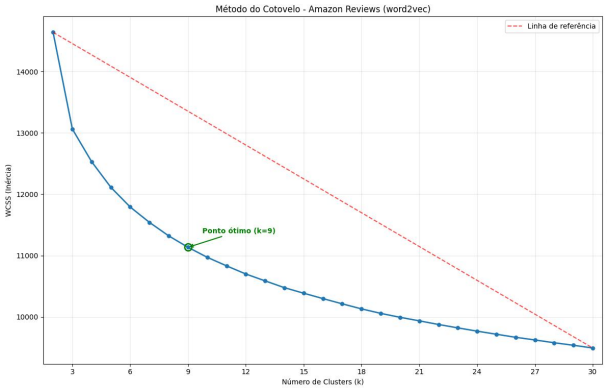


(c) *BBC News Summary — Word2Vec*

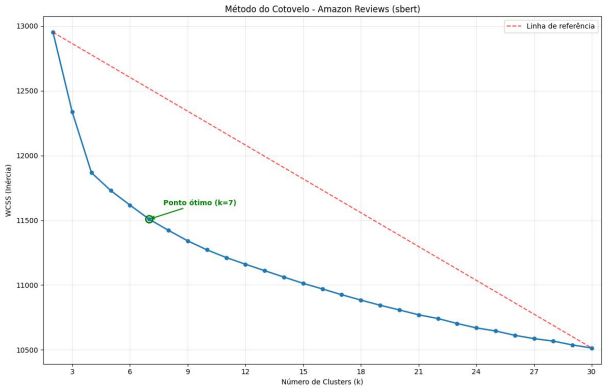


(d) *BBC News Summary* — SBERT

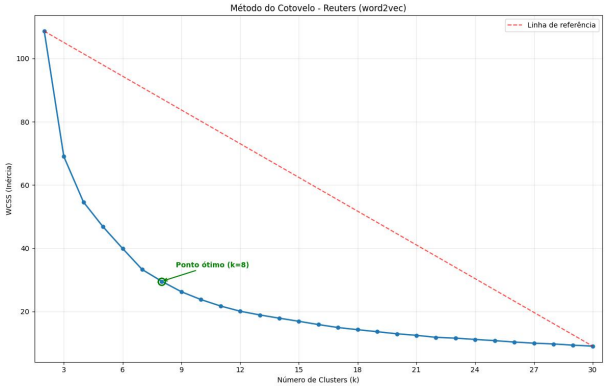
Figura 12 – Gráficos de cotovelo do *K-means* (parte 3). À esquerda: resultados para *Word2Vec*; à direita: resultados para SBERT.



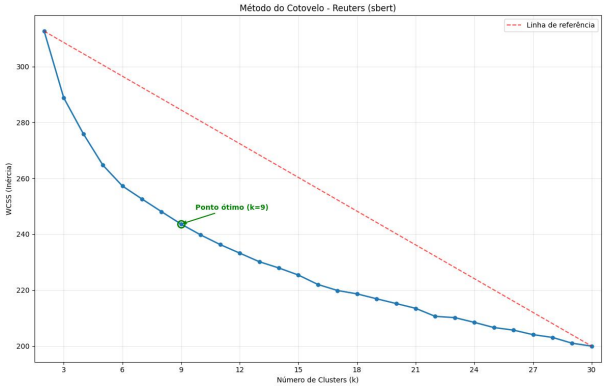
(a) *Amazon US Customer Reviews Dataset — Word2Vec*



(b) *Amazon US Customer Reviews Dataset* — SBERT

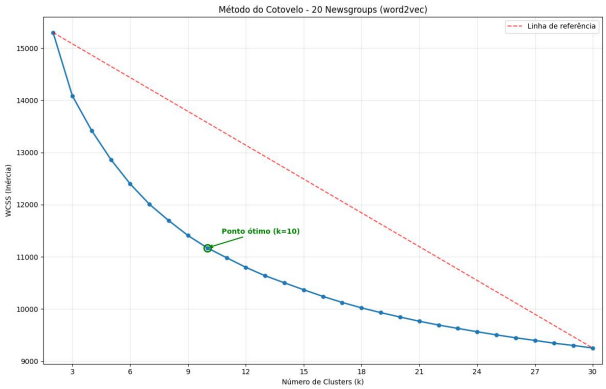


(c) *Reuters-21578 — Word2Vec*

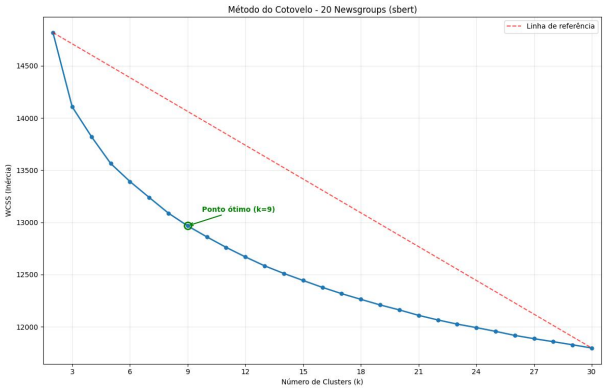


(d) *Reuters-21578* — SBERT

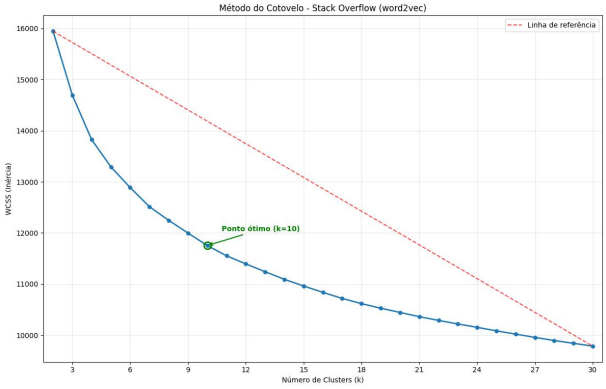
Figura 13 – Gráficos de cotovelo do *K-means* (parte 4). À esquerda: resultados para *Word2Vec*; à direita: resultados para SBERT.



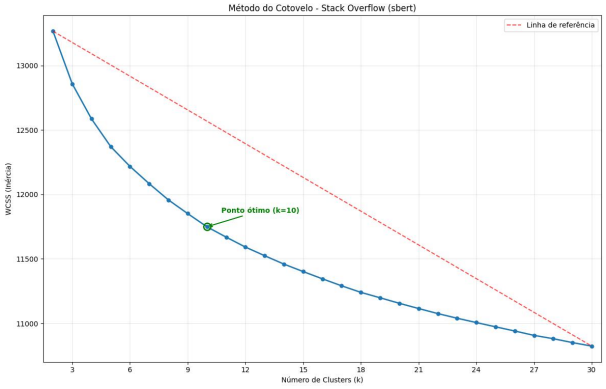
(a) *20 Newsgroups Dataset* — *Word2Vec*



(b) *20 Newsgroups Dataset* — SBERT



(c) *StackOverflow Dataset* — *Word2Vec*



(d) *StackOverflow Dataset* — SBERT

Figura 14 – Gráficos de cotovelo do *K-means* (parte 5). À esquerda: resultados para *Word2Vec*; à direita: resultados para SBERT.

4.5.1.3 Treinamento K-means

Após a definição do valor ideal de K através do método robusto do cotovelo, o algoritmo *K-Means* é executado utilizando os mesmos parâmetros de pré-processamento empregados durante a análise , garantindo consistência entre as fases de seleção e execução final. A inicialização dos centros dos grupos é realizada por meio do método *k-means++*(ARTHUR; VASSILVITSKII, 2007), técnica que seleciona centros iniciais de forma inteligente para acelerar a convergência e melhorar a qualidade dos resultados finais.

O algoritmo procede com a alocação de cada ponto de dado ao centro de grupo mais próximo, utilizando como métrica de similaridade a distância euclidiana aplicada aos vetores pré-processados e normalizados. Essa etapa garante que pontos com características semelhantes sejam agrupados juntos, favorecendo a formação de grupos coesos. Após a alocação, os centros dos grupos são atualizados com base na média aritmética das coordenadas dos pontos que lhes foram atribuídos, reposicionando os centros para refletirem melhor a distribuição atual dos dados.

Este processo iterativo de atribuição e atualização continua até que a variação na inércia entre duas iterações consecutivas seja menor que um limiar de tolerância, definido em 10^{-4}, ou até que o número máximo de 500 iterações seja atingido. O parâmetro *n_init*, que define o número de vezes que o algoritmo será executado com diferentes sementes de centroides, foi configurado como 10, e o resultado final é aquele com a menor inércia. A implementação foi realizada utilizando a função *KMeans* da biblioteca *scikit-learn*[23].

