Universidade Federal de São Carlos– UFSCar
Centro de Ciências Exatas e de Tecnologia– CCET
Departamento de Computação– DC
Programa de Pós-Graduação em Ciência da Computação– PPGCC

Paulo César Polastri

Organização de Termos e Documentos Utilizando Co-Clustering e Agrupamento de Word Embeddings

São Carlos
2021

Paulo César Polastri

Organização de Termos e Documentos Utilizando Co-Clustering e Agrupamento de Word Embeddings

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação do Centro de Ciências Exatas e de Tecnologia da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação.

Área de concentração: Metodologias e Técnicas de Computação

Orientador: Prof(a). Dra. Heloisa de Arruda Camargo

São Carlos
2021

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Defesa de Tese de Doutorado do candidato Paulo César Polastri, realizada em 15/12/2021.

Comissão Julgadora:

Profa. Dra. Heloisa de Arruda Camargo (UFSCar)

Profa. Dra. Maria do Carmo Nicoletti (UFSCar)

Profa. Dra. Helena de Medeiros Caseli (UFSCar)

Prof. Dr. Rafael Giraldeli Rossi (UFMS)

Prof. Dr. Ricardo Marcondes Marcacini (USP)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.
O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Este trabalho é dedicado aos meus queridos pais.

Agradecimentos

A Deus.

Aos meus pais, Aparecida e Lourenço, pelo incentivo, confiança, respeito, compreensão, paciência e orgulho que demonstraram ao longo deste caminho. Sem isso, nada seria possível;

À minha orientadora, professora Dra. Heloisa de Arruda Camargo, pela confiança, esforço, paciência e dedicação digiridos a mim e ao trabalho e pela aplicação do seu conhecimento e experiência na pesquisa, tornando este projeto possível;

À minha namorada, Brenda, pela compreensão nas horas mais difíceis;

Ao meu irmão, Wanderson, pelos conselhos;

Ao professor Dr. Ricardo Marcondes Marcacini pela ajuda, dedicação e disponibilidade;

Aos professores que compuseram a banca de defesa;

À todos meus amigos e amigos que fiz no DC;

À todas as pessoas importantes que estão ou passaram pela minha vida que, de alguma forma, seja com uma palavra, crítica, elogio ou um simples gesto, contribuíram;

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES);

"O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001".

I'm changing,
rearranging my destiny.
Learning to fly
touching sky on my way.
Learning to fly
high above the sorrows
to a new day, I fly
(Stratovarius)

Resumo

Existe uma grande quantidade de documentos de texto disponível na literatura que aumenta à medida que fluxo de informações aumenta. Analisar e organizar tais documentos considerando características como assunto torna-se uma tarefa cada vez mais dispendiosa, porém indispensável, considerando tarefas como mineração de textos e recuperação de informações e, sendo assim, meios para melhorar o desempenho de tais tarefas são amplamente investigados. A maioria das tarefas voltadas para a organização de documentos disponíveis atualmente, como tarefas de clustering, se concentram em apenas uma dimensão, ou seja, agrupar apenas os documentos considerando a ocorrência de termos. Porém, um aspecto importante do agrupamento de documentos é encontrar tópicos que identificam os clusters de documentos segundo seu conteúdo. Estratégias de clustering bidimensional, que agrupam simultaneamente documentos e termos, podem ser úteis nesse sentido. Entretanto, a representação utilizada é, em geral, na forma de matrizes de alta dimensionalidade e esparsidade, que não inclui nenhuma informação semântica. Neste trabalho é apresentada uma abordagem para organizar documentos usando co-clustering e a representação dos termos na forma de embeddings. Os termos da coleção de documentos são agrupados previamente, permitindo a redução da esparsidade e dimensionalidade da matriz. Além da nova representação, a estratégia proposta inclui contribuições para avaliar o resultado do co-clustering que exploram a associação entre clusters de documentos e termos. Em tarefas de co-clustering, os resultados mostraram que a representação supera a representação TF-IDF (Term Frequency Inverse Document Frequency) tradicional em muitos casos.

Palavras-chave: Organização de documentos e termos, co-clustering, avaliação de co-clustering, word embeddings.

Abstract

There is a large amount of text documents available on the web which increases as more devices and users connect to the network. Analyzing and organizing such documents considering characteristics such as subject and keywords becomes an increasingly expensive task, but indispensable, considering tasks such as text mining and information retrieval and, therefore, ways to improve the performance of such tasks are widely investigated. Most tasks aimed at organizing documents available today, such as clustering tasks, focus on only one dimension, that is, clustering only documents considering the occurrence of terms. However, an important aspect of clustering documents is finding topics that identify groups of documents by their content. Two-dimensional clustering strategies, which simultaneously group documents and terms, can be useful in this regard. However, the representation used is, in general, in the form of matrices of high dimensionality and sparsity, which does not include any semantic information. This work presents a new approach to organize documents using co-clustering and the representation of terms in the form of embeddings. The terms of the document collection are clustered in advance, allowing for the reduction of the sparsity and dimensionality of the matrix. In addition to the new representation, the proposed strategy includes contributions to assess the outcome of co-clustering that explore the association between groups of documents and terms. In co-clustering tasks, the results showed that the representation surpasses the traditional TF-IDF representation in specific cases.

Keywords: Document and term organization, co-clustering, co-clustering evaluation, word embeddings.

Lista de ilustrações

Lista de tabelas

Lista de siglas

ACC Acurácia

AM Aprendizado de Máquina

ARI Adjusted Rand Index

BILP Binary Integer Linear Programming

BoW Bag-of-Words

C3 Classic3

C4 Classic4

CBoW Continuous Bag-of-Words

DB Davies Bouldin

GloVe Global Vectors for Word Representation

IA Inteligência Artificial

IDF Inverse Document Frequence

INFO Information Theoretic Co-clustering

Jacc Similaridade de Jaccard

LBM Latent Block Model

MLP Multi-layer Perceptron

NG5 Newsgroup5

NLTK Natural Language Toolkit

NMI Normalized Mutual Information

PLN Processamento de Linguagem Natural

R8 Reuters8

Spec Spectral Co-clustering

TF Term Frequence

TF-IDF Term Frequence Inverse Document Frequence

VSM Vector Space Model

WE Word Embeddings

Sumário

Capítulo 1

Introdução

Este capítulo tem o propósito de contextualizar o leitor sobre os motivos que levaram ao desenvolvimento deste trabalho e também seus objetivos. Esse capítulo está organizado da seguinte forma: Na seção 1.1 é contextualizado o problema. Na seção 1.2 são apresentados os motivos que levaram a esta tese de doutorado. Na seção 1.3, são apresentados os objetivos almejados da pesquisa. Na seção 1.4 são apresentadas as contribuições do trabalho. Por fim, na seção 1.5 é apresentado como o decorrer do texto está organizado.

1.1 Contextualização

Nos últimos anos o número de documentos de texto disponíveis vem crescendo consideravelmente, o que torna a análise de tais dados uma tarefa cada vez mais necessária e custosa (VINAYKUMAR; RAJAVELU; BLESSING, 2020). O desenvolvimento de técnicas que permitam a análise e a organização de grandes coleções de dados textuais é de vital importância no avanço da tecnologia de descoberta de conhecimento (GUANGCE; LEI, 2020; AGGARWAL, 2018). Organizar objetos é uma tarefa que consiste na determinação de um conjunto finito de categorias a partir de um conjunto de objetos (no contexto do trabalho, uma coleção de documentos de texto), considerando a similaridade existente entre estes objetos (DHAR et al., 2021). Clustering é uma técnica que permite, quando aliada à documentos de texto, organizá-los em clusters considerando semelhanças.

Grupos de objetos geralmente ocorrem em diferentes tipos de conjunto de dados. Em conjuntos de dados é possível notar semelhanças e diferenças de acordo com as características de cada objeto, o que permite considerar que os objetos pertençam a um mesmo cluster (ou seja, apresentam características semelhantes) ou pertençam a clusters diferentes (ou seja, apresentam características diferentes) (GARCIA-DIAS et al., 2020).

Quando os dados são documentos de texto, agrupar documentos é considerada uma tarefa cujo objetivo é organizar automaticamente documentos em *clusters* significativos com assunto e termos coerentes. É uma estratégia amplamente utilizada em aplicações reais (AGGARWAL, 2018).

1.2 Motivação

Clustering de documentos é uma técnica importante de Aprendizado de Máquina (AM) amplamente utilizada em aplicações reais, como na organização e análise de documentos de texto. O objetivo é organizar automaticamente uma coleção de documentos em *clusters* significativos de tópicos coerentes com base na similaridade no conteúdo (HARRINGTON, 2012).

Witten et al. (2005) cita que a aplicação de técnicas de *clustering* ajuda a encontrar padrões nos dados. Na mineração de dados, Roiger (2017) cita que a utilização de algoritmos de *clustering* auxilia na exploração de dados, principalmente em larga escala, além de possibilitar encontrar *clusters* em subespaços de alta dimensão, escalabilidade e gerar compreensão dos resultados. Na recuperação de informação (IR), Büttcher, Clarke e Cormack (2016) dizem que a abordagem de *clustering* pode ser eficaz quanto à relevância dos resultados. Em aplicações médicas ou biológicas, Paul e Hoque (2010) citam que a aplicação de *clustering* em dados de pacientes pode auxiliar na predição do diagnóstico. Em motores de busca, Halavais (2017) menciona que a aplicação de *clustering* gera um aumento significativo na precisão dos resultados.

Em aplicações utilizando dados reais, como *corpora* ou coleções de documentos de texto, muitas vezes existe a necessidade de agrupar não apenas documentos mas também os termos mais importantes/relevantes que ocorrem em *clusters* de documentos. (DHILLON; MALLELA; MODHA, 2003). Para possibilitar tarefas de *clustering* de documentos e termos, uma das características mais significativas é a representação. Diferentemente dos seres humanos, máquinas e algoritmos não tem capacidade para interpretar informações contidas em termos considerando a forma natural como encontrados, tornando uma forma de representação numérica necessária para representar documentos de texto.

Normalmente, representações numéricas de documentos e termos apresentam características como esparsidade e alta dimensionalidade. Kummamuru, Dhawale e Krishnapuram (2003) dizem que algoritmos de *clustering* costumam apresentar alguns problemas quando os dados são esparsos. Porém, quando os dados são escaláveis e/ou apresentam alta dimensionalidade, estratégias de *clustering* unidimensional já não apresentam resultados tão significativos.

Uma forma de representação numérica de documentos e termos utilizada para contornar o problema da alta dimensionalidade e escalabilidade, referida como word embedding, é apresentada em Mikolov et al. (2013a). Word embedding é uma abordagem de repre-

sentação numérica de textos onde um termo é representado como um vetor de tamanho fixo e valores reais. Esta forma de representação permite que termos que tenham um contexto semelhante possuam representações vetoriais semelhantes. Outra característica é que word embeddings incorporam sintática e semântica contidas no texto (MIKOLOV et al., 2013a).

O *co-clustering* é uma técnica utilizada em diversos domínios para identificar padrões e estruturas em conjuntos de dados multidimensionais. Ao contrário de técnicas de *clustering* tradicionais, o *co-clustering* considera simultaneamente tanto os *clusters* de objetos quanto os *clusters* de atributos, o que o torna valioso em situações onde há interdependências entre ambas as dimensões. Estratégias de *co-clustering* buscam agrupar objetos e características simultaneamente, ou, neste caso, documentos de texto e tópicos/termos relevantes. Mais precisamente, *co-clustering* é uma técnica de aprendizado de máquina que agrupa simultaneamente dois tipos de entidades quando existe uma conexão entre elas (LACLAU; NADIF, 2016; BITON; KALECH; ROKACH, 2018).

Kummamuru, Dhawale e Krishnapuram (2003), Laclau e Nadif (2016) e Biton, Kalech e Rokach (2018), afirmam que os problemas de dimensionalidade e escalabilidade podem ser tratados com a utilização de *co-clustering* e que tal estratégia apresenta menor esforço computacional e representação simplificada de dados quando comparada a estratégias de *clustering* tradicionais.

Estratégias de *co-clustering* tem sido utilizadas em vários domínios, especialmente para tratar da organização de documentos de texto, a partir do pressuposto que termos que co-ocorrem em documentos tendem a estar associadas a conceitos similares. Portanto, o *clustering* simultâneo de termos similares e de documentos similares são igualmente importantes.

Apesar das vantagens em descobrir *clusters* em duas dimensões, a avaliação dos resultados obtidos por *co-clustering* apresenta desafios, pois, para isso, as métricas de avaliação utilizadas são adaptadas de medidas tradicionais, ignorando a natureza bidimensional do *co-clustering*. Em muitos casos, as medidas de avaliação convencionais, como a precisão, são aplicadas separadamente para avaliar os *clusters* apenas nos objetos ou apenas nos atributos. Isso negligencia a inter-relação entre ambas as dimensões, comprometendo a análise global do desempenho do *co-clustering*. A aplicação isolada dessas métricas pode levar a interpretações equivocadas e subestimar a eficácia do algoritmo.