[23] https://scikit-learn.org/1.5/modules/generated/sklearn.cluster.KMeans.html

4.5.2 HDBSCAN

Após a aplicação do *K-means*, foi implementado o HDBSCAN para agrupar as *embeddings* com base na densidade do espaço de dados, uma abordagem que permite a identificação de grupos com formas arbitrárias e a classificação de pontos como ruído (*outliers*). A metodologia adotada envolve uma busca robusta por hiperparâmetros.

4.5.2.1 Otimização de Hiperparâmetros

Diferentemente de algoritmos que exigem a pré-definição do número de grupos, o HDBSCAN o determina automaticamente. O desafio central, portanto, reside na otimização de seus hiperparâmetros. Para isso, foi desenvolvida uma rotina de busca amostrada robusta, que avalia diversas combinações de parâmetros em um espaço de busca abrangente.

A busca por hiperparâmetros é realizada sobre um *grid* expandido, que inclui os seguintes parâmetros:

- *min_cluster_size*: define o número mínimo de pontos necessários para formar um grupo válido. Valores menores resultam em grupos mais granulares, enquanto valores maiores produzem agrupamentos mais conservadores e robustos ao ruído;
- *min_samples*: controla a densidade local necessária para que um ponto seja considerado como pertencente ao núcleo de um grupo. Este parâmetro influencia diretamente a robustez do algoritmo contra *outliers*;
- *cluster_selection_method*: determina a estratégia utilizada para extrair grupos finais da hierarquia de densidade. As opções incluem EOM (Excess of Mass (Excesso de Massa)), que seleciona grupos com maior estabilidade, e *leaf*, que favorece grupos mais específicos nas folhas da árvore hierárquica;
- *cluster_selection_epsilon*: estabelece um limiar de distância para a seleção de grupos, permitindo controle adicional sobre a granularidade dos agrupamentos finais. Valores menores tendem a produzir grupos mais compactos e numerosos.

A avaliação de cada combinação utiliza uma pontuação composta que combina múltiplas métricas: coeficiente da silhueta e o índice de *Davies-Bouldin* A busca incorpora um mecanismo de parada antecipada quando não há melhoria na pontuação após 20 tentativas consecutivas.

Os hiperparâmetros otimizados para cada conjunto de dados e modelo de *embedding* são apresentados nas tabelas 2 para *word2vec* e 3 para SBERT

Tabela 2 – Hiperparâmetros do HDBSCAN otimizados para *embeddings* com *Word2Vec*

conjunto de dados	***min_cluster_size***	***min_samples***	***metric***	***cluster_selection_method***	***cluster_selection_epsilon***
Goodbooks10k	8	8	*euclidean*	*eom*	0.05
Spotify Music Dataset	5	15	*euclidean*	*leaf*	0.05
MovieLens Dataset	3	2	*euclidean*	*eom*	0.05
Steam Games Dataset	5	8	*euclidean*	*leaf*	0.03
RecipeNLG	15	5	*cosine*	*eom*	0.0
BBC	3	3	*euclidean*	*leaf*	0.01
Amazon US Customer Reviews Dataset	10	10	*euclidean*	*eom*	0.01
Reuters-21578 Text Categorization Collection	5	5	*cosine*	*eom*	0.01
20 Newsgroups Dataset	5	5	*cosine*	*eom*	0.01
Stack Overflow Questions Dataset	23	2	*euclidean*	*eom*	0.05

Tabela 3 – Hiperparâmetros do HDBSCAN otimizados para *embeddings* com SBERT

conjunto de dados	*min_cluster_size*	*min_samples*	*metric*	*cluster_selection_method*	*cluster_selection_epsilon*
Goodbooks10k	5	8	*cosine*	*leaf*	0.03
Spotify Music Dataset	5	3	*euclidean*	*eom*	0.03
MovieLens Dataset	5	5	*cosine*	*eom*	0.03
Steam Games Dataset	8	2	*cosine*	*leaf*	0.05
RecipeNLG	8	2	*cosine*	*leaf*	0.05
BBC	5	3	*euclidean*	*eom*	0.03
Amazon US Customer Reviews Dataset	5	8	*cosine*	*leaf*	0.03
Reuters-21578 Text Categorization Collection	3	3	*cosine*	*leaf*	0.01
20 Newsgroups Dataset	10	1	*euclidean*	*leaf*	0.1
Stack Overflow Questions Dataset	3	3	*cosine*	*leaf*	0.01

4.5.2.2 Treinamento do Modelo HDBSCAN

A implementação do algoritmo foi realizada utilizando a biblioteca *hdbscan*[24],onde o modelo final é instanciado com os parâmetros otimizados. e é então aplicado as *embeddings*, o que resulta na construção de uma hierarquia de agrupamentos baseada em densidade.

4.5.3 Self-Organizing Maps (SOM)

O Mapa Auto-Organizável (SOM) é uma rede neural não supervisionada que projeta dados de alta dimensionalidade em um mapa bidimensional, geralmente uma grade retangular. Este processo preserva as relações topológicas dos dados originais, agrupando vetores semelhantes em neurônios vizinhos na grade. A implementação adotada neste trabalho utiliza uma *pipeline* automatizada para a otimização de hiperparâmetros e treinamento do modelo.

4.5.3.1 Otimização de Hiperparâmetros

Foi implementada uma busca por hiperparâmetros que avalia um vasto espaço de configurações para identificar a mais adequada para os dados.

O espaço de busca inclui múltiplos parâmetros essenciais:

- **Tamanho da Grade (*Grid Size*)**: define as dimensões da grade bidimensional de neurônios que compõe o mapa auto-organizável. Uma heurística propõe múltiplos tamanhos de grade candidatos, tanto quadrados quanto retangulares, com base no volume e na dimensionalidade dos dados. Grades maiores proporcionam maior resolução e capacidade de representação, mas requerem mais dados para treinamento adequado;
- **Taxa de Aprendizado (*learning_rate*)**: controla a intensidade das atualizações dos pesos dos neurônios durante o treinamento. Valores altos aceleram o aprendizado inicial, mas podem causar instabilidade, enquanto valores baixos garantem convergência mais estável, porém mais lenta;
- **Raio de Vizinhança Inicial (*sigma*)**: determina o tamanho inicial da região de influência ao redor do neurônio vencedor. Este parâmetro decresce ao longo do treinamento, iniciando com valores altos para formar a estrutura topológica global e reduzindo gradualmente para refinamentos locais;
- **Função de Vizinhança**: define como a influência do neurônio vencedor se propaga para seus vizinhos na grade. As opções incluem *gaussian*, que produz transições suaves e preserva melhor a topologia, e *bubble*, que aplica influência uniforme dentro de um raio fixo, resultando em fronteiras mais definidas entre regiões;
- **Número de Épocas (*epochs*)**: especifica quantas vezes o conjunto completo de dados será apresentado à rede durante o treinamento. Um número insuficiente pode resultar em

[24] https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html

convergência incompleta, enquanto excesso pode levar ao *overfitting* e perda da capacidade de generalização.

A avaliação de cada combinação de parâmetros é realizada utilizando o **Coeficiente da Silhueta**, onde a configuração que maximiza a mesma é selecionada como ótima para o treinamento final. As configurações ótimas estão representadas nas tabelas: 4 para *word2vec* e 5 para SBERT

Tabela 4 – Hiperparâmetros otimizados do SOM para *embeddings* com *Word2Vec*

conjunto de dados	***grid_size***	***sigma***	***learning_rate***	***neighborhood_function***	***epochs***
Goodbooks10k	(36, 24)	1.0	0.5	*gaussian*	2000.0
Spotify Music Dataset	(30, 20)	1.5	0.25	*gaussian*	2000.0
MovieLens Dataset	(18, 12)	1.0	0.25	*gaussian*	600
Steam Games Dataset	(24, 16)	1.0	0.05	*bubble*	600
RecipeNLG	(30, 20)	1.0	0.1	*bubble*	600
BBC	(18, 12)	1.0	0.25	*gaussian*	300
Amazon US Customer Reviews Dataset	(30, 20)	1.0	0.1	*bubble*	300.0
Reuters-21578 Text Categorization Collection	(14, 9)	1.5	0.25	*gaussian*	1000.0
20 Newsgroups Dataset	(20, 30)	0.5	0.25	*bubble*	300
Stack Overflow Questions Dataset	(20, 20)	1.5	0.25	*gaussian*	600

Tabela 5 – Hiperparâmetros otimizados do SOM para *embeddings* com SBERT

conjunto de dados	***grid_size***	***sigma***	***learning_rate***	***neighborhood_function***	***epochs***
Goodbooks10k	(25, 25)	1.5	0.1	*gaussian*	1500.0
Spotify Music Dataset	(30, 20)	1.5	0.25	*gaussian*	2000.0
MovieLens Dataset	(24, 16)	1.5	0.05	*gaussian*	300
Steam Games Dataset	(24, 16)	1.0	0.05	*bubble*	600
RecipeNLG	(30, 20)	1.0	0.1	*bubble*	300
BBC	(18, 12)	1.0	0.25	*gaussian*	300
Amazon US Customer Reviews Dataset	(30, 20)	1.0	0.1	*bubble*	300.0
Reuters-21578 Text Categorization Collection	(14, 9)	1.5	0.25	*gaussian*	1000.0
20 Newsgroups Dataset	(36, 24)	1	0.1	*bubble*	300
Stack Overflow Questions Dataset	(20, 20)	1.5	0.25	*gaussian*	600

4.5.3.2 Treinamento do SOM

Com os melhores hiperparâmetros definidos, o modelo SOM final é instanciado utilizando a biblioteca *MiniSom*[25]. Os vetores de peso dos neurônios são inicializados de forma inteligente através da Análise de Componentes Principais (*PCA weights init*), uma técnica que acelera a convergência em comparação com a inicialização aleatória, pois posiciona os vetores de peso ao longo dos dois primeiros componentes principais dos dados. O modelo é então treinado com os dados de *embeddings* conforme a melhor estratégia encontrada na fase de otimização.

Durante o processo iterativo, para cada *embedding* é identificado o BMU através da distância euclidiana mínima entre o vetor de entrada e os pesos dos neurônios. Os pesos do BMU e de seus neurônios vizinhos são atualizados segundo a função de vizinhança otimizada (*gaussian*, *bubble* ou *mexican_hat*) e taxa de aprendizado selecionadas, ambas decaindo

4.5.4 Agrupamento Hierárquico Aglomerativo

4.5.4.1 Seleção de Parâmetros e Otimização do Modelo

A eficácia do HCA depende da escolha de seus hiperparâmetros, principalmente o número de grupos, o método de ligação (*linkage*) e a métrica de distância. A abordagem adotada substituiu a análise manual por um processo automatizado em múltiplas etapas.

Inicialmente, os dados passam por um pré-processamento para garantir a estabilidade numérica, onde valores nulos são tratados. Em seguida, o número ótimo de grupos (k) é estimado por meio do método do cotovelo.

[25] https://github.com/JustGlowing/minisom

Com o número de grupos definido, uma segunda etapa de otimização é realizada para encontrar a melhor combinação de **método de ligação** e **métrica de distância**. Uma busca amostrada avalia as seguintes combinações:

- **Método de Ligação (*linkage*)**: define o critério utilizado para calcular a distância entre grupos durante o processo de fusão hierárquica. As opções incluem *ward*, que minimiza a variância intra-grupo e é adequado para grupos esféricos e de tamanhos similares; *average*, que utiliza a distância média entre todos os pares de pontos dos grupos, proporcionando comportamento equilibrado; e *complete*, que considera a distância máxima entre pontos de diferentes grupos, favorecendo grupos compactos;
- **Métrica de Distância**: determina como a similaridade entre pontos de dados é quantificada. As métricas disponíveis incluem *euclidean*, que mede a distância geométrica direta e é sensível à magnitude das características; *cosine*, que avalia o ângulo entre vetores, sendo invariante à magnitude e adequada para dados de alta dimensionalidade; e *manhattan*, que calcula a soma das diferenças absolutas entre coordenadas, sendo mais robusta a *outliers* que a distância euclidiana.

A avaliação de cada combinação é feita utilizando o coeficiente de silhueta.

Os hiperparâmetros otimizados para cada conjunto de dados e modelo de *embedding* são apresentados nas tabelas 6 para *word2vec* e 7 para SBERT

Tabela 6 – Hiperparâmetros do HCA otimizados para *embeddings* com *Word2Vec*

conjunto de dados	***n_clusters***	***metric***	***linkage***
Goodbooks10k	18	*cosine*	*complete*
Spotify Music Dataset	18	*cosine*	*complete*
MovieLens Dataset	10	*cosine*	*average*
Steam Games Dataset	14	*euclidean*	*ward*
RecipeNLG	13	*cosine*	*complete*
BBC	19	*cosine*	*complete*
Reuters-21578 Text Categorization Collection	14	*cosine*	*complete*
20 Newsgroups Dataset	17	*euclidean*	*ward*
Stack Overflow Questions Dataset	16	*euclidean*	*ward*

Tabela 7 – Hiperparâmetros do HCA otimizados para *embeddings* com SBERT

conjunto de dados	***n_clusters***	***metric***	***linkage***
Goodbooks10k	11	*cosine*	*complete*
Spotify Music Dataset	18	*euclidean*	*ward*
MovieLens Dataset	11	*cosine*	*complete*
Steam Games Dataset	12	*euclidean*	*ward*
RecipeNLG	18	*euclidean*	*ward*
BBC	12	*cosine*	*complete*
Reuters-21578 Text Categorization Collection	5	*cosine*	*average*
20 Newsgroups Dataset	20	*euclidean*	*ward*
Stack Overflow Questions Dataset	14	*euclidean*	*ward*

4.5.4.2 Treinamento HCA

Após a conclusão da otimização, o modelo HCA final é treinado utilizando a função *AgglomerativeClustering* da biblioteca *scikit-learn*[26], utilizando o número de grupos, o método de ligação e a métrica de distância encontrados na etapa de otimização do modelo. Onde o algoritmo começará tratando cada elemento como um grupo individual e, em cada etapa subsequente, une os dois grupos mais semelhantes até que todos os elementos estejam unidos em uma estrutura única e, em cada etapa subsequente, unindo os dois grupos mais similares segundo a métrica e *linkage* otimizados, até atingir o número desejado de grupos.

[26] https://scikit-learn.org/dev/modules/generated/sklearn.cluster.AgglomerativeClustering.html

5 EXPERIMENTOS E RESULTADOS

5.1 *GOODBOOKS10K*

Após determinar os hiperparâmetros ótimos para o conjunto *Goodbooks10k*, aplicamos cada método de agrupamento com os parâmetros selecionados. Os resultados estão apresentados na Tabela 8, onde percebe-se que a combinação **SOM + *Word2Vec*** obteve melhor desempenho, pois títulos de livros funcionam como identificadores lexicais concisos que frequentemente condensam sinais temáticos e estilísticos, como por exemplo *dragon, magic e chronicles* ocorrendo em títulos de fantasia, enquanto *murder, death, mystery* aparecem com frequência em títulos de suspense. O *Word2Vec* captura essas coocorrências distributivas e gera representações que refletem proximidade semântica entre termos.

O SOM complementa essa representação ao organizar os vetores em um mapa topológico bidimensional que preserva relações de vizinhança, permitindo transições graduais entre regiões temáticas em vez de fronteiras categóricas rígidas.

Tabela 8 – Resultados agrupamento *Goodbooks10k*

Embedding	**Algoritmo**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
SBERT	HDBSCAN	106	0.254	1.439	0.092	-
	K-Means	10	0.011	4.623	-	-
	HCA	11	-0.012	7.401	-	-
	SOM	22	-0.024	4.775	-	-
Word2Vec	HDBSCAN	90	0.314	0.241	0.098	-
	K-Means	9	-0.076	3.647	-	-
	HCA	18	-0.259	4.078	-	-
	SOM	**29**	**0.761**	**0.870**	**-**	**-**

5.2 *SPOTIFY MUSIC CONJUNTO DE DADOS*

O mesmo foi feito para o conjunto *Spotify Music conjunto de dados*, e observando os resultados da tabela 9, podemos perceber que a melhor combinação para este conjunto de dados é **HDBSCAN + *Word2Vec***.

Em relação ao *embedding*, isso ocorre porque a riqueza semântica pré-treinada do SBERT introduziu nuances que eram irrelevantes para a estrutura organizacional natural presente nos dados. O conhecimento semântico complexo do SBERT ofuscou padrões lexicais simples, porém informativos, que eram cruciais para revelar a organização latente do conjunto.