1.3 Objetivos

Como mencionado ao longo do capítulo, organizar documentos de texto pode beneficiar várias áreas do conhecimento. O *clustering* de documentos produz conhecimento útil. Tal conhecimento pode ser utilizado de diversas formas e aplicado em diversas pesquisas.

A partir das pesquisas realizadas supõem-se que a combinação da representação de

documentos utilizando word embeddings e algoritmos de *co-clustering* seja benéfica para a organização de documentos em *clusters* de conteúdo similar e identificados por descritores que podem ser utilizados de forma eficaz nas tarefas de classificação e recuperação de documentos. Uma abordagem mais adequada para avaliar o *co-clustering* deve utilizar de medidas específicas que considerem a simultaneidade das dimensões, procurando capturar a qualidade do *clustering* em ambas as dimensões, proporcionando assim uma avaliação mais precisa e abrangente do desempenho do *co-clustering*.

Sendo assim, esta tese tem por objetivo:

"Desenvolver um método eficiente para organização de documentos de texto usando clusters de termos e algoritmos de co-clustering na representação textual para criar clusters de documentos de conteúdo semelhante e com associação representativa com clusters de termos além de desenvolver medidas de avaliação eficientes para avaliar a qualidade entre clusters de documentos de conteúdo semelhante com associação representativa com clusters de termos específicos."

E como objetivos específicos:

- Propor e investigar o uso de *clustering* de termos usando embeddings como forma de diminuir a dimensionalidade e esparsidade das matrizes utilizadas para representação estruturada de documentos;
- Propor uma forma representativa para termos e documentos considerando vetores word embeddings;
- Desenvolver e avaliar um método eficiente para organização de documentos de texto usando *clusters* de termos e algoritmos de *co-clustering*.
- Encontrar formas efetivas de avaliar o resultado de algoritmos de *co-clustering*, com ênfase na relação entre *clusters* de documentos e termos representativos desse *cluster*, com foco na avaliação de *clusters* de termos e avaliação do ajuste entre *clusters* de termos e *clusters* de documentos.

Durante as pesquisas realizadas neste projeto questões foram levantadas sobre o tema alvo. Este trabalho investigou questões como:

- Representar documentos e termos usando word embeddings melhora o desempenho de algoritmos de *co-clustering*?
- Fazer o *clustering* termos previamente à aplicação de *co-clustering* pode melhorar o desempenho de algoritmos de *co-clustering*?
- Considerando medidas de avaliação de algoritmos de *clustering* e *co-clustering*, a representação de documentos de textos gerada por *clusters* de termos resulta em formas mais representativas de análise do resultado de *co-clustering*?

Em busca dos objetivos propostos, foram realizadas várias pesquisas e estudos direcionados a trabalhos (LACLAU; NADIF, 2016; GOVAERT; NADIF, 2003; DHILLON,

2001; ROLE; MORBIEU; NADIF, 2019), técnicas e algoritmos que utilizam as estratégias citadas ao longo deste capítulo (*co-clustering*, representação de documentos e termos e word embeddings), principalmente em trabalhos cujo foco é a organização de documentos ou o *clustering* de documentos.

As pesquisas realizadas levaram ao projeto e desenvolvimento de uma abordagem para organização de documentos que utiliza algoritmos de *co-clustering* após fazer o *clustering* de termos usando seus embeddings e gerar uma matriz da forma documentos × grupos, com um cálculo da medida TF-IDF adaptada para *clusters* de termos. Essa representação permite reduzir a dimensionalidade e esparsidade de representações tradicionais, como TF-IDF, vetores de contagem e *bag-of-words*, por exemplo, e favorece análises dos resultados do *co-clustering*.

A escolha por vetores de word embedding parte do pressuposto que documentos de texto com conteúdo semelhante podem ser escritos de formas diferentes (HARRIS, 1954), utilizando termos diferentes. Assim, entende-se que o uso de relações semânticas no processo de *clustering* dos objetos é de grande importância para produzir *clusters* mais precisos.

Combinado com a proposta de representação baseada em *clusters*, a abordagem criada neste trabalho inclui propostas de variações de métricas de validação já conhecidas. Devido ao resultado dos algoritmos de *co-clustering* encontrarem *co-clusters*, que são *clusters* de documentos relacionados a *clusters* de termos, as avaliações demandam métricas e análises adicionais àquelas utilizadas em estratégias de *clustering* unidimensionais. As avaliações foram feitas em três partes. Na primeira parte, os *clusters* de documentos são avaliados separadamente, usando métricas já conhecidas, como Normalized Mutual Information, Adjusted Rand Index e Acurácia. Na segunda parte, são avaliados os *clusters* de documentos e de termos separadamente, usando uma métrica proposta neste trabalho e a métrica Davies-Bouldin, já conhecida. Na terceira parte, é avaliado o ajuste entre *clusters* de documentos e *clusters* de termos, que é a forma de avaliação que só pode ser feita em estratégias de *co-clustering* e que pode embasar a criação de métodos para extração de tópicos representativos de *clusters* e de decisões acerca do pré-processamento de coleções. Nesse tipo de avaliação foram usadas duas métricas propostas neste trabalho como adaptação de métricas encontradas na literatura. Um segundo conjunto de experimentos foi realizado com o algoritmo Information Theoretic co-clustering, definindo números de *clusters* de documentos diferente do número de *clusters* de termos.

Agrupar documentos de texto não é uma tarefa trivial. Para que sejam criados *clusters* de qualidade ao final do processo é necessário muito esforço, tanto humano quanto computacional. Quando bons resultados são obtidos, estes podem ser aplicados em várias áreas de conhecimento, como, por exemplo, recuperação de informações (BELLOT; EL-BÈZE, 1999), classificação de texto (KYRIAKOPOULOU; KALAMBOUKIS, 2006), detecção e rastreamento de tópicos (WARTENA; BRUSSEE, 2008), tradução automática

(TAN et al., 2019), sumarização (ALGULIYEV et al., 2019), entre outras.

1.4 Contribuições

Pode-se afirmar que as principais contribuições deste trabalho são:

– Proposta de geração da matriz da forma documentos × grupos com cálculo estendido da medida TF-IDF, com base no *clustering* prévio de termos do vocabulário da coleção usando word embeddings;

– Proposta de adaptações de métricas encontradas na literatura para avaliar os *clusters* de termos separadamente e o ajuste entre *clusters* de documentos e *clusters* de termos encontrados pelo *co-clustering*;

– Conjunto extensivo de experimentos em que são explorados três tipos de avaliações: qualidade de *clusters* de documentos, qualidade de *clusters* de termos e ajuste entre *clusters* de documentos e *clusters* de termos.

1.5 Organização do Texto

A sequência deste documento está organizada como descrito a seguir.

No Capítulo 2 (Fundamentação Teórica) são apresentados conceitos de Aprendizado de Máquina (AM), *clustering* e medidas de avaliação, para a compreensão da proposta.

No Capítulo 3 (Representações Numéricas de Termos e Documentos) são apresentados conceitos relacionados a representações numéricas de termos e documentos.

No Capítulo 4 (Biclustering e Co-clustering) são detalhados *Biclustering* e *Co-clustering*, suas características, diferenças e aplicações.

No Capítulo 5 (Abordagem para Co-clustering de Documentos com Avaliação de Co-clusters) é apresentada uma abordagem para *co-clustering* de documentos com *clustering* prévio de termos, as ferramentas, recursos, técnicas e estratégias utilizadas para a implementação do método e as métricas de avaliação propostas, como a avaliação do ajuste entre *clusters* de documentos e de termos, métrica original proposta neste trabalho.

No Capítulo 6 (Experimentos e Resultados) é apresentada uma descrição dos experimentos e resultados.

Por fim, no Capítulo 7 são apresentadas as conclusões.

Capítulo 2

Fundamentação Teórica

Este capítulo tem o propósito de apresentar ao leitor conceitos encontrados na literatura atual sobre aprendizado de máquina e clustering. Esse capítulo está organizado da seguinte forma: Na seção 2.1 é apresentado o conceito de aprendizado de máquina, focando principalmente nos tipos de aprendizado, aprendizado supervisionado, aprendizado não-supervisionado e aprendizado semissupervisionado. Na seção 2.2 é apresentado o conceito de clustering, mencionando o funcionamento do algoritmo k-means e medidas de similaridade e dissimilaridade.

2.1 Aprendizado de Máquina

Aprendizado de Máquina (AM) é uma área da Inteligência Artificial (IA) cujo objetivo é dar a um sistema a capacidade para adquirir conhecimento de forma automática através de estratégias de aprendizado computacional (MITCHELL, 1997). O processo de aprendizado inclui análise de dados, como, por exemplo, experiência e/ou instrução, com o objetivo de encontrar padrões nos dados e possibilitar tomar melhores decisões no futuro.

A estratégia é aplicada para a construção de sistemas que melhoram seu desempenho em uma tarefa quando são fornecidos exemplos de desempenho ideal ou melhoram seu desempenho com a experiência repetida na tarefa, com o objetivo de permitir que os sistemas aprendam automaticamente, sem intervenção humana ou necessidade de ajustar as ações. Em outras palavras, o AM busca dar autonomia para sistemas com relação a tratar situações ainda não vistas (RUSSELL; NORVIG, 2002). Algoritmos de aprendizado de máquina são utilizados, por exemplo, em sistemas de reconhecimento de voz (TANDEL;

PRAJAPATI; DABHI, 2020), sistemas de detecção de fraude (VARMEDJA et al., 2019), sistemas de recomendação (BERTENS et al., 2018), entre outros.

O AM indutivo é aquele que utiliza dados para generalizar conhecimento (MITCHELL, 1997). Através do princípio da indução é possível generalizar o conhecimento embutido em um conjunto de exemplos específicos, podendo gerar hipóteses que conservam ou não a verdade. Para isso é importante dispor de uma quantidade suficiente de exemplos de boa relevância, o que é muito importante para que o sistema possa adquirir conhecimento e fazer a indução corretamente.

Normalmente cada exemplo (também usualmente referido como instância ou objeto no contexto de AM) é representado por uma coleção de características (ou atributos): numéricos (como números reais; inteiros; etc.) e categóricos, ou seja, tipos de dados que podem ser divididos em categorias. Exemplos de categorias podem, por exemplo, ser: etnia (caucasiana, indígena, negra, etc.), cor do cabelo (loiro, ruivo, castanho, preto); sexo; faixa etária; entre outros (XU; WUNSCH, 2008). Os objetos estão geralmente disponíveis de duas formas: rotulados e não rotulados. Objetos rotulados são associados a um rótulo que é definido como o valor de um atributo especial chamado de atributo meta que, assim como os demais atributos, pode ser numérico ou categórico. Objetos não rotulados não possuem o atributo meta e, portanto, não são associados a nenhum rótulo.

O AM indutivo é geralmente dividido em três formas de aplicação: aprendizado supervisionado, aprendizado não-supervisionado e aprendizado semissupervisionado.

No aprendizado supervisionado, para todo exemplo de entrada é fornecido o rótulo com uma categoria ou valor numérico. Quando tal categoria for um valor discreto de um conjunto finito de valores, o problema da aprendizado é classificação. Quando for um valor contínuo, como uma expectativa condicional ou valor médio, o problema de aprendizado é regressão (RUSSELL; NORVIG, 2002).

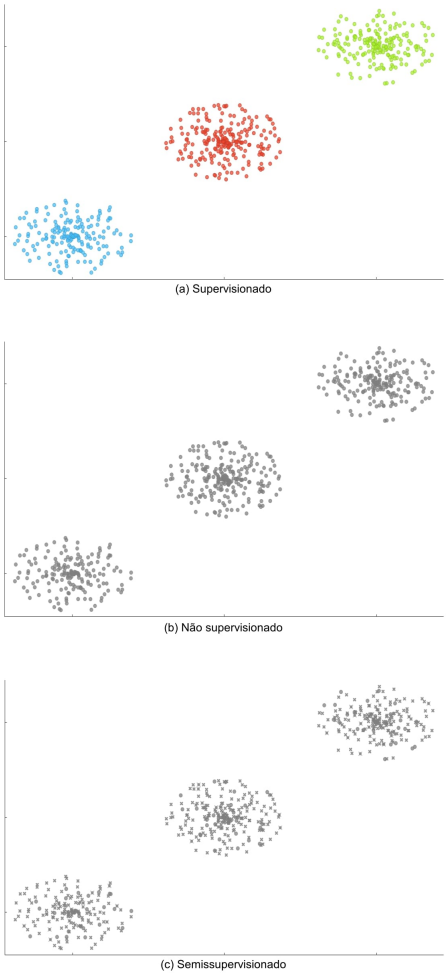
No aprendizado não-supervisionado, o valor do atributo meta não é conhecido, ou seja, não há uma interferência humana para auxiliar o algoritmo de aprendizado ou guiá-lo de forma a processar os exemplos de forma correta. Uma das abordagens mais utilizadas de aprendizado não supervisionado é a análise de *clustering* de dados, cujo objetivo é encontrar *clusters* de dados de modo que aqueles dados que ficam no mesmo *cluster* são mais semelhantes entre si do que dados em *clusters* diferentes (MONARD; BARANAUSKAS, 2003).