Quanto aos algoritmos de agrupamento, o HDBSCAN mostrou-se superior por sua capacidade de revelar a estrutura de densidade intrínseca dos dados. Ao contrário do *K-Means*, que impõe partições rígidas, e do SOM, que pressupõe uma topologia gradual, o HDBSCAN identificou automaticamente grupos com densidades variáveis, além de detectar automaticamente *outliers*, e esta capacidade é muito útil neste conjunto de dados, pois quando se trata de músicas, tende-se a ter muito mais música de um gênero, do que de outro, como, por exemplo, há mais músicas de *pop* do que *death metal*, assim como outras são automaticamente classificadas como *outlier* por conter tão poucas músicas.

Os valores do ARI obtidos, embora aparentemente baixos, são considerados aceitáveis para este conjunto de dados, pois a análise se baseia exclusivamente em dados textuais limitados (*track_name* e *track_artist*) em um domínio onde a classificação musical é naturalmente ambígua e os rótulos de gênero podem não refletir perfeitamente as similaridades semânticas dos nomes das músicas, como, por exemplo, nas músicas *Bad Ideia Right?* da cantora *Olivia Rodrigo* e a música *Bad Idea!* da cantora *Girl in Red*, são muito parecidas entre si, diferenciando de apenas uma palavra basicamente, porém uma é do gênero *pop* e outra *indie*, assim demonstrando que não tem como basear o gênero de uma música apenas pelo seu título

Tabela 9 – Resultados agrupamento *Spotify*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	35	**44**	**0.751**	**0.023**	**0.218**	**0.290**
	HCA	35	2	-0.0727	0.131	-	0.067
	SOM	35	2	0.0547	0.770	-	0.045
	K-Means	35	11	0.040	2.830	-	0.070
SBERT	HDBSCAN	35	312	0.252	1.452	0.556	0.069
	HCA	35	2	0.028	5.774	-	0.052
	K-Means	35	7	0.016	5.306	-	0.046
	SOM	35	22	0.007	4.051	-	0.018

5.3 *MOVIELENS CONJUNTO DE DADOS*

De acordo com a tabela 10 a combinação ***Word2Vec* + HDBSCAN** demonstrou desempenho excepcional no conjunto de dados *MovieLens* devido à natureza específica das *tags* de usuários, que formam uma *folksonomia* (sistema de classificação colaborativa onde usuários atribuem livremente palavras-chave (*tags*) a recursos) coletiva com vocabulário distintivo por gênero cinematográfico. O *Word2Vec* mostrou-se particularmente eficaz em capturar as assinaturas lexicais emergentes dessas *tags* colaborativas, onde usuários naturalmente convergiram para termos similares ao descrever filmes do mesmo gênero (por exemplo, *"scary"*, *"creepy"*, *"suspense"* para horror; *"funny"*, *"hilarious"*, *"humor"* para comédia). Esta abordagem de *embedding* conseguiu preservar as relações semânticas simples, mas distintivas presentes no vocabulário coletivo, resultando em representações vetoriais que refletem adequadamente os padrões de categorização emergentes da comunidade de usuários. O modelo baseado em co-ocorrência de palavras capturou efetivamente a sabedoria coletiva expressa nas *tags*, diferentemente do SBERT que, treinado em textos formais e estruturados, teve dificuldades com a linguagem coloquial e inconsistente das *tags*, gerando representações inadequadas para a hibridez entre títulos formais de filmes e metadados sociais informais.

O HDBSCAN complementou perfeitamente essas representações ao lidar eficientemente com a densidade variável inerente aos dados do *MovieLens*, onde filmes populares possuem abundantes *tags* (formando grupos densos) enquanto filmes menos conhecidos apresentam *tags* esparsas ou inconsistentes (apropriadamente identificados como ruído de apenas 13,8%). Diferentemente de algoritmos tradicionais como *K-Means*, que assumem grupos esféricos de tamanhos similares, o HDBSCAN conseguiu identificar grupos de formas arbitrárias e densidades heterogêneas, características fundamentais em dados de *folksonomia* onde a popularidade e consenso variam drasticamente entre diferentes gêneros e subgêneros cinematográficos, onde assim como no *spotify* a popularidade do filme define também o numero de *tags* aplicadas a ele, onde filmes como *Senhor dos aneis* tem muito mais informações atribuidas, do que um filme de fetival de cinemas mais desconhecidos.

Ao contrario do *Goodbooks10k*, aqui o SOM não se deu melhor, pois títulos de filmes não tendem a ter tantas informações quanto títulos de livros, eles são mais comerciais, assim ficando difícil de saber seus gêneros e subgêneros, como, por exemplo, o livro *Androides Sonham com ovelhas elétricas?*, onde pelas palavras *Androides* e *Ovelhas elétricas* percebe-se que se trata de um livro de ficção científica, enquanto ao ser adaptado para filme o mesmo foi chamado de *Blade Runner*, onde pelo título mais comercial não se consegue retirar nenhuma informação de seu gênero

Os valores do ARI obtidos, embora aparentemente modestos, são considerados satisfatórios para este conjunto de dados, uma vez que a análise se fundamenta exclusivamente em dados textuais limitados (títulos dos filmes e tags de usuários) em um domínio onde a classificação cinematográfica é naturalmente ambígua e os rótulos de gênero podem não refletir adequadamente as similaridades semânticas presentes. A natureza intrinsecamente complexa dos dados cinematográficos, em que um único filme frequentemente pertence a múltiplos gêneros simultaneamente (como *"Action|Adventure|Sci-Fi"*), aliada à informação semântica disponível nos títulos, configura um cenário desafiador que naturalmente resulta em métricas de concordância mais baixas. Adicionalmente, a métrica ARI é particularmente rigorosa em cenários de

múltipla associação categórica, penalizando severamente agrupamentos que, mesmo sendo semanticamente coerentes, não coincidem perfeitamente com as taxonomias de referência predefinidas.

Tabela 10 – Resultados agrupamento *MovieLens*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	95	**65**	**0.872**	**0.699**	**0.025**	**0.130**
	HCA	95	10	0.317	3.063	-	0.006
	SOM	95	31	0.389	1.843	-	0.085
	K-Means	95	8	0.306	3.185	-	0.096
SBERT	HDBSCAN	95	156	0.182	1.684	-0.027	0.011
	HCA	95	11	-0.004	13.271	-	0.006
	K-Means	95	12	0.016	5.471	-	0.011
	SOM	95	138	0.000	4.427	-	0.006

5.4 *STEAM GAMES CONJUNTO DE DADOS*

Como visto na tabela 11 a combinação *Word2Vec* + HDBSCAN alcançou performance superior devido à sinergia entre a capacidade do *Word2Vec* de capitalizar sobre a redundância informativa estruturada do conjunto de dados e a aptidão do HDBSCAN para revelar a topologia de densidade natural do domínio. O *Word2Vec* demonstrou-se ideal ao explorar as coocorrências sistemáticas presentes nos campos ***Name***, ***Developers*** e ***Tags***. Essa estrutura de três camadas, onde o nome de um jogo, seu desenvolvedor e suas *tags* descritivas se reforçam mutuamente, amplifica os padrões lexicais.

O HDBSCAN complementou perfeitamente esses *embeddings* por sua habilidade de identificar agrupamentos de densidade variável, uma característica intrínseca do mercado de jogos, onde gêneros populares formam grupos grandes e densos, enquanto nichos criam grupos menores e mais esparsos. Diferentemente de algoritmos que impõem estruturas rígidas, o HDBSCAN descobriu a organização natural dos gêneros sem a necessidade de pré-especificar o número de grupos. Adicionalmente, sua capacidade de tratar uma porção significativa dos dados como ruído (62,8%) foi fundamental, permitindo que jogos experimentais ou que mesclam múltiplos gêneros fossem corretamente classificados como *outliers*, em vez de serem forçados a categorias inadequadas, o que preservou a pureza e a definição dos grupos resultantes.

O índice de concordância externa alcançado apresenta-se substancialmente baixo, indicando limitações na capacidade do agrupamento em reproduzir fielmente a taxonomia de gêneros estabelecida no *conjunto de dados* Steam. Esta discrepância entre as métricas internas favoráveis (*Silhouette Score*, *Davies-Bouldin* e *DBCV*) e o baixo ARI sugere que, embora o algoritmo tenha identificado estruturas coesas e bem separadas no espaço de *embeddings*, tais agrupamentos não se alinham adequadamente com as categorias genéricas predefinidas. A fragmentação excessiva dos dados, evidenciada pela classificação de 62,8% das instâncias como ruído, contribui significativamente para esta divergência, uma vez que o HDBSCAN priorizou a pureza dos *clusters* em detrimento da cobertura categórica abrangente.