Como alternativa ao aprendizado não supervisionado e supervisionado existe o aprendizado semissupervisionado. No aprendizado semissupervisionado, durante o treinamento do modelo, são apresentados tanto objetos rotulados quanto não rotulados ao algoritmo de aprendizado (MONARD; BARANAUSKAS, 2003).

Na Figura 1 estão representadas amostras para algoritmos de aprendizado supervisionado, não supervisionado e semissupervisionado. Na Figura 1(a) (método supervisionado) é possível notar que os dados já estão rotulados (têm uma forma) e formam *clusters* bem

definidos. Dessa forma é possível notar três tipos de dados: um *cluster* representado na cor azul, um *cluster* representado pela cor vermelha e outro *cluster* representado pela cor verde. Na Figura 1(b) (método não-supervisionado) é apresentado um exemplo de dados para aprendizado não-supervisionado. Os rótulos dos objetos são desconhecidos e, neste caso, todos as amostras estão representadas com a mesma cor (cinza, neste caso). Na Figura 1(c) (método semissupervisionado) é apresentado um exemplo de dados utilizados para aprendizado semissupervisionado, onde há alguns objetos rotulados (símbolo x) em menor proporção, e esferas em cinza, em maior proporção, representando objetos não rotulados.

Figura 1 – Exemplos de conjuntos de dados entradas para métodos de aprendizado indutivo



Fonte: Produzido pelo autor.

Na seção 2.2 é apresentado o *clustering* de dados, tarefa de aprendizado aplicada neste trabalho.

2.2 Clustering

O *clustering* é um método de AM muito utilizado visando tratar problemas de organização de dados. O objetivo é agrupar objetos em diferentes *clusters* de acordo com o grau de associação: entre objetos do mesmo *cluster* o grau de associação é alto, ou seja, os objetos são semelhantes considerando determinadas características. Em contrapartida, o grau de associação entre objetos de *clusters* diferentes é baixo (XU; WUNSCH, 2008). As técnicas de *clustering* podem ser úteis para várias finalidades em aplicações de mineração de dados, como: mineração de texto e recuperação de informação (ZHAI; MASSUNG, 2016); *marketing* (CHATFIELD; COLLINS, 2018) e diagnósticos médicos (THANH; ALI; SON, 2017; THONG et al., 2015).

Segundo Xu e Wunsch (2008), o objetivo do *clustering* é separar um conjunto de dados finito e não rotulado em um conjunto finito e discreto de estruturas naturais ocultas. Os objetos de dados no mesmo *cluster* devem ser semelhantes entre si, enquanto os objetos de dados em *clusters* diferentes devem ser diferentes uns dos outros. As semelhanças e diferenças devem ser claras e significativas.

A Figura 2 é formada por 4 veículos que serão utilizados como exemplo para *clustering*. Se o objetivo for agrupar os veículos pela marca, por exemplo, o *clustering* deve resultar em 3 *clusters*: o primeiro *cluster* composto pela Lamborghini Urus (canto superior direto da Figura 2) e a Lamborghini Gallardo (canto inferior direito da Figura 2), o segundo *cluster* composto pela Ferrari F40 (canto superior esquerdo da Figura 2) e o terceiro *cluster* formado pelo Toyota Sequoia (canto inferior esquerdo da Figura 2). Se o objetivo for agrupar pela categoria a qual pertencem, o *clustering* deve resultar em 2 *clusters*: o primeiro *cluster* composto pela Ferrari F40 e pela Lamborghini Gallardo, pois ambos pertencem a categoria "sport", e o segundo *cluster* composto pela Lamborghini Urus e pela Toyota Sequoia, pois ambos pertencem a categoria "SUV". Um outro modo possível de agrupar os veículos seria pela sua cor predominante. Neste caso o *clustering* deve resultar em 2 *clusters*: o primeiro *cluster* composto pela Ferrari F40 e pela Lamborghini Urus (ambos veículos na cor vermelha) e o segundo *cluster* composto pela Lamborghini Gallardo e Toyota Sequoia (ambos veículos na cor laranja). Pode-se observar que os *clusters* resultantes do processo de *clustering* têm forte dependência do objetivo do *clustering*, ou seja, de acordo com as características determinantes para o processo de divisão dos *clusters*.

Imagine que o objeto de entrada seja um ser humano, como um paciente de um consultório médico, por exemplo. Como característica do objeto paciente pode-se utilizar, por exemplo, os atributos Altura, Sexo, e Doente?. O atributo Altura é definido como um atributo real (ou inteiro, dependendo da configuração dos dados). O atributo Sexo pode ser definido como um tipo de dados categórico com poucos valores (masculino, feminino, outro, etc.). Já o atributo Doente? é um atributo categórico que é representado por um conjunto com apenas dois valores: sim ou não. Vale mencionar que o atributo

Figura 2 – Exemplo de clustering

Fonte: Pixabay.

Doente? pode ser convertido facilmente para valores booleanos (utilizando o valor 1 para representar um paciente doente e o valor 0 para representar um paciente que não está doente).

Existem diferentes métodos e algoritmos que visam agrupar objetos com características semelhantes em seus respectivos *clusters*, como particionais, hierárquicos, baseados na densidade e algoritmos baseados em grafos (REDDY; VINZAMURI, 2018):

– Algoritmos baseados no método hierárquico buscam organizar o conjunto de dados em uma estrutura de dendrograma considerando a proximidade entre os objetos. Tal aplicação produz uma série de *clusters* aninhados, o que permite que *clusters* específicos possam ser selecionados através de um corte no dendrograma. Como exemplo, no algoritmo *Agglomerative Hierarchical Clustering* (AHC) (ou Clustering Hierárquico Aglomerativo), no início do processo, cada objeto corresponde a um *cluster* e são combinados de acordo com a medida de proximidade utilizada (XU; WUNSCH, 2008).

– Algoritmos baseados em densidade definem *clusters* considerando regiões densas, ou seja, regiões que concentram um grande volume de objetos, e tais regiões são separadas por regiões de menor densidade. Um exemplo é o algoritmo *Density Based Spatial Clustering of Application with Noise* (Clusterização Espacial Baseada em Densidade de Aplicações com Ruído), mais conhecido como DBSCAN (ESTER et al., 1996).

– Algoritmos baseados em grafos buscam representar um conjunto de dados utilizando o modelo tradicional de grafo, considerando que cada objeto do conjunto de dados é representado por um vértice no grafo e a proximidade de objetos é representada por uma aresta conectando dois vértices. Assim, no *clustering* baseado em grafos, os elementos dentro de um *cluster* são conectados uns aos outros mas não têm conexão com elementos

fora desse *cluster* (ROY; CHAKRABARTI, 2017). Um exemplo de algoritmo baseado em grafo é o algoritmo *Spectral Clustering* (REDDY; VINZAMURI, 2018).

– Algoritmos particionais buscam decompor um conjunto de dados em um conjunto de *clusters* separados, construindo uma K partição dos dados com cada partição representando um *cluster* distinto, satisfazendo os critérios: cada *cluster* contém pelo menos um exemplo e cada exemplo pertence a exatamente um *cluster*.

Como é do interesse do trabalho aqui apresentado, é dado maior enfoque em algoritmos particionais.

Mais formalmente, segundo Xu e Wunsch (2008), considerando um conjunto de objetos $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_N\}$, onde $\boldsymbol{x}_i = (x_{i1}, x_{i2}, ..., x_{id}) \in \Re^d$, com cada medida x_{ij} chamada de atributo (recurso), o *clustering* particional tenta buscar a K-partição, ou seja, uma partição com K *clusters* de X, onde $C = \{C_1, ..., C_K\}$, para $(K \leq N)$, sendo que N é o número de objetos e K é o número de *clusters*, tal que:

– $C_i \neq \phi, i = 1, ..., K$;

– $\bigcup_{i=1}^{k} C_i = X$;

– $C_i \cap C_j = \phi, i, j = 1, ..., K$ e $i \neq j$.

O algoritmo K-Means (MACQUEEN et al., 1967) é um dos algoritmos particionais mais tradicionais. Os centroides ou protótipos dos *clusters* são, inicialmente, gerados aleatoriamente. Todas as instâncias de treinamento são adicionadas ao *cluster* mais próximo, considerando a distância Euclidiana entre cada exemplo e os centros de *cluster*. Em seguida, os protótipos dos *clusters* são recalculados como o centroide de todos os exemplos do *cluster* e se tornam os novos representantes de seus respectivos *clusters*. As distâncias entre os exemplos e os centroides são recalculadas e todas as instâncias são relocadas ao *cluster* mais próximo, considerando o novo centroide encontrado. Esse processo continua até a convergência, alcançada quando os centroides recalculados correspondem aos centroides da iteração anterior ou a diferença entre centroides de duas iterações consecutivas está dentro de alguma margem predefinida.

Como definido em Xu e Wunsch (2008), o K-means visa minimizar a soma dos erros quadrados. A Equação 1 representa a função objetivo do K-means, onde $\boldsymbol{x}_j^{(i)}$ representa o caso j, $\boldsymbol{m}_i$ representa o centroide para o cluster i e as barras duplas ($\|$ e $\|$) denotam o comprimento do vetor.

$$J = \sum_{i=1}^{N} \sum_{j=1}^{K} \left\| \boldsymbol{x}_j^{(i)} - \boldsymbol{m}_i \right\|^2 \tag{1}$$

Aplicando um procedimento iterativo de otimização, o algoritmo K-means é definido da seguinte forma:

Algoritmo *K*-means:

1. Inicializar uma partição K considerando conhecimento prévio ou aleatoriamente. Calcular a matriz protótipo do *cluster* $M = [m_1, ..., m_K]$;

2. Atribuir cada objeto no conjunto de dados ao *cluster* C_l mais próximo, considerando a Equação 2:

$$\boldsymbol{x}_i \in C_l, se\ \|\boldsymbol{x}_i - \boldsymbol{m}_l\| < \|\boldsymbol{x}_i - \boldsymbol{m}_j\|\, para\, i = 1, ..., N \neq l, e\, j = 1, ..., K; \quad (2)$$

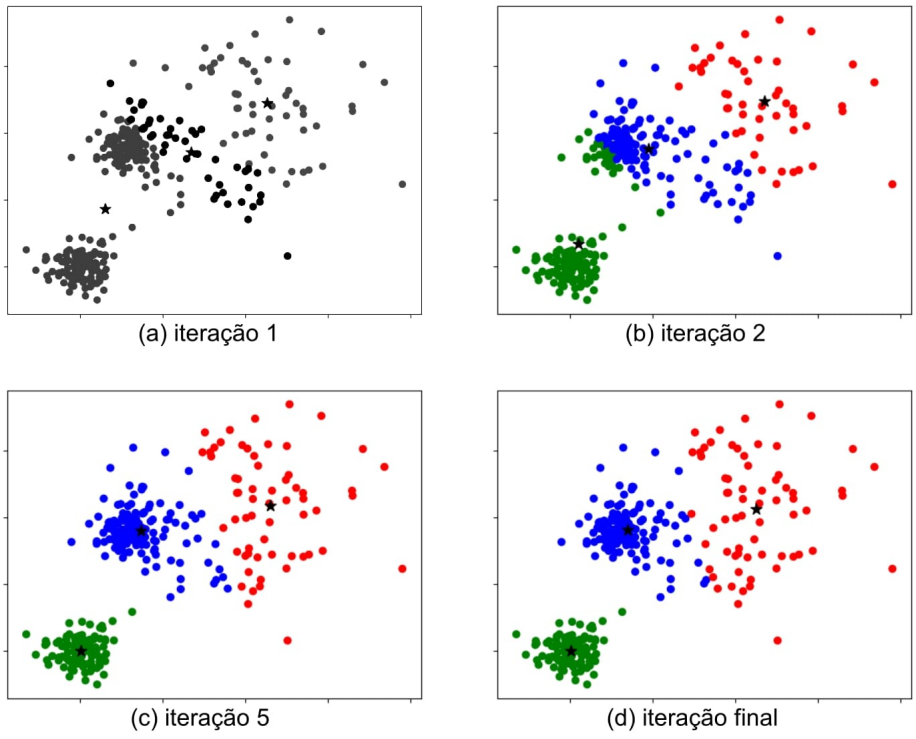
3. Recalcular a matriz do protótipo dos *clusters* com base na atual partição conforme Equação 3, onde N_j é o número de objetos no *cluster* C_j;

$$\boldsymbol{m}_j = \frac{1}{N_j} \sum_{x_i \in C_j} \boldsymbol{x}_i; \quad (3)$$

4. Repita os passos 2 e 3 até que não haja nenhuma alteração para cada *cluster*.

Na Figura 3 é representado o processo de *clustering* e ajuste de centroides de *cluster* durante algumas iterações do K-means.

Figura 3 – Exemplo de iterações e reajustes de centroides de *cluster* em métodos de *clustering*



Fonte: Produzido pelo autor.

Os dados de entrada não trazem informações sobre a qual *cluster* pertencem. Na Figura 3(a) são mostrados objetos e ainda não foram calculadas as distâncias para os centroides (representados por estrelas, em preto). A partir da segunda iteração já existem 3 *clusters* sendo previamente formados: *cluster* de objetos em azul, em verde e em vermelho (Figura 3 (b)) e também é possível notar a atualização da posição dos centroides. Após nova atualização dos centroides na iteração 5 (Figura 3 (c)) é possível notar que objetos que faziam parte do *cluster* exibido em verde agora fazem parte do *cluster* de objetos em azul. Nota-se o movimento dos centroides e os *clusters* sendo definidos. Na Figura é (Figura 3 (d)) está representado o resultado final do processo de *clustering*.

Como mencionado, o problema de *clustering* consiste em, considerando um conjunto finito X de objetos, agrupar tais objetos de acordo com as características presentes nos objetos. Segundo Bezdek et al. (1999), partições em que um objeto pertence a um só *cluster*, referidas por partições *crisp* (rígidas), podem não representar os *clusters* naturais adequadamente, considerando problemas como imprecisão ou dados parciais. Em aplicações práticas e com dados reais, o uso de partições *crisp* pode ser inviável devido a imprecisão, incerteza e outros fatores presentes no conjunto de dados. A incerteza pode ser proveniente da sobreposição de *clusters*.

O *clustering* particional também pode ser feito de forma mais flexível, chamado de *fuzzy clustering*, em que os objetos podem pertencer a mais de um *cluster*, com diferentes graus de pertinência. Com a aplicação da abordagem *fuzzy*, o objetivo do problema de *clustering* se torna a obtenção de uma partição ou pseudo-partição *fuzzy* em um conjunto de dados X. Deste modo é possível obter diferentes graus de relacionamento entre exemplos do conjunto de dados e exemplos podem pertencer a mais de um *cluster* (JAIN, 2010).

Considerando o conjunto de objetos $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$, uma pseudo-partição *fuzzy* P do conjunto X é uma família de conjuntos *fuzzy* de X, $P = \{A_1, A_2, ..., A_c\}$, tal que:

$$\sum_{k=1}^{c} A_k(\boldsymbol{x}_i) = 1, i = 1, ..., n \tag{4}$$

$$0 < \sum_{i=1}^{n} A_k(\boldsymbol{x}_i) < n, k = 1, ..., c \tag{5}$$

Na Equação 4 e na Equação 5, $\boldsymbol{x}_k$ geralmente é um vetor de características, A_c é uma pseudo-partição *fuzzy*, n representa o número de elementos do conjunto X e $A_k(\boldsymbol{x}_i)$ representa o grau de pertinência de $\boldsymbol{x}_i$ em relação a pseudo-partição A_k, considerando a Equação 4, para todo $k \in N_n$ e a Equação 5, para todo $i \in N_c$, onde c é um número inteiro positivo que representa o número de clusters e A representa subconjuntos *fuzzy* de X.

O algoritmo *Fuzzy C*-Means tenta particionar uma coleção finita de $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n\}$ em uma coleção de c *clusters fuzzy* em relação a algum critério. Dado um conjunto finito

de dados, o algoritmo retorna uma lista de $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_c\}$ e uma matriz de partição $W = w_{ij} \in [0, 1]$, com $i = 1, ...n$ e $j = 1, ..., c$ onde w_{ij} informa o grau em que elemento $\boldsymbol{x}_i$ pertence ao cluster $\boldsymbol{c}_j$ (BEZDEK et al., 1999). O algoritmo *Fuzzy C*-Means visa minimizar a função objetivo representada na Equação 6, onde w_{ij} é representado conforme Equação 7:

$$J(W, C) = \sum_{i=1}^{n} \sum_{j=1}^{c} w_{ij}^{m} \left\| \boldsymbol{x}_i - \boldsymbol{c}_j \right\| \tag{6}$$

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|\boldsymbol{x}_i - \boldsymbol{c}_j\|}{\|\boldsymbol{x}_i - \boldsymbol{c}_k\|} \right)^{\frac{2}{m-1}}} \tag{7}$$

Por exemplo, considerando o conjunto $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}$, a pseudo-partição P seria $A_1 = 0.6/\boldsymbol{x}_1 + 1/\boldsymbol{x}_2 + 0.1/\boldsymbol{x}_3$ e $A_2 = 0.4/\boldsymbol{x}_1 + 0/\boldsymbol{x}_2 + 0.9/\boldsymbol{x}_3$.

2.2.1 Medidas de proximidade

Como o objetivo do *clustering* é agrupar objetos com características semelhantes, um elemento de fundamental importância dos algoritmos é a medida de proximidade utilizada. A medida de proximidade entre pares de objetos pode ser uma medida de similaridade ou de dissimilaridade entre esses objetos. Uma medida de proximidade deve ser escolhida considerando-se os tipos e escalas dos atributos. Quando todos os atributos são contínuos, as medidas de dissimilaridade mais utilizadas são as medidas de distância, enquanto que as medidas de similaridade mais utilizadas são as de correlação. Na sequência são apresentadas algumas dessas medidas para objetos descritos por atributos quantitativos.

2.2.1.1 Medidas de distância

Medidas de distância são medidas numéricas que calculam quão diferentes dois objetos são. Duas das principais medidas de distância são apresentadas na sequência.

Distância Euclidiana

A distância Euclidiana é a distância entre dois pontos definida como a raiz quadrada da soma dos quadrados das diferenças entre as coordenadas correspondentes a tais pontos, ou seja, é uma medida da distância em linha reta entre dois pontos no espaço Euclidiano. A distância Euclidiana pode ser calculada como na Equação 8, onde $\boldsymbol{x_i}$ e $\boldsymbol{x_j}$ são objetos de dados d-dimensionais.

$$D(x_i, x_j) = \sqrt{\left(\sum_{l=1}^{d} |x_{il} - x_{jl}| \right)^2} \tag{8}$$

A distância Euclidiana funciona bem quando os dados tem baixa dimensionalidade. É uma das medidas de distância mais usadas por ser simples de implementar e obter bons

resultados em muitos casos de uso. Como desvantagens, em certos casos é necessário primeiro normalizar os dados. Outro problema é que a distância Euclidiana vai perdendo significância a medida que a dimensionalidade dos dados aumenta (AGGARWAL; HINNEBURG; KEIM, 2001).

Distância de Manhattan

A distância de Manhattan entre dois pontos, como, por exemplo, $\boldsymbol{x_i} = (\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_d})$ e $\boldsymbol{x_j} = (\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_d})$, representados como vetores no espaço n-dimensional, é a soma das distâncias em cada dimensão, ou seja, a soma das diferenças absolutas entre os dois vetores (CRAW, 2010). A Equação 9 mostra como a distância de Manhattan entre dois pontos é calculada.

$$Manhattan(x_i, x_j) = \sum_{i,j=1}^{n} |(\boldsymbol{x}_i - \boldsymbol{x}_j)| \tag{9}$$

A distância de Manhattan funciona bem em conjuntos de dados de alta dimensionalidade e também para conjunto de dados com atributos discretos ou binários, pois leva em consideração os possíveis caminhos, de modo realista. Como desvantagens, é uma medida menos intuitiva do que a distância Euclidiana e não apresenta o caminho mais curto entre objetos (AGGARWAL; HINNEBURG; KEIM, 2001).

2.2.1.2 Medidas de Correlação

Um coeficiente de correlação é uma medida numérica de correlação, ou seja, uma relação estatística entre duas variáveis. As medidas de correlação possibilitam verificar como uma ou mais variáveis estão relacionadas entre si (TAYLOR, 1997). No cenário das tarefas de *clustering*, as medidas de correlação permitem avaliar a similaridade entre pares de objetos.

Similaridade de cosseno

A similaridade de cosseno é a medida do ângulo entre dois vetores. Calcula o ângulo formado a partir do conjunto de d dimensões que cada objeto tem, e o ângulo resultante representa o grau de similaridade entre os objetos. Quanto menor o ângulo maior a similaridade entre os objetos. O ângulo pode variar no intervalo de 0 a π. O cálculo da similaridade de cosseno é representado na Equação 10, onde $\boldsymbol{x}_i \cdot \boldsymbol{x}_j$ é o produto (ponto) dos vetores $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$, $\|\boldsymbol{x}_i\|$ e $\|\boldsymbol{x}_j\|$ representam o comprimento dos vetores $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$, respectivamente, e $\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|$ representa o produto vetorial dos vetores $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$.

$$SimCos_{(x_i, x_j)} = \frac{(\boldsymbol{x}_i \cdot \boldsymbol{x}_j)}{\|\boldsymbol{x}_i\| \cdot \|\boldsymbol{x}_j\|} = \frac{\sum_{i,j=1}^{n} x_i x_j}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{j=1}^{n} x_j^2}} \tag{10}$$

A similaridade de cosseno é utilizada frequentemente quando os dados possuem alta dimensão e tal dimensão não é relevante como, por exemplo, para representar textos utilizando contagem de termos, onde os vetores resultantes da representação podem ter dimensões diferentes. Uma desvantagem da similaridade de cosseno é que apenas sua direção do vetor é considerada, não sua dimensão, ou seja, diferenças de valores não são totalmente levadas em consideração como, por exemplo, a diferença na escala de classificação entre usuários diferentes em sistemas de recomendação (SINGH et al., 2020).

Correlação de Pearson

A correlação de Pearson é uma medida usada comumente na tarefa de *clustering* para avaliar a similaridade entre pares de objetos (FACELI et al., 2011). A correlação de Pearson é dada pela Equação 11, onde $\bar{x_i} = \sum_{l=1}^{d} x_i^l/d$.

$$pearson(x_i, x_j) = \frac{covariância(\boldsymbol{x}_i, \boldsymbol{x}_j)}{variância(\boldsymbol{x}_i)variância(\boldsymbol{x}_j)} = \frac{\sum_{l=1}^{d}(x_i^l - \bar{x_i})(x_j^l - \bar{x_j})}{\sqrt{(\sum_{k=1}^{d}(x_i^l - \bar{x_i})^2 \sum_{l=1}^{d}(x_j^l - \bar{x_j})^2)}} \tag{11}$$

Considerando os vetores $\boldsymbol{x}_i$ e $\boldsymbol{x}_j$, a correlação de Pearson é interpretada como o cosseno dos ângulos entre os vetores transformados, ou seja, o cosseno do ângulo entre os vetores correspondentes aos objetos transformados, com os valores de média igual a 0 e variância igual a 1, ou seja, $\boldsymbol{x}_i^{'} = (\boldsymbol{x}_i - \bar{x}_i)/variância(\boldsymbol{x}_i)$ e $\boldsymbol{x}_j^{'} = (\boldsymbol{x}_j - \bar{x}_j)/variância(\boldsymbol{x}_j)$ (FACELI et al., 2011).

Os valores da correlação de Pearson variam entre $[-1, 1]$. Valores próximos de 1 e valores próximos de -1 indicam a similaridade (correlação) entre os objetos; o valor 1 indica que os objetos são diretamente correlacionados e o valor -1 indica que os objetos são inversamente correlacionados, do mesmo modo que a correlação igual a 1 indica que os vetores dos objetos são paralelos e apontam para o mesmo lado e a correlação igual a -1 indica que os vetores dos objetos são paralelos mas apontam para sentidos opostos. A correlação igual a 0 indica que os vetores dos objetos formam um ângulo de 90 graus (FACELI et al., 2011).

Capítulo 3

Representações numéricas de termos e documentos

Este capítulo tem o objetivo de contextualizar o leitor sobre diferentes formas de representação textual, assim como apresentar algumas técnicas de aprendizado de word embeddings.

A modelagem da linguagem é uma etapa fundamental em tarefas de processamento de documentos de texto, como classificação e clustering. É necessário utilizar técnicas de representação de documentos de texto para que então seja possível processá-los automaticamente. O principal objetivo da modelagem é transformar sequências de textos em vetores numéricos, tornando possível a aplicação das tarefas em AM supervisionado, AM não supervisionado e AM semissupervisionado (DIENG; RUIZ; BLEI, 2020).

Esse capítulo está organizado da seguinte forma: na seção 3.1 são abordadas as principais tarefas de pré-processamento de texto; na seção 3.2 são abordadas as formas de representações numéricas para termos e documentos.

3.1 Pré-processamento de textos

Uma coleção de documentos é um conjunto de dados não estruturado, chamado de *corpus*. Na mineração de texto, texto não estruturado se refere a texto em formato nativo e não processado, isto é, que não possui metadados e não pode ser indexado ou mapeado sem que seja preparado para isso, enquanto que o texto estruturado é um texto que passou por processos como análise sintática, adição e remoção de recursos linguísticos, derivando padrões e assim possibilitando análise e interpretação (MINER et al., 2012).

Para tornar possível a utilização dos termos do documento como unidades para representação é preciso considerar algumas etapas de pré-processamento. O propósito é transformar o conjunto de dados original e não estruturado em um conjunto de dados estruturado.

Na Tabela 1 estão representados exemplos de documentos extraídos do *dataset* CSTR, utilizado nesta tese, sem aplicação de quaisquer tarefas de pré processamento.