Tabela 11 – Resultados agrupamento *Steam Games conjunto de dados*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	48	**144**	**0.842**	**0.792**	**0.062**	**0.096**
	HCA	48	14	0.231	1.983	-	0.085
	SOM	48	64	0.218	1.769	-	0.025
	K-Means	48	10	0.258	1.837	-	0.021
SBERT	HDBSCAN	48	133	0.234	1.572	0.058	0.019
	HCA	48	12	0.015	4.438	-	0.040
	K-Means	48	9	0.046	3.562	-	0.052
	SOM	48	46	0.021	3.318	-	0.037

5.5 *RECIPENLG*

A combinação *Word2Vec* + HDBSCAN demonstrou performance superior no conjunto de dados *RecipeNLG*, como visto na tabela 12 devido à natureza híbrida específica da estrutura informativa culinária. O conjunto de dados apresenta uma dualidade informativa única, onde o título da receita fornece contexto

semântico rico (*"Mediterranean Quinoa Salad with Feta and Olives"*) enquanto a lista de ingredientes oferece vocabulário lexicalmente específico e distintivo (*"quinoa", "feta cheese", "olives", "olive oil"*). Esta combinação título+ingredientes cria redundância informativa benéfica, onde o *Word2Vec* consegue capturar efetivamente as co-ocorrências sistemáticas entre ingredientes específicos que caracterizam diferentes tradições culinárias, como as associações entre *"soy sauce"*, *"ginger"* e *"rice"* para cozinha asiática, ou "brasil", "oregano"e *"tomato"* para cozinha mediterrânea. O HDBSCAN complementa esta capacidade ao identificar clusters de densidade variável correspondentes aos diferentes tipos de cozinha sem necessidade de especificação prévia do número de grupos.

A eficácia desta combinação também se explica pela estrutura semi-estruturada do domínio culinário, que apresenta tradições gastronômicas discretas, mas com sobreposições controladas devido adaptações regionais. Diferentemente de domínios completamente heterogêneos, o vocabulário culinário possui especificidade moderada onde ingredientes servem como assinaturas distintivas de diferentes tradições, embora com compartilhamentos naturais (sal, açúcar, farinha, etc.). O *Word2Vec* consegue mapear essas relações distributivas entre ingredientes e estilos culinários de forma mais precisa que embeddings semânticos complexos, enquanto o HDBSCAN lida adequadamente com a taxa de ruído moderada (18.6%) característica de dados culinários, onde receitas híbridas e adaptações regionais são apropriadamente identificadas como *outliers*. A estrutura resultante permite descoberta natural de agrupamentos culinários sem necessidade de generalização semântica excessiva que poderia mascarar especificidades de ingredientes distintivos.

Tabela 12 – Resultados agrupamento *RecipeNLG*

Embedding	**Algoritmo**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	**40**	**0.229**	**1.104**	**0.051**	**-**
	K-Means	8	0.134	2.062	-	-
	HCA	13	0.073	2.597	-	-
	SOM	49	-0.123	2.770	-	-
SBERT	HDBSCAN	47	0.147	1.845	-0.131	-
	K-Means	11	0.041	3.710	-	-
	HCA	18	0.022	3.895	-	-
	SOM	47	0.026	3.229	-	-

5.6 *BBC*

De acordo com a tabela 13 a combinação *K-Means* + *Word2Vec* demonstrou performance superior no conjunto de dados *BBC News Summary* devido à natureza homogênea e estruturalmente consistente do conteúdo jornalístico. O conjunto de dados apresenta características fundamentais que favorecem abordagens lexicais: vocabulário especializado por categoria (*Sports: "player", "team", "match"; Technology: "software", "digital", "internet"; Politics: "government", "minister", "policy"*), estilo jornalístico padronizado da *BBC*, e cinco categorias bem definidas e lexicalmente distintas. Esta estrutura permite que o *Word2Vec* capture efetivamente as assinaturas lexicais específicas de cada domínio noticioso, criando representações vetoriais onde diferentes tópicos ocupam regiões bem separadas do espaço semântico. O algoritmo consegue identificar que notícias esportivas consistentemente utilizam vocabulário distinto de notícias tecnológicas ou políticas, criando clusters baseados em co-ocorrências distributivas características de cada área.

A superioridade do *K-Means* sobre algoritmos baseados em densidade como HDBSCAN se explica pela natureza balanceada e estruturada das categorias jornalísticas, que se alinham perfeitamente com o pressuposto de clusters esféricos e de tamanho similar assumido pelo algoritmo. Diferentemente de dados heterogêneos onde a densidade variável é crucial, as cinco categorias da *BBC* (Business, Entertainment, Politics, Sport, Tech) formam agrupamentos naturalmente equilibrados que se beneficiam da abordagem de centroide. O *K-Means* consegue posicionar centroides de forma ótima no espaço *Word2Vec*, aproveitando a separabilidade lexical clara entre domínios jornalísticos sem a fragmentação excessiva característica de

métodos de densidade que podem identificar subtópicos desnecessários. A combinação resulta em clusters bem separados que correspondem diretamente às categorias editoriais estabelecidas, evitando tanto a *over-segmentation* quanto a generalização semântica excessiva que poderia mascarar distinções lexicais importantes entre diferentes áreas noticiosas.

O coeficiente de concordância externa obtido pela combinação **K-Means + Word2Vec** demonstra desempenho robusto e alinhado com as expectativas para este tipo de conjunto de dados jornalístico estruturado. Este resultado reflete as características favoráveis inerentes ao conjunto de dados *BBC News Summary*, onde as cinco categorias temáticas apresentam vocabulários especializados e distinções lexicais bem definidas, facilitando a identificação de padrões distributivos característicos por domínio. A natureza curada do conteúdo jornalístico da *BBC*, com sua consistência estilística e terminológica, cria condições ideais para algoritmos de agrupamento baseados em similaridade semântica. A combinação **HDBSCAN + SBERT** alcançou o melhor índice de concordância externa, indicando superior capacidade de reproduzir a taxonomia categórica original do conjunto de dados. Contudo, esta superioridade no alinhamento categórico foi acompanhada por métricas internas menos favoráveis; esta discrepância sugere que, embora o HDBSCAN tenha identificado agrupamentos mais fidedignos às categorias jornalísticas estabelecidas, a estrutura destes grupos apresenta menor coesão interna, possivelmente devido à natureza baseada em densidade do algoritmo que pode gerar grupos com geometrias irregulares.

Tabela 13 – Resultados agrupamento *BBC News Summary*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	HDBSCAN	5	55	0.279	1.194	-0.162	0.141
	K-Means	5	**8**	**0.313**	**1.082**	-	0.533
	HCA	5	19	0.106	1.524	-	0.491
	SOM	5	38	0.131	1.643	-	0.226
SBERT	HDBSCAN	5	43	0.138	1.666	**0.027**	**0.647**
	K-Means	5	9	0.061	3.785	-	0.634
	HCA	5	12	0.023	4.871	-	0.463
	SOM	5	47	0.020	3.081	-	0.279

5.7 *AMAZON US CUSTOMER REVIEWS CONJUNTO DE DADOS*

Através da tabela 14, percebe-se que o SBERT se destacou pela primeira vez em algumas pontuações, como *Silhouette* e numero de grupos achados próximo a número de grupo ideal, devido à natureza heterogênea e semanticamente complexa do conteúdo de *e-commerce*, por conta de conter produtos de diversas categorias (eletrônicos, livros, roupas) e uma grande variabilidade linguística, indo desde textos técnicos até expressões coloquiais. Esta estrutura permite que o SBERT capture efetivamente nuances contextuais complexas onde o significado emerge da combinação de palavras em contexto específico, diferentemente de abordagens lexicais que perdem informação semântica crucial ao fazer médias de embeddings de palavras individuais. O SBERT consegue distinguir que "produto excelente"(referindo-se à qualidade) e "atendimento excelente"(referindo-se a serviço) representam aspectos semanticamente distintos, mesmo compartilhando vocabulário comum.

A eficácia do HDBSCAN complementa perfeitamente esta capacidade semântica ao lidar com a estrutura natural de densidade variável característica de dados de *e-commerce*, onde produtos populares geram clusters densos de *reviews* enquanto produtos nichos formam agrupamentos esparsos. O algoritmo consegue identificar apropriadamente *outliers* (26.15% de ruído) que representam *reviews* atípicas, *spam* ou conteúdo genuinamente único.