Tabela 1 – Exemplos de documento de texto sem pré processamento.

Rhetorical (Rhet) is a programming\knowledge
For years, researchers have used knowledge-intensive
This study of the Fall 2002 Computer

Fonte: Produzido pelo autor.

O pré-processamento de textos é a coleção de tarefas em que os documentos de texto são pré-processados com o objetivo de prepará-los para as tarefas de AM. As principais técnicas de pré-processamento de documentos de texto propostas na literatura (PORTER, 1980; VIJAYARANI et al., 2015) são descritas na sequência:

– Remoção de caracteres especiais, como números, datas, símbolos e abreviações: esta etapa do pré-processamento de textos é aplicada com o objetivo de eliminar caracteres que influenciam negativamente no desempenho, prejudicando processos de aprendizado;

– Padronização de altura de caractere: nesta etapa de pré-processamento os caracteres maiúsculos são transformados em caracteres minúsculos. Essa tarefa é importante porque ocorrências de termos iguais, como *'HOME'*, *'Home'* e *'home'*, que possuem a mesma função em um documento de texto são interpretados de modo diferente pelas tarefas de Processamento de Linguagem Natural (PLN). Com tal padronização, os termos passam a ser iguais;

– Lematização (*Lemmatizer*): é comum encontrar na literatura termos como *'forming'*, *'formation'* e *'form'*, entre outras flexões gramaticais, considerando as características da linguagem. Tais flexões podem afetar o desempenho de sistemas de tratamento de texto por serem considerados como termos diferentes. Sendo assim, a lematização é utilizada para agrupar as formas flexionadas de um termo, identificando cada termo pelo lema, com o objetivo de analisá-los como um único item;

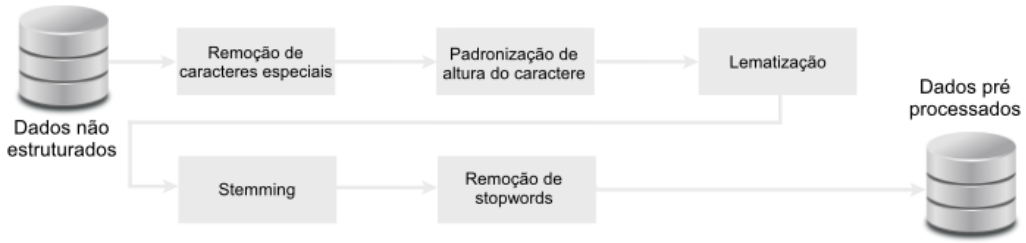
– Stemming: *stemming* é uma técnica usada para extrair a forma básica dos termos através da remoção ou modificação de afixos para que formas de termos que se diferem de maneiras não relevantes possam ser mescladas e tratadas como equivalentes. Os termos não precisam ser sinônimos, porém devem se referir ao mesmo conceito central (como, por exemplo, os termos *"computed"* e *"computable"* são transformados em *"comput"*). Para efeito de uma determinada aplicação, os termos podem ser considerados como equivalentes. O objetivo do *stemming* e da lematização é o mesmo, porém funcionam

de maneira diferente. Enquanto do *stemming* remove o final ou o início do termo, levando em consideração uma lista de prefixos e sufixos comuns que podem ser encontrados em um termo flexionado, a lematização considera a análise morfológica (LIU; ÖZSU, 2009);

– Remoção de *stopwords*: na linguística computacional, o termo *stopwords* é utilizado para se referir a termos que são irrelevantes para um processo computacional e, portanto, são removidos antes do processamento dos dados. Geralmente a remoção de *stopwords* consiste em eliminar termos ou grupos de termos que pertençam a classes fechadas, ou seja, que ocorrem corriqueiramente em um idioma, como conjunções, advérbios, artigos e preposições. A remoção das *stopwords* auxilia na redução de esforço computacional e tempo de processamento do sistema, uma vez que o número de termos analisados é reduzido consideravelmente. A remoção de *stopwords* geralmente é feita com base em comparação com *stoplists*, ou seja, listas de *stopwords*. Caso o termo em questão pertença a uma *stoplist*, é removido do processamento.

Na Figura 4 é apresentado um esquema das etapas de pré-processamento de textos.

Figura 4 – Etapas de pré-processamento de textos.



Fonte: Produzido pelo autor.

Na Tabela 2 estão representados os mesmos documentos representados na Tabela 1 após a aplicação de algumas das técnicas de pré-processamento mencionadas. Neste exemplo é ilustrada a remoção de caracteres especiais, a padronização de altura de caractere e a remoção de *stopwords*. Deste modo, alguns termos e caracteres que ocorrem no exemplo representado na Tabela 1 não ocorrem na Tabela 2, pois foram removidos devido ao pré-processamento.

Tabela 2 – Exemplos de documento de texto com pré processamento.

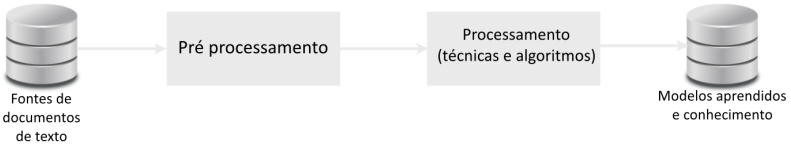
rhetorical rhet programming knowledge
years researcher used knowledge intensive
study fall computer

Fonte: Produzido pelo autor.

Após o pré-processamento, para que os dados textuais possam ser utilizados por tarefas de AM, é necessário criar representações numéricas adequadas à utilização de algoritmos

de aprendizado. Na Figura 5 é representado um fluxograma e as etapas de tratamento de documentos de texto. A representação de documentos tem como objetivo, principalmente, tornar um documento não estruturado compreensível por sistemas/algoritmos para que estes sejam capazes de identificar e capturar as características do documento.

Figura 5 – Fluxograma das etapas de tratamento de documentos de texto, criação de modelos e produção de conhecimento.



Fonte: Produzido pelo autor.

Tradicionalmente, para a representação de *corpus*, eram usadas abordagens de representação baseadas em contagem de termos, visando a extração de atributos, tais como *bag-of-words* e TF-IDF. Essas abordagens são eficazes mas lidam apenas com termos individualmente, o que resulta em uma representação textual sem informações importantes, como contexto e estrutura textual. Atualmente a representação de textos é feita através representação vetorial de termos ou incorporação (*word embeddings*) (JANG et al., 2020).

3.2 Representações numéricas de termos e documentos

Nesta seção serão apresentadas formas de representações numéricas de termos e documentos encontradas com frequência na literatura. As definições e nomenclatura são baseadas em Sebastiani (2002).

3.2.1 Vetorização de contagem (Count Vectorizing)

Considere um *corpus* $D = \{d_1, d_2, \ldots, d_N\}$ e termos exclusivos extraídos do *corpus* D. Os termos exclusivos formarão o vocabulário V e o tamanho da matriz de contagem M será dado por $N \times V$. Cada linha i na matriz M contém a frequência de termos no documento $d(i)$.

Considerando um exemplo simples com dois documentos:

– d_1: *"He is a smart dog. Cat also smart."*

– d_2: *"Tom is a smart worker."*

Considerando a remoção de *stopwords*, o vocabulário criado a partir dos termos exclusivos é composto por: [*'He'*, *'Cat'*, *'smart'*, *'dog*, *'Tom'*, *'worker'*]. A partir desta análise é possível dizer que o conjunto de documentos $|D| = 2$ e o vocabulário $|V| = 6$.

A matriz de contagem M de tamanho 2×6 será representada na Tabela 3 como:

Tabela 3 – Matriz de contagem.

	He	**Cat**	**smart**	**dog**	**Tom**	**worker**
d_1	1	1	2	1	0	0
d_2	0	0	1	0	1	1

Fonte: Produzido pelo autor.

Nesta forma de representação, cada coluna pode ser entendida como um vetor de termos para o termo correspondente na matriz M. Por exemplo, o vetor correspondente ao termo *'smart'* na matriz representada na Tabela 3 é [2, 1] e o vetor correspondente ao termos *'worker'* é [0, 1]. As linhas correspondem aos documentos do *corpus* e as colunas correspondem aos termos no vocabulário. Por exemplo, o conteúdo da segunda linha na matriz (documento d_2) é representado na forma: $d_2 = [0, 0, 1, 0, 1, 1]$, onde todos os termos aparecem com uma ocorrência.

Vale mencionar que este formato possui algumas variações de construção comuns à utilização dessa estratégia de vetorização. Entre elas, uma alternativa é selecionar, utilizando um método adequado, os termos principais com base na frequência. Outra variação comum está na forma como a contagem de cada termo é feita. Nesta variação de formato geralmente é considerado o número de vezes que um termo aparece no documento ou é considerada simplesmente a presença, ou seja, se o termo aparece ou não no documento.

Quando a vetorização de contagem é utilizada em aplicações reais, pelo fato de que existem *corpora* que contém milhões de documentos ou mais, é possível que contenha também um enorme número de termos exclusivos, fazendo com que a matriz tenha grau de esparsidade muito elevado e portanto podendo causar ineficiência no desempenho de algoritmos de processamento (KULKARNI; SHIVANANDA, 2021).

3.2.2 Term Frequency - Inverse Document Frequency (TF-IDF)

A utilização da vetorização Term Frequency - Inverse Document Frequency (TF-IDF) é baseada no método da frequência, considerando, além da frequência de ocorrência de um termo no documento, a ocorrência de um termo em todo o *corpus*. Termos que ocorrem corriqueiramente em uma linguagem (considerando termos em inglês, como: *'the', 'a', 'an', 'in'*, entre outras) tendem a aparecer com maior frequência, enquanto que os termos que são importantes para um documento aparecem com uma frequência menor. Como exemplo, podemos destacar os substantivos e nomes próprios. Por exemplo, um documento sobre Joe Biden conterá mais ocorrências do termo *'Biden'* em comparação com outros documentos enquanto que termos comuns como *'the', 'a', 'an', 'in'* também estarão presentes com alta frequência em todos os documentos (LUHN, 1957).

Desse modo, o cálculo do TF-IDF visa diminuir o peso dos termos comuns que ocorrem em quase todos os documentos e dar mais importância aos termos que aparecem em

apenas um documento ou um subconjunto de documentos. O cálculo TF-IDF penaliza os termos comuns atribuindo-lhes pesos menores e atribui pesos maiores aos termos com maior importância para um documento (LUHN, 1957).

Para fins de exemplificação, considere os documentos d_3 e d_4:

– d_3: *"Dataframe. Pandas Dataframe. Document about Pandas Dataframe. Dataframe"*

– d_4: *"Is. this is about TF-IDF."*

A Tabela 4 e a Tabela 5 que representam a contagem de termos em dois documentos, d_3, na Tabela 4, e d_4, na Tabela 5.

Tabela 4 – Contagem de termos em d_3.

Termo	**Contagem**
Document	1
about	1
Pandas	2
Dataframe	4

Fonte: Produzido pelo autor.

Tabela 5 – Contagem de termos em d_4.

Termo	**Contagem**
this	1
is	2
about	1
TF-IDF	1

Fonte: Produzido pelo autor.

O TF de um termo t no documento d é calculado como na Equação 12 (SONG, 2019):

$$TF(t,d) = \left(\frac{t}{d}\right) \tag{12}$$

Então, neste exemplo, o TF do termo *'about'* no documento d_3 é calculado da seguinte forma: TF = (about, d_3) = 1/8, já que d_3 é formado por 8 termos e o termo *'about'* aparece apenas uma vez. No d_4, o TF é calculado da seguinte forma: TF (about, d_4) = 1/5, já que o documento d_4 é formado por 5 termos e o termo *'about'* também aparece uma única vez. Pode-se dizer que o TF define qual a contribuição de um termo para o documento, ou seja, termos relevantes para o documento costumam aparecer com mais frequência. Neste trabalho foi utilizada a versão normalizada do cálculo do TF.

O IDF (Inverse Document Frequency) é calculado como na Equação 13, onde o dividendo (N) representa o número total de documentos no *corpus* e o *count* ($d \in D : t \in d$) representa o número de documentos em que o termo t está presente (SONG, 2019).

$$IDF(t,D) = log\left(\frac{N}{count(d \in D : t \in d)}\right) \tag{13}$$

Então, o IDF do termo *'about'* é calculado como na Equação 14:

$$IDF(about) = log\left(\frac{2}{2}\right) = 0 \tag{14}$$

Neste tipo de representação, considerando que um termo aparece em todos os documentos, então provavelmente este termo não é relevante para um documento específico. Considerando que um termo aparece em um único documento ou subconjunto de documentos, então provavelmente o termo possui certa relevância para os documentos nos quais está presente.

Como exemplo, considerando o documento d_3, para constatar a relação da relevância de um termo em um documento, considere o cálculo do IDF na Equação 15 para o termo *'Pandas'*:

$$IDF(Pandas) = log\left(\frac{2}{1}\right) = 0,301 \tag{15}$$

O TF-IDF é calculado como na Equação 16 (SONG, 2019):

$$TF\text{-}IDF = TF \times IDF \tag{16}$$

Em vias de comparação de relevância, considerando o documento d_3, o TF-IDF para um termo comum, como *'about*', e um termo mais específico, como *'Dataframe'*, que se mostra relevante para o documento d_3, o resultado é mostrado na Equação 17:

$$\begin{aligned} TF\text{-}IDF(about, d_3) &= (1/8) * (0) = 0 \\ TF\text{-}IDF(about, d_4) &= (1/5) * (0) = 0 \\ TF\text{-}IDF(Dataframe, d_3) &= (4/8) * 0,301 = 0,15 \end{aligned} \tag{17}$$

No documento d_3, o método TF-IDF penaliza pesadamente o termo *'about'*, mas atribui maior peso ao termo *'Dataframe'*. Na aplicação do TF-IDF, isto indica que *'Dataframe'* é um termo importante para o documento d_3 no contexto de todo o *corpus*.

3.2.3 Matriz de co-ocorrência com janela de contexto fixa

A ideia da aplicação da matriz de co-ocorrência parte do pressuposto que termos semelhantes terão contexto semelhante. Por exemplo, considerando as sentenças *"Curry is a basketball player"* e *"Bryant was a basketball player"*. O contexto permite deduzir que *'Curry'* e *'Bryant'* são similares.

Neste tipo de formato de representação existem duas variáveis importantes que devem ser analisadas: a co-ocorrência e a janela de contexto. Considerando um *corpus* de entrada D, a co-ocorrência de um par de termos, digamos w_1 e w_2, é calculada pelo número de vezes que os termos apareceram juntos em uma janela de contexto. Por sua vez, a janela de contexto é definida como o número de termos que devem ser considerados

no contexto e a direção (direita e esquerda) a se considerar. Por exemplo, considere a sentença representada na Tabela 6:

Tabela 6 – Representação de janela de co-ocorrência.

time	to	learn	about	the	co-occurrence	matrix	vectorizing

Fonte: Produzido pelo autor.

No exemplo representado na Tabela 6, os termos destacadas na cor verde representam uma janela de contexto de tamanho 2, considerando como centro ou termo central o termo *'learn'*. Neste exemplo, para calcular a co-ocorrência, somente esta janela de termos é considerada. Também é importante notar que a janela de contexto é considerada em ambas direções, tanto termos anteriores (termos à esquerda) quanto posteriores (termos à direita) ao termo central. Considerando a janela de contexto tendo como o termo central *'co-ocurrence'*, os termos de contexto são representados, em verde, na Tabela 7:

Tabela 7 – Janela de contexto.

time	to	learn	about	the	co-occurrence	matrix	vectorizing

Fonte: Produzido pelo autor.

Para calcular a matriz de co-ocorrência, considere como exemplo o *corpus D*:
D = AI is not boring. AI is daedal. It is cool.
Considerando o exemplo da matriz de co-ocorrência exemplificada na Tabela 8, é destacado em vermelho o número de vezes que *'AI'* e *'is'* aparecem considerando a janela de contexto de tamanho 2. Dessa forma é possível observar que isto ocorre 4 vezes. Na Tabela 8 é ilustrada todas as ocorrências do *corpus* considerando a janela de contexto de tamanho 2.

Tabela 8 – Ocorrências no corpus C.

	AI	**is**	**not**	**boring**	**daedal**	**cool**
AI	0	4	2	1	2	1
is	4	0	1	2	2	1
not	2	1	0	1	0	0
boring	1	2	1	0	0	0
daedal	2	2	0	0	0	0
cool	1	1	0	0	0	0

Fonte: Produzido pelo autor.

Do mesmo modo, o termo *'boring'* não aparece nenhuma vez com o termo *'daedal'* na janela de contexto e, portanto, foi atribuído o valor 0, destacado em azul na Tabela 8. Na Tabela 9 estão representadas todas as ocorrências entre os termos *'AI'* e *'is'*, onde cada linha representa uma ocorrência dos termos *'AI'* e *'is'* na janela de contexto.

Como vantagens, o modelo de representação de matriz de co-ocorrência apresenta:

Tabela 9 – Contagem de co-ocorrências.

AI	is	not	boring	AI	is	daedal	AI	is	cool
AI	is	not	boring	AI	is	daedal	AI	is	cool
AI	is	not	boring	AI	is	daedal	AI	is	cool
AI	is	not	boring	AI	is	daedal	AI	is	cool

Fonte: Produzido pelo autor.

– Preservação das relações contextuais entre os termos, característica muito relevante e importante em diversas aplicações;

– Produz representações de vetores de termos consideravelmente precisas;

– É necessário calcular para a construção do modelo uma única vez e o modelo pode ser recuperado e utilizado a qualquer momento, diversas vezes, em diferentes aplicações.

3.2.4 Bag-of-Words (BoW)

Bag-of-Words (BoW) é um método utilizado para a representação de textos que descreve a ocorrência de termos dentro de um documento. O método transforma textos em vetores de comprimento fixo através da contagem do número de vezes que um termo aparece em um documento (vetorização). Como a ocorrência de um termo em um documento é muitas vezes representativa com relação ao conteúdo ou assunto de um texto, a multiplicidade é utilizada para determinar os possíveis assuntos contidos no documento (LI; JAIN, 1998).

Como exemplo, considere as sentenças S_1 e S_2:

– S_1: *"learning partitional and agglomerative clustering"*

– S_2: *"learn machine learning"*

Considerando as sentenças S_1 e S_2, o vocabulário V composto da seguinte forma: V = [*"learning", "partitional", "and", "agglomerative", "clustering", "learn", "machine"*]. Através das frequências dos termos do vocabulário nas sentenças é possível construir o vetor de cada sentença:

– Sent1 = [1, 1, 1, 1, 1, 0, 0]

– Sent2 = [1, 0, 0, 0, 0, 1, 1]

Perceba que os vetores de sentença e o vocabulário têm a mesma dimensão. Sendo assim, a representação utilizando BoW tende a produzir uma representação esparsa, o que acaba prejudicando o desempenho de tarefas subsequentes, como *clustering* e classificação (ZHANG; LI; WANG, 2019).

O modelo de construção do BoW, apesar de suas vantagens, infelizmente apresenta limitações consideráveis. Entre estas limitações, podem ser destacadas: alta dimensionalidade da representação dos documentos, alta esparsidade dos vetores, perda de correlação

com termos adjacentes e perda de relação contextual existente entre os termos em um documento (MIKOLOV et al., 2013b). Considerando tais limitações, Mikolov et al. (2013a) apresenta uma técnica com o objetivo de aprender vetores de termos a partir de conjuntos de dados cujo vocabulário é composto por milhões de termos, com o objetivo de reduzir a dimensionalidade dos vetores de termos além de manter a correlação existente entre os termos em um documento. Tal técnica de representação de termos em um documento é baseada em Word Embeddings.

3.2.5 Word embeddings

O conceito word embedding (também as vezes chamado de *Word Representation* ou, em português, representação de palavras) foi introduzido por Hinton et al. (1986) e é aplicado, de modo geral, à conjuntos de dados e métodos de seleção de características. Mais especificamente, word embedding é uma abordagem aplicada à mineração de textos que utiliza um conjunto de documentos de texto como entrada e transforma cada termo em sequências numéricas.

O objetivo principal da aplicação de word embedding é mapear termos ou sequências de termos em um espaço contínuo, geralmente de baixa dimensão. Os termos do vocabulário formado pelo conjunto de documentos de texto são mapeados e transformados em conjuntos de vetores em um espaço p-dimensional, onde cada dimensão do vetor corresponde a um número real. O mapeamento produz representações vetoriais do vocabulário (termos ou sequências de termos) daquele conjunto de documentos de texto (MIKOLOV et al., 2013a).

Uma das características mais relevantes das word embeddings é que suas estratégias de implementação tornam possível recuperar informações sobre o contexto e significado dos documentos, o que possibilita capturar o contexto de um termo em um determinado documento, assim como semelhanças contextuais e sintáticas, além da relação existente com qualquer outro termo do vocabulário, considerando o modo como cada termo foi empregado naquele documento (ALMEIDA; XEXÉO, 2019).

Como word embeddings carregam informações sobre contexto e significado, como características sintáticas e contextuais, pode-se considerar que termos que apresentam significados contextuais semelhantes possuem representações vetoriais semelhantes/próximas. Da mesma forma, termos que têm significados contextuais diferentes possuem representações vetoriais diferentes/distantes. Vale mencionar que cada termo possui uma representação vetorial única no conjunto de word embeddings.

O aprendizado de tais representações vetoriais é feito utilizando tanto técnicas supervisionadas quanto não supervisionadas. O aprendizado utilizando técnicas supervisionadas normalmente é feito utilizando modelos treinados de Redes Neurais Artificiais (RNAs) considerando tarefas como análise de sentimentos e classificação. Já o aprendizado utili-

zando técnicas não supervisionadas geralmente realizam análise estatística de documentos (ALMEIDA; XEXÉO, 2019).

O uso da representação de termos, como na forma de *word embeddings*, apresenta relevância considerando estratégias do PLN porque estratégias computacionais que lidam com dados textuais necessitam de representações adequadas. Sendo assim, a representação numérica, principalmente a representação de termos em vetores numéricos, permitem, entre outras possibilidades, quantificar termos.

Os algoritmos, em geral, como os algoritmos de aprendizado de máquina, não são capazes de interpretar e processar sequências de texto da forma em que, geralmente, estão disponíveis. Como mencionado, os algoritmos precisam que estas sequências de texto sejam representadas em formato numérico para que então possam ser utilizadas em tarefas de aprendizado.

Devido à grande e crescente quantidade de dados em formato de texto disponível, a representação adequada deste conteúdo possibilita à algoritmos de aprendizado extrair conhecimento e construir modelos de aprendizado, além de aplicações, como aplicações de análise de sentimento, de classificação de documentos ou notícias, tarefas de *clustering*, entre outras.

A utilização de vetores numéricos pode ser uma boa forma de representar termos. A representação na forma de word embeddings, ou seja, textos convertidos em vetores numéricos, ganhou ênfase com o trabalho de Mikolov et al. (2013a). Na sequência serão apresentadas formas de vetorização e termos. São utilizados termos em inglês para exemplificação.

Como exemplo, considere a seguinte sentença: *"Word embeddings are word vectors transformed into numerical values"*. O vocabulário V de um conjunto de textos é a lista de todos os termos exclusivos. Neste caso, o vocabulário V é formado por: [*'Word'*, *'Embeddings'*, *'are'*, *'vectors'*, *'transformed'*, *'into'*, *'numerical'*, *'values'*]. Na sequência desta seção são apresentados alguns modelos de word embeddings.

3.2.5.1 Word2vec

No trabalho de Mikolov et al. (2013a) é apresentado o Word2vec, um modelo de representação composto por dois métodos baseados na previsão que capturam o significado dos termos do documento ou *corpus* considerando o contexto. Uma das possibilidades que tais métodos possuem é permitir comparar vetores, como, por exemplo, a comparação entre os vetores $(king - men + women)$ resultam no vetor similar ao que representa o termo *'queen'*.

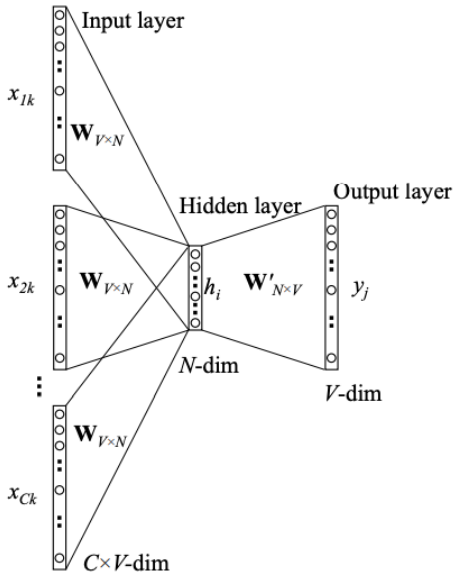
As técnicas apresentadas em Mikolov et al. (2013a) visam contornar os problemas de dimensionalidade e falta de semântica, aprendendo vetores com dimensionalidade reduzida e correlação entre termos do documento. Duas representações são apresentadas: Continuous Bag of Words e Skip-gram. Ambas são arquiteturas para aprender as re-

presentações de termos subjacentes para cada termo usando redes neurais que mapeiam termos para a variável de destino, que também é um termo. Ambas técnicas aprendem pesos que são utilizados como representações de vetores de termos.

Continuous Bag of Words (CBoW)

O modelo Continuous Bag of Words (CBoW) busca prever a probabilidade de um termo em uma janela de contexto, que pode variar de acordo com a configuração desejada, de um único termo a um conjunto de termos. Na Figura 6 é representada, de forma simplificada, o modelo CBoW em uma rede neural onde o contexto é utilizado para prever um termo no centro.