O índice de concordância externa alcançado pelo ***HDBSCAN+ Word2Vec***, apesar de estar com um valor alto, o número de grupos achados está muito longe do número de grupo ideal, e isso se dá, pois neste conjunto de dados, 2 categorias apenas(Livro e tecnologia) representa quase *40%* da base inteira, assim ao separar em apenas dois grupos você tem chance de embarcar quase metade do conjunto de dados, e por conta deste baixo numero de grupos achados, o SBERT + HDBSCAN foi considerado a

melhor opção, pois é o que mais se aproximou do número de grupos ideais e não ficou pior em todas as métricas, quando comparado com a combinação com Word2vec

Tabela 14 – Resultados agrupamento *Amazon Product Reviews*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	HDBSCAN	125	2	0.026	**0.176**	**0.092**	**0.598**
	K-Means	125	9	0.077	2.824	-	0.456
	HCA	125	18	-0.059	9.330	-	0.264
	SOM	125	47	-0.090	3.389	-	0.153
SBERT	**HDBSCAN**	125	**160**	**0.244**	1.437	0.061	0.046
	K-Means	125	7	0.037	4.471	-	0.454
	SOM	125	41	0.009	3.754	-	0.173
	HCA	125	15	0.006	7.494	-	0.285

5.8 *REUTERS-21578 TEXT CATEGORIZATION COLLECTION*

Já para este conjunto de dados, conforme a tabela 15, percebemos que o *word2vec* voltou a performar melhor, e isto se dá por conta da natureza estruturada e padronizada do conteúdo jornalístico-financeiro, cujo é altamente padronizado, seguindo convenções editoriais globais.

Além disto, o HDBSCAN performou melhor, por conseguir gerar agrupamentos correspondentes aos diferentes tópicos econômicos sem necessidade de especificação prévia do número de categorias. A taxa de ruído muito baixa (12.0%) confirma a estrutura editorial organizada característica de agências de notícias profissionais como a *Reuters*, onde esta capacidade de identificar os 12% de tópicos fora do padrão garantiu que os grupos formados fossem mais homogêneos e fiéis aos tópicos.

Tabela 15 – Resultados agrupamento *REUTERS-21578 TEXT CATEGORIZATION COLLECTION*

Embedding	**Algoritmo**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	2	**0.742**	**0.431**	**0.062**	-
	HCA	2	0.678	0.517	-	-
	K-Means	6	0.300	0.991	-	-
	SOM	14	0.241	0.983	-	-
SBERT	HDBSCAN	29	0.233	1.338	-0.012	-
	HCA	2	0.064	3.696	-	-
	K-Means	6	0.064	3.313	-	-
	SOM	21	-0.026	1.305	-	-

5.9 *20 NEWSGROUPS CONJUNTO DE DADOS*

Aqui, para este conjunto de dados o *word2vec* performou melhor pelos mesmos motivos apresentados na seção 5.8 como visto na tabela 16.

Porém, os resultados obtidos revelam uma divergência entre métricas internas e externas, onde o HCA demonstrou superioridade em coesão interna (*Silhouette* e *Davies-Bouldin*), enquanto o *K-Means* alcançou maior concordância externa. Esta divergência sugere que, embora o HCA tenha conseguido identificar estruturas hierárquicas semanticamente coerentes explorando as macro-categorias naturais (*comp.**, *sci.**, *talk.**), sua abordagem aglomerativa pode ter gerado agrupamentos que, apesar de internamente consistentes, não se alinham perfeitamente com as 20 categorias específicas. O *K-Means*, por sua vez, conseguiu uma segmentação mais direta que corresponde melhor às categorias individuais, mesmo que com menor pureza interna. Para aplicações práticas, a preferência dependeria do objetivo específico: HCA para descoberta de estruturas semânticas latentes e *K-Means* para reprodução fiel de categorias estabelecidas. Os valores de ARI obtidos, embora moderados, são compreensíveis considerando a natureza conversacional e informal dos *newsgroups*, onde a linguagem coloquial, gírias e variabilidade estilística individual introduzem ruído semântico que dificulta a segmentação categórica precisa.

Tabela 16 – Resultados agrupamento *The 20 newsgroups text conjunto de dados*

Embedding	**Algoritmo**	**N° Grupos ideal**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HCA**	20	2	**0.815**	**0.130**	-	0.161
	HDBSCAN	20	2	0.178	1.160	-0.524	0.006
	K-Means	20	**8**	0.064	2.720	-	**0.262**
	SOM	20	83	-0.069	2.486	-	0.202
SBERT	*K-Means*	20	**8**	0.034	4.779	-	0.256
	HDBSCAN	20	2	0.016	2.007	**-0.425**	0.000
	HCA	20	6	0.012	7.548	-	0.134
	SOM	20	46	-0.005	3.628	-	0.195

5.10 *STACK OVERFLOW QUESTIONS CONJUNTO DE DADOS*

Como observado pelos resultados da tabela 17, a estrutura do *Stack Overflow* cria um cenário de "reforço léxico", onde um problema técnico é apresentado de forma concisa no título e, em seguida, detalhado no corpo utilizando a mesma terminologia especializada. Termos como *"API"*, *"function"*, *"JavaScript"* ou *"Python"* possuem co-ocorrências consistentes e previsíveis, capturadas com alta fidelidade pelo modelo distributivo do *Word2Vec*. Essa redundância informativa e o vocabulário técnico padronizado permitiram a criação de *embeddings* onde a distância vetorial refletia com precisão a similaridade técnica entre as perguntas

A eficácia do HDBSCAN complementa esta capacidade lexical ao lidar adequadamente com a taxa de ruído extremamente alta (93.2%) característica de domínios técnicos ultra-especializados, onde cada pergunta representa individualização técnica muito específica envolvendo combinações únicas de linguagens, *frameworks* e problemas particulares. Diferentemente de domínios criativos onde alta individualização favorece embeddings semânticos, a individualização técnica do *Stack Overflow* mantém consistência terminológica que o *Word2Vec* consegue estruturar efetivamente. O HDBSCAN identifica apropriadamente que a vasta maioria das perguntas são *outliers* técnicos únicos, mas quando clusters são formados, representam comunidades coerentes de problemas técnicos específicos (por exemplo, questões relacionadas a *React hooks*, problemas de *SQL queries*, ou *debugging* de *APIs*).

E além disto, o HDBSCAN se saiu melhor justamente por conseguir gerar grupos de densidades diferentes, onde neste conjunto de dados perguntas com *python* podem ser muito mais frequentes do que perguntas com *COBOL*, dado a popularidade de ambas linguagens de programação, assim ambos grupos teriam densidades diferentes

Tabela 17 – Resultados agrupamento *Stack Overflow Questions conjunto de dados*

Embedding	**Algoritmo**	**N° Grupos achados**	***Silhouette Score*** ↑	***Davies-Bouldin Score*** ↓	**DBCV** ↑	**ARI** ↑
Word2Vec	**HDBSCAN**	201	**0.236**	**1.200**	**0.046**	-
	HCA	16	0.020	3.096	-	-
	SOM	141	0.037	2.681	-	-
	K-Means	10	0.086	2.410	-	-
SBERT	HDBSCAN	210	0.160	1.504	0.011	-
	HCA	14	-0.007	5.091	-	-
	K-Means	10	0.018	4.035	-	-
	SOM	141	0.003	3.938	-	-

5.11 CONCLUSÃO

A análise comparativa indicou que não há uma combinação universalmente superior de *embedding* e algoritmo de agrupamento para dados textuais. Contudo, a observação dos resultados permitiu identificar padrões sistemáticos que correlacionam as características estruturais dos *corpora* com o desempenho dos métodos avaliados.

5.11.1 Desempenho de Algoritmos Baseados em Densidade e Partição

Houve uma distinção entre o desempenho de algoritmos baseados em densidade (HDBSCAN) e os baseados em partição ou centroide (*K-Means*, HCA, SOM). O HDBSCAN apresentou o desempenho mais consistente, sendo a melhor opção em 72% dos casos avaliados. A superioridade do HDBSCAN em dados reais fundamenta-se em duas capacidades críticas: o tratamento de *outliers* e a identificação de densidades variáveis.