Figura 6 – Modelo Continuous Bag-of-Word (CBoW)



Fonte: Menon (2020)

Considere como exemplo um *corpus* D, onde $D =$ *"this is an example for single context word"*, com vocabulário $|V| = 8$ e com janela de contexto de um termo, considerando ambas direções. O *corpus* D pode ser convertido em um conjunto de treinamento para um modelo CBoW. A entrada é mostrada na Tabela 10, que mostra o vetor *one-hot* codificado para cada possível entrada. *One-hot* é um método de conversão de dados que prepara tais dados para algoritmos, como algoritmos de *clustering* e classificação, buscando melhorar o desempenho (MENON, 2020). Considerando dados categóricos, cada valor categórico é convertido em uma nova coluna e é atribuído um valor binário a essas coluna. Por exemplo, supondo a categoria *"sexo"*, que conteria como opções *"masculino"* e *"feminino"* poderia ser convertida para duas novas colunas, *"masculino"* e *"feminino"* e em tais colunas seriam atribuídos valores binários para representar o gênero dos indivíduos.

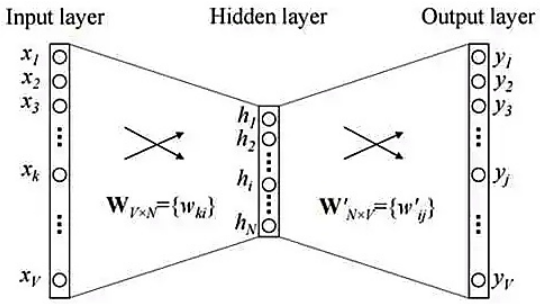
Tabela 10 – Vetores para CBoW.

Alvo	**Contexto**	this	is	an	example	for	single	context	word
this	is	1	0	0	0	0	0	0	0
Is	this	0	1	0	0	0	0	0	0
is	an	0	1	0	0	0	0	0	0
an	is	0	0	1	0	0	0	0	0
an	example	0	0	1	0	0	0	0	0
example	an	0	0	0	1	0	0	0	0
example	for	0	0	0	1	0	0	0	0
for	example	0	0	0	0	1	0	0	0
for	single	0	0	0	0	1	0	0	0
single	for	0	0	0	0	0	1	0	0
single	context	0	0	0	0	0	1	0	0
context	single	0	0	0	0	0	0	1	0
context	word	0	0	0	0	0	0	1	0
word	context	0	0	0	0	0	0	0	1

Fonte: Produzido pelo autor.

A matriz mostrada na Tabela 10 é processada por uma rede neural com três camadas: uma camada de entrada, uma camada oculta e uma camada de saída. Na camada de saída são somados os pesos neuronais obtidos na camada oculta. A propagação funciona para calcular a ativação dos neurônios da camada oculta. Na Figura 7 é ilustrada uma rede neural formada por uma camada de entrada, uma camada oculta e uma camada de saída.

Figura 7 – Representação de rede neural para word embedding



Fonte: Mikolov et al. (2013a)

Para representar o processamento da matriz de dados pelo algoritmo CBoW, considere as seguintes etapas:

a) A camada de entrada e o alvo, ambos são codificados em um único ponto de tamanho $[1 \times V]$, com $|V| = 8$ neste exemplo;

b) Dois conjuntos de pesos: um conjunto de pesos entre a camada de entrada e a camada oculta e um conjunto de pesos entre a camada oculta e a camada de saída. O tamanho da matriz de entrada da camada oculta é $[V \times N]$ e o tamanho da

matriz de saída da camada oculta é $[N \times V]$, onde N é o número de dimensões que representa o termo e o número de dimensões de embeddings. Vale mencionar que a escolha de N é aleatória, pois representa um hiper parâmetro para redes neurais. Além disso, N é o número de neurônios na camada oculta que, neste exemplo, $N = 4$;

c) Não há função de ativação entre as camadas;

d) A entrada do algoritmo (linha correspondente na matriz) é multiplicada pelos pesos na camada de entrada e na camada oculta;

e) A entrada oculta então é multiplicada por pesos na camada de saída e a saída é calculada;

f) O erro (diferença) entre o valor de saída real e o valor de saída desejado é calculado e retro-propagado para reajustar os pesos; e

g) Os pesos entre a camada oculta e a camada de saída são considerados para a representação vetorial do termo.

Comparando Multi-layer Perceptron (MLP) e CBoW, uma diferença considerável é que o gradiente de erro em relação aos pesos de entrada e saída da camada oculta são diferentes, assim como as funções de ativação das MLPs, que geralmente são sigmoidais, enquanto que no CBoW são lineares. Com relação às tradicionais MLPs, o CBoW apresenta algumas vantagens e desvantagens. Como vantagens do CBoW, é possível citar:

– Geralmente apresenta desempenho superior;

– Apresenta baixo consumo de memória.

Como desvantagens do CBoW, é possível citar:

– A composição de cada vetor produzido é uma medida baseada no contexto do termo. Por exemplo, se no mesmo *corpus* existir sentenças ou documentos contendo o termo *'Apple'*, que pode representar uma fruta ou uma empresa, o cálculo do vetor resultante é feito com base em ambos contextos sobre os dois temas, frutas e empresas. Este tipo de ocorrência pode prejudicar o desempenho do algoritmo;

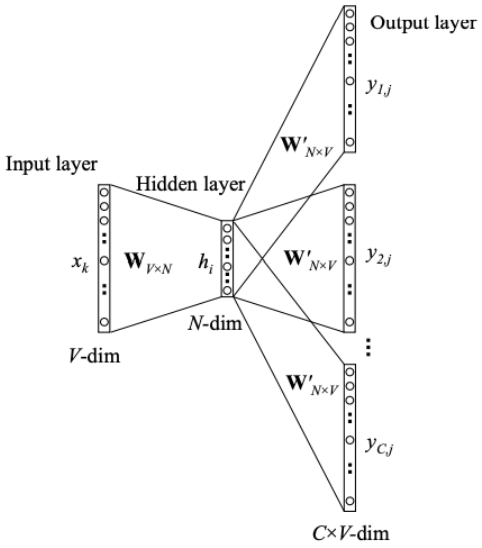
– Os dados de treinamento necessitam de pré-processamento e o algoritmo CBoW precisa ser otimizado para que o custo computacional não seja muito elevado.

Skip-gram

Enquanto que o modelo CBoW busca prever a probabilidade de um termo em uma janela de contexto, no modelo Skip-gram, a representação distribuída do termo de entrada é usada para prever o contexto, ou seja, prever os termos vizinhos, com distância determinada através do tamanho da janela de contexto. Da mesma forma que o CBoW, o objetivo é aprender os pesos da camada oculta e então construir os vetores de termos. Na

Figura 8 é representado, de forma simplificada, o modelo Skip-gram de uma rede neural onde o termo central é utilizado para prever o contexto (MIKOLOV et al., 2013a).

Figura 8 – Modelo Skip-gram



Fonte: Menon (2020)

Como o modelo Skip-gram tem uma construção semelhante ao modelo CBoW, o vetor de entrada também é semelhante ao vetor do modelo CBoW e os cálculos para a ativação das camadas ocultas são os mesmos. A principal diferença entre os modelos é a forma como relacionam os termos de contexto.

Como exemplo, considere a sentença *"I will continue studying word embeddings"* com tamanho de janela igual a 2. Considerando que o centro seja o termo '*studying*', os termos vizinhos são *['will', 'continue', 'word', 'embeddings']*, produzindo os seguintes pares de termos de entrada e de destino: *['studying', 'will']*, *['studying', 'continue']*, *['studying', 'word']* e *['studying', 'embeddings']*. Na Tabela 11 são representas as amostras de treinamento geradas pela sentença. O termo destacado em verde é o termo central e os termos em amarelo representam as termos de contexto.

É importante salientar que a proximidade de cada termo com o termo no centro não tem relevância maior, ou seja, desde que estejam na janela de contexto, possuem a mesma relevância. Sendo assim, considerando como centro o termo *'studying'*, os termos *'will'* e *'embeddings'* têm a mesma relevância tanto quanto os termos *'continue'* e *'word'*.

De acordo com os pares de entrada, dois erros são calculados, considerando as variáveis de destino. São obtidos dois vetores de erro que então são processados para obter um vetor de erro que é retropropagado para atualizar os pesos. Os pesos entre a camada de entrada e a camada oculta são considerados como a representação do vetor de termos,

Tabela 11 – Amostras de treinamento.

Entradas						Pares
I	will	continue	studying	word	embeddings	(will, i), (will, continue), (will, studying)
I	will	continue	studying	word	embeddings	(continue, i), (continue, will), (continue, studying), (continue, word)
I	will	continue	studying	word	embeddings	(studying, will), (studying, continue), (studying, word), (studying, embedding)
I	will	continue	studying	word	embeddings	(word, continue), (word, studying), (word, embedding)

Fonte: Produzido pelo autor.

resultantes do treinamento, com a mesma função de ativação (JATNIKA; BIJAKSANA; SURYANI, 2019).

Como vantagens, o modelo Skip-gram apresenta:

– Possibilidade de utilização de qualquer texto bruto (*corpus* sem pré-processamento) para o aprendizado;

– Tendência a ter bom desempenho considerando pequena quantidade de dados de treinamento (MIKOLOV et al., 2013a);

– Precisão ligeiramente melhor (quando comparado ao modelo CBoW) para os termos que aparecem com frequência nos textos de treinamento (JATNIKA; BIJAKSANA; SURYANI, 2019).

3.2.5.2 Doc2vec

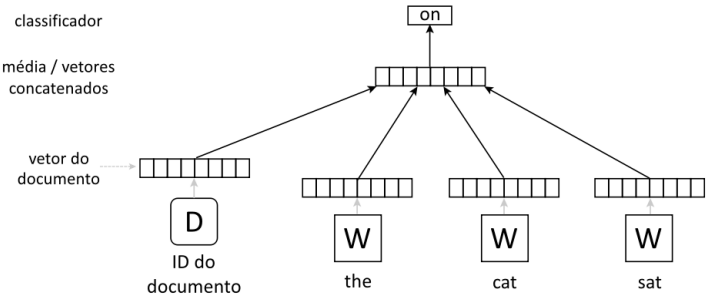
Doc2vec é uma abordagem de aprendizado não supervisionado utilizada para criar uma representação vetorial de um conjunto de termos considerando-os como unidades, ou seja, criar uma representação numérica de um documento, independentemente do tamanho do documento (LE; MIKOLOV, 2014). O modelo Doc2vec é baseado em Word2vec, diferenciando pela adição de outro vetor (ID do documento) à entrada. Diferentemente dos termos, os documentos não contém nenhuma estrutura gramatical lógica. Dessa forma, para que o modelo possa aprender, o vetor ID do documento precisa ser adicionado ao modelo Word2vec (LE; MIKOLOV, 2014).

Existem dois modelos de Doc2vec: Distributed Memory version of Paragraph Vector (PV-DM) e Distributed Bag of Words do Paragraph Vector (PV-DBOW).

O modelo Distributed Memory version of Paragraph Vector (PV-DM) atua como uma memória que guarda o termo faltante considerando o contexto atual. Enquanto os vetores de termos representam o conceito de um termo, o vetor de documentos visa representar o conceito de um documento. PV-DM é um modelo semelhante ao Continuous-Bag-of-Words (CBoW no Word2vec), que busca prever o termo de destino, considerando o

contexto, com a adição de um ID do documento. No CBoW apenas termos são utilizadas para prever o próximo termo, enquanto que no PV-DM também é adicionado outro vetor de atributos, que é exclusivo do documento. Assim, ao treinar os vetores de termos $\boldsymbol{W}$, o vetor de documento $\boldsymbol{D}$ também é treinado e, ao final do treinamento, armazena uma representação numérica do documento. Na Figura 9 é apresentada uma representação do modelo PV-DM.

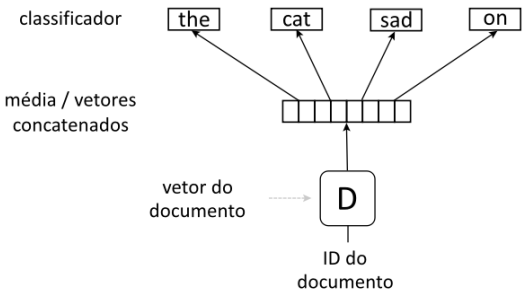
Figura 9 – Modelo Distributed Memory version of Paragraph Vector (PV-DM)



Fonte: Le e Mikolov (2014)

O modelo Distributed Bag of Words do Paragraph Vector (PV-DBOW) é semelhante ao modelo Skip-gram (Word2vec), cujo objetivo é prever os termos de contexto com base em um termo de destino. No modelo Skip-gram apenas o termo de destino é utilizado como entrada, enquanto que o PV-DBOW também utiliza o ID do documento com o objetivo de prever termos amostrados aleatoriamente do documento. Na Figura 10 é apresentada uma representação do modelo PV-DBOW.

Figura 10 – Modelo Distributed Bag of Words do Paragraph Vector (PV-DBOW)



Fonte: Le e Mikolov (2014)

Os modelos Doc2vec podem ser utilizados da seguinte forma: para treinamento é necessário um conjunto de documentos. Um vetor de termos $\boldsymbol{W}$ é gerado para cada termo e um vetor de documento $\boldsymbol{D}$ é gerado para cada documento. O modelo também

treina pesos para uma camada oculta. Na etapa de inferência, um novo documento pode ser apresentado e todos os pesos são fixados para calcular o vetor do documento (LE; MIKOLOV, 2014).