Primeiramente, o tratamento robusto de **outliers** mostrou-se essencial. Enquanto algoritmos de partição forçada, como o *K-Means* e o HCA, tentam atribuir compulsoriamente cada ponto a um grupo, resultando em grupos contaminados por elementos únicos, o HDBSCAN adota uma abordagem realista, reconhecendo que nem todos os dados pertencem a um grupo denso. Esta filtragem permitiu taxas de ruído superiores a 60% em conjuntos de dados como *Stack Overflow* e *Spotify*, onde a identificação correta de perguntas técnicas únicas ou músicas idiossincráticas resultou em agrupamentos menores, porém com pureza semântica significativamente superior

Em segundo lugar, a capacidade de lidar com **densidades variáveis**. Dados textuais do mundo real apresentam concentrações díspares de informação baseadas na popularidade: itens populares (como filmes famosos no *MovieLens* ou produtos de sucesso na *Amazon*) geram nuvens de pontos densas devido à abundância de *tags* e avaliações, enquanto itens de nicho formam estruturas esparsas. O HDBSCAN adaptou-se a essa topologia heterogênea, gerando grupos coerentes independente da densidade local, ao passo que o *K-Means* tendeu a fragmentar grupos densos ou fundir grupos esparsos devido à sua busca por variância uniforme (grupos esféricos).

5.11.2 Mecanismos de Embeddings: Reforço Léxico contra Contexto Semântico

A escolha do *embedding* ideal demonstrou depender diretamente da natureza do vocabulário e da estrutura da informação em cada domínio.

Domínios Técnicos e Estruturados (Word2Vec): A combinação HDBSCAN + *Word2Vec* foi predominante em domínios técnicos, comerciais e jornalísticos. O sucesso do *Word2Vec* nestes cenários fundamenta-se no mecanismo de **reforço léxico**. Em conjuntos de dados como *Stack Overflow* e *Steam Games*, a terminologia técnica (ex: nomes de funções, *tags* de gênero) repete-se entre o título, o corpo do texto e os metadados O *Word2Vec*, que opera baseado em co-ocorrência e proximidade sintática, beneficia-se dessa redundância: a repetição consistente de termos específicos em diferentes campos fortalece a representação vetorial e a proximidade entre documentos similares.

Domínios Heterogêneos e Criativos (SBERT):

A combinação HDBSCAN + SBERT superou as demais em domínios de alta complexidade semântica, especificamente no *Amazon Reviews*. A vantagem do SBERT reside na sua capacidade de interpretar o **contexto semântico** além da simples co-ocorrência. Em avaliações de produtos, onde o vocabulário varia do técnico ao coloquial e a mesma palavra pode ter conotações diferentes dependendo da frase, o modelo pré-treinado consegue capturar nuances que modelos puramente lexicais perdem. No entanto, em textos muito curtos ou com vocabulário restrito (como no *Spotify*), essa complexidade do modelo transformador pode introduzir ruído, inferindo relações semânticas onde a simples correspondência lexical seria suficiente.

5.11.3 Síntese das Aplicações Práticas

Com base nos padrões observados, há indícios de que as seguintes diretrizes se aplicam para agrupamento de *embeddings*:

- Para dados com vocabulário técnico padronizado e redundância de termos (**reforço léxico**), a aplicação de modelos distributivos como o *Word2Vec* tende a ser mais eficaz.

- Para dados com linguagem natural variada, heterogênea e dependente de interpretação contextual, modelos baseados em *Transformers* como o SBERT são mais indicados.
- O uso de algoritmos baseados em densidade (HDBSCAN) é preferível em cenários reais de mineração de texto para garantir a pureza dos grupos através da exclusão de ruído e para lidar com a variação natural de densidade entre categorias populares e de nicho.

5.11.4 Divergências e Limitações

Observou-se uma divergência entre métricas internas (coesão) e externas (ARI) em experimentos como *20Newsgroups* e *BBC News*, indicando que grupos compactos matematicamente nem sempre correspondem às categorias humanas predefinidas.

Outra limitação deste trabalho, é que o tipo de dado dos *corpora* utilizados neste experimento difere daquele provável treinamento do SBERT, embora os detalhes específicos do grupo de dados de treino do SBERT não sejam públicos, assume-se que sejam baseados em estruturas sintáticas tradicionais e formais. Em contrapartida, muitos conjuntos de dados deste trabalho apresentam sentenças curtas e fragmentadas, carecendo frequentemente de estruturas semânticas completas, o que pode ter penalizado o desempenho dos modelos transformadores em comparação aos modelos baseados em co-ocorrência lexical.

Por fim, uma última limitação observada durante os experimentos refere-se à disparidade no custo computacional entre as abordagens. Enquanto a geração de *embeddings* com *Word2Vec* e a execução de algoritmos como *K-Means*, HCA e HDBSCAN foram concluído em poucas horas para a totalidade dos conjuntos de dados, enquanto as abordagens envolvendo modelos baseados em SBERT e o treinamento de redes neurais como o SOM apresentaram um custo de processamento significativamente superior, chegando a levar dias para processar a totalidade dos conjuntos de dados. Essa diferença de magnitude temporal impõe uma barreira prática para a utilização de SBERT e SOM em cenários que exigem retreino frequente ou baixa latência, mesmo que estes apresentem vantagens em domínios específicos.

5.12 TRABALHOS FUTUROS

Os resultados obtidos nesta investigação abrem diversas oportunidades para pesquisas subsequentes que podem ampliar a compreensão sobre agrupamento de texto não supervisionado. A expansão do espectro de *embeddings* através da investigação de modelos como *FastText*, *GloVe*, RoBERTa (Robustly Optimized BERT Pretraining Approach (Abordagem de Pré-treinamento BERT Robustamente Otimizada)) e BERTimbau (BERT model for Brazilian Portuguese (Modelo BERT para Português Brasileiro)), além de modelos mais simples como TF-IDF (Term Frequency-Inverse Document Frequency (Frequência de Termo - Frequência Inversa de Documento)), visto a performance do *Word2Vec*, permitiria validar a generalização dos padrões identificados para outros idiomas e arquiteturas. Adicionalmente, a exploração de algoritmos de agrupamento avançados, incluindo técnicas de *deep clustering*, métodos espectrais e abordagens *ensemble*, poderia revelar combinações ainda mais eficazes.

Além disto, testar outras técnicas de pré-processamento, como remoção de *Stop Words* especificas, como "produto"para *Amazon*, *Error* para o *Stack Overflow*, e assim por diante, palavras contaminantes que se repetem em quase todas as entradas e não agregam aos *embeddings*, pois assim podemos investigar se processamentos específicos para cada conjunto de dados melhorariam o desempenho dos agrupamentos.

REFERENCES

ARTHUR, David; VASSILVITSKII, Sergei. k-means++: the advantages of careful seeding. *In*: PROCEEDINGS of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. New Orleans, Louisiana: Society for Industrial e Applied Mathematics, 2007. (SODA '07), p. 1027–1035.

BERBATOVA, Melania. Overview on NLP Techniques for Content-based Recommender Systems for Books. *In*: KOVATCHEV, Venelin; TEMNIKOVA, Irina; ŠANDRIH, Branislava; NIKOLOVA, Ivelina (Ed.). **Proceedings of the Student Research Workshop Associated with RANLP 2019**. Varna, Bulgaria: INCOMA Ltd., set. 2019. P. 55–61.

BOJANOWSKI, Piotr; GRAVE, Edouard; JOULIN, Armand; MIKOLOV, Tomas. Enriching Word Vectors with Subword Information. **Transactions of the Association for Computational Linguistics**, v. 5, p. 135–146, 2017.

CAMPELLO, R. J. G. B.; MOULAVI, D.; ZIMEK, A.; SANDER, J. A framework for semi-supervised and unsupervised optimal extraction of clusters from hierarchies. **Data Mining and Knowledge Discovery**, v. 27, n. 3, p. 344–371, nov. 2013. ISSN 1573-756X.

CAMPELLO, Ricardo J. G. B.; MOULAVI, Davoud; SANDER, Joerg. Density-Based Clustering Based on Hierarchical Density Estimates. *In*: PEI, Jian; TSENG, Vincent S.; CAO, Longbing; MOTODA, Hiroshi; XU, Guandong (Ed.). **Advances in Knowledge Discovery and Data Mining**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. P. 160–172.

DAVIES, David L.; BOULDIN, Donald W. A Cluster Separation Measure. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, PAMI-1, n. 2, p. 224–227, abr. 1979. ISSN 1939-3539.

DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *In*: BURSTEIN, Jill; DORAN, Christy; SOLORIO, Thamar (Ed.). **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)**. Minneapolis, Minnesota: Association for Computational Linguistics, jun. 2019. P. 4171–4186.

ESTER, Martin; KRIEGEL, Hans-Peter; SANDER, Jörg; XU, Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. *In*: PROCEEDINGS of the Second International Conference on Knowledge Discovery and Data Mining. Portland, Oregon: AAAI Press, 1996. (KDD'96), p. 226–231.

HUBERT, Lawrence; ARABIE, Phipps. Comparing partitions. **Journal of Classification**, v. 2, n. 1, p. 193–218, dez. 1985. ISSN 1432-1343.

KOCOŃ, Jan; GAWOR, Michał. Evaluating KGR10 Polish word embeddings in the recognition of temporal expressions using BiLSTM-CRF. **Schedae Informaticae**, v. 27, 2018.

KOHONEN, T. The self-organizing map. **Proceedings of the IEEE**, v. 78, n. 9, p. 1464–1480, set. 1990. ISSN 1558-2256.

KULA, Maciej. **Mixture-of-tastes Models for Representing Users with Diverse Interests**. [*S.l.*: *s.n.*], 2018. arXiv: 1711.08379 [cs.IR]. Disponível em: https://arxiv.org/abs/1711.08379.

LLOYD, Stuart P. Least squares quantization in PCM. **IEEE Trans. Inf. Theory**, v. 28, p. 129–136, 1982.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. *In*.

MCINNES, Leland; HEALY, John; ASTELS, Steve. hdbscan: Hierarchical density based clustering. **Journal of Open Source Software**, The Open Journal, v. 2, n. 11, p. 205, 2017.

MCQUITTY, L. L. Hierarchical linkage analysis for the isolation of types. **Educational and Psychological Measurement**, v. 20, p. 55–67, 1960.

MEHTA, Vivek; BAWA, Seema; SINGH, Jasmeet. WEClustering: word embeddings based text clustering technique for large datasets. **Complex & Intelligent Systems**, v. 7, n. 6, p. 3211–3224, dez. 2021. ISSN 2198-6053.

MIKOLOV, Tomas; CHEN, Kai; CORRADO, Greg; DEAN, Jeffrey. **Efficient Estimation of Word Representations in Vector Space**. [*S.l.*: *s.n.*], 2013. arXiv: 1301.3781 [cs.CL]. Disponível em: https://arxiv.org/abs/1301.3781.

MOULAVI, Davoud; JASKOWIAK, Pablo A.; CAMPELLO, Ricardo J. G. B.; ZIMEK, Arthur; SANDER, Jörg. Density-Based Clustering Validation. *In*: PROCEEDINGS of the 2014 SIAM International Conference on Data Mining (SDM). [*S.l.*: *s.n.*]. P. 839–847. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611973440.96.

MUENNIGHOFF, Niklas; TAZI, Nouamane; MAGNE, Loïc; REIMERS, Nils. **MTEB: Massive Text Embedding Benchmark**. [*S.l.*: *s.n.*], 2023. arXiv: 2210.07316 [cs.CL]. Disponível em: https://arxiv.org/abs/2210.07316.

PAUDEL, Bibek; LUCK, Sandro; BERNSTEIN, Abraham. **Loss Aversion in Recommender Systems: Utilizing Negative User Preference to Improve Recommendation Quality**. [*S.l.*: *s.n.*], 2018. arXiv: 1812.11422 [cs.IR]. Disponível em: https://arxiv.org/abs/1812.11422.

PENNINGTON, Jeffrey; SOCHER, Richard; MANNING, Christopher. GloVe: Global Vectors for Word Representation. *In*: MOSCHITTI, Alessandro; PANG, Bo; DAELEMANS, Walter (Ed.). **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)**. Doha, Qatar: Association for Computational Linguistics, out. 2014. P. 1532–1543.

RAHUTOMO, Reza; PERBANGSA, Anzaludin Samsinga; SOEPARNO, Haryono; PARDAMEAN, Bens. Embedding Model Design for Producing Book Recommendation. *In*: 2019 International Conference on Information Management and Technology (ICIMTech). [*S.l.*: *s.n.*], 2019. v. 1, p. 537–541.

RAND, William M. Objective Criteria for the Evaluation of Clustering Methods. **Journal of the American Statistical Association**, ASA Website, v. 66, n. 336, p. 846–850, 1971. eprint: https://www.tandfonline.com/doi/pdf/10.1080/01621459.1971.10482356.

REIMERS, Nils; GUREVYCH, Iryna. **Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks**. [*S.l.*: *s.n.*], 2019. arXiv: 1908.10084 [cs.CL]. Disponível em: https://arxiv.org/abs/1908.10084.

REIMERS, Nils; GUREVYCH, Iryna. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. **CoRR**, abs/1908.10084, 2019. arXiv: 1908.10084.

REIMERS, Nils; SCHILLER, Benjamin; BECK, Tilman; DAXENBERGER, Johannes; STAB, Christian; GUREVYCH, Iryna. Classification and Clustering of Arguments with Contextualized Word Embeddings. *In*: KORHONEN, Anna; TRAUM, David; MÀRQUEZ, Lluís (Ed.). **Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics**. Florence, Italy: Association for Computational Linguistics, jul. 2019. P. 567–578.

ROUSSEEUW, Peter J. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. **Journal of Computational and Applied Mathematics**, v. 20, p. 53–65, 1987. ISSN 0377-0427.

SAHA, Rohan. **Influence of various text embeddings on clustering performance in NLP**. [*S.l.*: *s.n.*], 2023. arXiv: 2305.03144 [cs.LG]. Disponível em: https://arxiv.org/abs/2305.03144.

SHAFFER, Kyle. Language Clustering for Multilingual Named Entity Recognition. *In*: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. FINDINGS of the Association for Computational Linguistics: EMNLP 2021. [*S.l.*: *s.n.*], 2021. P. 40–45.

SLOKOM, Manel. Comparing recommender systems using synthetic data. *In*: PROCEEDINGS of the 12th ACM Conference on Recommender Systems. Vancouver, British Columbia, Canada: Association for Computing Machinery, 2018. (RecSys '18), p. 548–552.

SNEATH, P. H. A. The application of computers to taxonomy. **Journal of General Microbiology**, v. 17, p. 201–226, 1957.

SOKAL, R. R.; MICHENER, C. D. A statistical method for evaluating systematic relationships. **University of Kansas Scientific Bulletin**, v. 38, p. 1409–1438, 1958.

SOKAL, R. R.; SNEATH, P. H. A. **Principles of Numerical Taxonomy**. San Francisco: Freeman, 1963.

SONG, Kaitao; TAN, Xu; QIN, Tao; LU, Jianfeng; LIU, Tie-Yan. MPNet: Masked and Permuted Pre-training for Language Understanding. **CoRR**, abs/2004.09297, 2020. arXiv: 2004.09297.

TAN, Xu; CHEN, Jiale; HE, Di; XIA, Yingce; QIN, Tao; LIU, Tie-Yan. Multilingual Neural Machine Translation with Language Clustering. *In*: INUI, Kentaro; JIANG, Jing; NG, Vincent; WAN, Xiaojun (Ed.). **Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)**. Hong Kong, China: Association for Computational Linguistics, nov. 2019. P. 963–973.

VITHANAGE, Dinithi; YU, Ping; WANG, Lei; DENG, Chao. Contextual Word Embedding for Biomedical Knowledge Extraction: a Rapid Review and Case Study. **Journal of Healthcare Informatics Research**, v. 8, p. 1–22, jan. 2024.

WALKOWIAK, Tomasz; GNIEWKOWSKI, Mateusz. Evaluation of vector embedding models in clustering of text documents. *In*: MITKOV, Ruslan; ANGELOVA, Galia (Ed.). **Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)**. Varna, Bulgaria: INCOMA Ltd., set. 2019. P. 1304–1311.

YANG, Pengcheng; CHE, Chunjiang; ZHAO, Lu; ZHANG, Ziyu. Highway Service Areas Recommendation System with Collaborative Filtering using Channel Attention Module based Graph Convolutional Neural Network. *In*: 2024 Second International Conference on Data Science and Information System (ICDSIS). [*S.l.*: *s.n.*], 2024. P. 1–4.

ZHU, An-Qing; LI, Hong-Yuan; WU, Rui-Hui. Network Data Resource Recommendation Based on Spark and Fast Spectral Clustering Algorithms. *In*: 1. JOURNAL of network intelligence. [*S.l.*: *s.n.*], 2024. v. 9, p. 273–288.