Segundo Le e Mikolov (2014), a melhor escolha de modelo depende da necessidade de uso. O modelo PV-DBOW apresenta melhor desempenho em documentos curtos, além do bom desempenho e maior velocidade de treinamento quando comparado ao modelo PV-DM. Em situações gerais, o modelo PV-DM apresenta desempenho melhor que o PV-DBOW, porém leva mais tempo para treinar.

3.2.5.3 GloVe

Segundo Pennington, Socher e Manning (2014), as estatísticas de ocorrências de termos em um *corpus* são a principal fonte de informação disponível para métodos não supervisionados aprender em representações de termos. Com base na premissa, argumentam que a abordagem usada por Word2vec não explora totalmente as informações estatísticas sobre co-ocorrências de termos. Visando explorar as estatísticas de ocorrências de termos em um *corpus*, Pennington, Socher e Manning (2014) propõem GloVe (Global Vectors for Word Representation), um método de aprendizado não supervisionado para a produzir representações vetoriais de termos.

GloVe utiliza matriz de co-ocorrência termo-termo para produzir word embeddings. De modo diferente do Word2vec, que utiliza janela fixa de contexto, GloVe considera a frequência em que um termo específico aparece em relação a outro termo no *corpus*. Considere que M representa a matriz de co-ocorrência entre termos, e $M_{i,j}$ representa o número de vezes que o termo j apareceu no contexto do termo i e $M_i = \Sigma_k M_{ik}$ o número que vezes que qualquer termo aparece no contexto no termo i. Deste modo, a probabilidade em que o termo j aparece no contexto de i é calculada como na Equação 18.

$$P_{(i,j)} = \frac{M_{ij}}{\sum_{k=0}^{n} M_{i,k}} \tag{18}$$

A relação das palavras pode ser melhor examinada quando as razões de probabilidades são comparadas. Conside, por exemplo, do ponto de vista da termodinâmica, os termos $k = solid$ (sólido), $i = ice$ (gelo) e $j = steam$ (vapor). Uma vez que o termo *ice* e o termo *solid* se relacionam, P_{ik} apresenta um valor maior do que P_{jk}, já que o termo *solid* e o termo *steam* tendem a não coocorrer frequentemente e, portanto, a proporção P_{ik}/P_{jk} apresenta valor alto. Da mesma forma, considerando que $k = gas$ (gás), a proporção P_{ik}/P_{jk} apresenta valor baixo, pois o termo *gas* tende a ocorrer mais vezes no contexto do termo *steam* do que no contexto do termo *ice*. Quando a proporção é calculada com termos como, por exemplo, *water*, que ocorre tanto no contexto do termo *ice* quanto no contexto do termo *steam*, a proporção P_{ik}/P_{jk} apresenta valor próximo de 1. Do mesmo

modo, para termos como, por exemplo, *fashion*, que geralmente não ocorre no contexto de *ice* e *steam*, a proporção P_{ik}/P_{jk} também apresenta valor próximo de 1. Sendo assim, a proporção ajuda a filtrar o significado preciso dos termos. Considerando o exemplo, Pennington, Socher e Manning (2014) afirmam que a proporção entre P_{ik} e P_{jk} torna possível distinguir termos relevantes de termos irrelevantes, mostrando como aspectos do significado podem ser extraídos diretamente das probabilidades de co-ocorrência.

GloVe baseia-se nas ocorrências de um termo no *corpus*, em duas etapas: a criação da matriz de coocorrências a partir do corpus e a fatoração da matriz para obter os vetores de tais termos. Com base nesta construção, o modelo aprende dois conjuntos de vetores de termos: vetor do termo principal e vetor de contexto, o que produz vetores de word embeddings. GloVe é um modelo de decomposição e a matriz resultante do processo de decomposição contém os vetores de termos. Embora a matriz seja composta pelos vetores finais obtidos pelo modelo, geralmente é utilizada a média ou a soma do vetor principal e do vetor dos termos de contexto (PENNINGTON; SOCHER; MANNING, 2014).

Como vantagens, GloVe apresenta:

– Treinamento rápido;

– Escalabilidade;

– Bom desempenho tanto com *corpora* pequenos quanto *corpora* grandes, assim como em produzir vetores pequenos ou grandes.

Como desvantagens, GloVe apresenta:

– Alto uso de memória;

– Alta sensibilidade com relação à taxa de aprendizado inicial.

3.2.5.4 FastText

Joulin et al. (2016) propõe uma abordagem, chamada FastText, baseada no modelo Skip-gram, onde cada termo é representado como um *bag-of-character* de *p-grams*. Diferentemente de modelos tradicionais, como CBoW e Skip-gram, FastText considera que a representação vetorial está associada a cada caractere e cada termo é representado como a soma dos vetores dos caracteres que compõe tal termo. Dessa forma, como principal diferença para modelos Word2vec e GloVe, ao invés de aprender vetores de termos diretamente, representa termos como um *p-gram* de caracteres.

Considere, por exemplo, o termo $'artificial'$ com o valor de $p = 3$. Desta forma, a representação do termo em FastText é $< ar, art, rti, tif, ifi, fic, ici, ial, al >$. Segundo Joulin et al. (2016), esta forma de representação ajuda a capturar o significado de termos mais curtos, permitindo assim que sejam aprendidos prefixos e sufixos.

Após a representação utilizando *p-grams*, um modelo Skip-gram é treinado para aprender embeddings. Como FastText aprende representações associadas a cada caractere, é

possível gerar o vetor que representa um termo não encontrado no *corpus* de treinamento (BOJANOWSKI et al., 2017).

Joulin et al. (2016) afirma que FastText funciona bem com termos que raramente aparecem em um *corpus*, uma vantagem com relação aos modelos Word2Vec e GloVe. Por aprender *p*-gramas em vez de termo completo para alimentar uma rede neural, pode aprender também o relacionamento entre os caracteres.

3.2.5.5 BERT - Bidirectional Encoder Representations from Transformers

BERT (Bidirectional Encoder Representations from Transformers) (DEVLIN et al., 2018) é um método de pré-treinamento de representações de linguagem baseado na arquitetura Transformer (VASWANI et al., 2017). É geralmente utilizado visando dois objetivos: criar modelos usados para extrair recursos de alta qualidade de linguagem e; ajustar modelos para uma tarefa específica, como, por exemplo, classificação e sistemas de respostas a perguntas.

Em comparação com modelos tradicionais, como Word2Vec, BERT contém algumas vantagens. Enquanto cada termo tem uma representação fixa no Word2Vec, independentemente do contexto em que o termo aparece, BERT produz representações de termos que são informadas dinamicamente pelos termos de contexto. A principal contribuição é generalizar para arquiteturas bidirecionais profundas, permitindo que o mesmo modelo pré treinado possa ser treinado facilmente em vários tipos de tarefas de PLN sem fazer nenhuma mudança importante na arquitetura do modelo.

Como exemplo, considere as sentenças:

– *"I need to go to the bank to deposit money."*

– *"I want fishing on the river bank."*

Nas sentenças apresentadas, o termo *"bank"* está sendo utilizado com significados diferentes. Considerando as sentenças do exemplo, Word2Vec produziria um mesmo vetor de word embeddings para o termo *"bank"* em ambas as sentenças, enquanto BERT produz vetores diferentes para *"bank"* em cada sentença. Isso ocorre porque BERT é um modelo bidirecional, ou seja, aprende informações de modo bidirecional do contexto de um termo.

BERT é um processo organizado em duas etapas: treinamento do modelo de linguagem e o ajuste fino.

Treinamento do modelo de linguagem

Durante o pré-treinamento, o modelo é treinado em dados não rotulados em diferentes tarefas de pré-treinamento. Para ajuste fino, o modelo BERT é inicializado primeiro com os parâmetros pré-treinados e todos os parâmetros são ajustados usando dados rotulados das tarefas anteriores (fluxo de tarefas). Cada tarefa anterior tem modelos ajustados separados, mesmo que sejam inicializados com os mesmos parâmetros pré-treinados.

Cada embedding de entrada é uma combinação de 3 word embeddings:

– Embeddings de posição: BERT aprende e usa embeddings posicionais para expressar a posição dos termos em uma sentença.

– Embeddings de segmentos: o BERT também pode receber pares de sentenças como entradas para tarefas. Dessa forma aprende word embeddings exclusivos para a primeira e a segunda sentença para ajudar o modelo a distinguir entre elas.

– Embeddings de termos: Estes são os word embeddings aprendidos para o termo específico do vocabulário.

Considerando a combinação de embeddings, para um determinado termo, a representação de entrada é construída somando os embeddings de posição, termo e segmentos correspondentes.

BERT utiliza Masked Language Model (MLMs) para entender a relação entre os termos. O MLM é uma etapa de ajuste fino que consiste em, através de uma sentença de entrada, otimizar os pesos para produzir a mesma sentença, na saída da etapa de ajuste. Termos e *tokens* da sentença de entrada são mascarados (omitidos), ou seja, é dado ao BERT uma sentença incompleta para que o BERT a complete. Esta etapa de treinamento permite ajustar o BERT para entender melhor o uso específico da linguagem em um domínio mais específico. Para treinar uma representação bidirecional profunda, é construído um modelo para prever um termo ausente dentro da própria sentença ou sequência de termos. O modelo então se torna capaz de prever o termo alvo em um contexto de várias camadas.

BERT também é treinado na tarefa de Previsão da Próxima Sentença (*Next Sentence Prediction*) para tarefas que exigem uma compreensão da relação entre as sentenças. Nesta etapa do treinamento, dadas duas sentenças, A e B, o objetivo é tornar possível ao modelo identificar se B é a sentença subsequente de A no corpus ou se B é simplesmente uma sentença aleatória. Durante o treinamento, 50% das entradas são um par de sentenças em que a segunda sentença é a sentença subsequente no documento original, enquanto nos outros 50% das entradas uma sentença aleatória do corpus é escolhida como segunda sentença.

Ao treinar o modelo BERT, *Masked Language Model* e *Next Sentence Prediction* são treinados juntos, com o objetivo de minimizar a função de perda das duas estratégias.

Ajuste fino

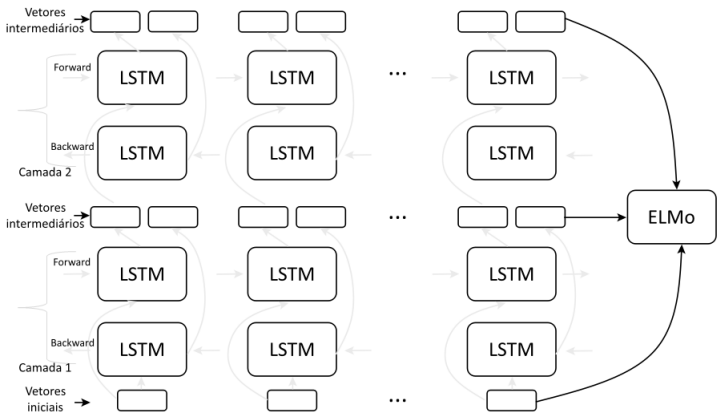
O ajuste do BERT, ou ajuste fino, é uma tarefa de configuração e ajuste do modelo, que é feito para lidar com tarefas de modo diferente com específicas tarefas do PLN. Para cada tarefa são adicionadas etapas de entrada e saída específicas e o BERT ajusta todos os parâmetros de ponta a ponta. Por exemplo, em tarefas de classificação, como a análise de sentimentos, o ajuste é feito de maneira semelhante à classificação da previsão da

próxima sentença, adicionando uma camada de classificação sobre a saída do Transformer (VASWANI et al., 2017) para o termo. Em tarefas de resposta à perguntas usando o BERT, um modelo de perguntas e respostas pode ser treinado aprendendo dois vetores extras que marcam o início e o fim da resposta.

3.2.6 ELMo - Embeddings from Language Models

Embeddings from Language Models (ELMo) (PETERS et al., 2018) é um *framework* de PLN que foi desenvolvido pela AllenNLP[1]. No modelo ELMo, os vetores de termos são calculados usando um modelo de linguagem bidirecional (*bidirectional Language Model* - biLM) de duas camadas. Ao contrário do GloVe e do Word2Vec, ELMo representa vetores de word embeddings de um termo usando a sentença completa que contém tal termo. Dessa forma, ELMo é capaz de produzir embeddings que carregam o contexto do termo usado na sentença e possibilita gerar diferentes word embeddings para o mesmo termo quando aparece em contextos diferentes, considerando sentenças diferentes, o que torna ELMo capaz de produzir word embeddings sensíveis ao contexto. Na Figura 11 é ilustrada uma representação de linguagem bidirecional de duas camadas utilizada no ELMo.

Figura 11 – Exemplos de entradas para métodos de aprendizado indutivo



Fonte: Peters et al. (2018)

No ELMo, os vetores de termos são calculados de acordo com um modelo de linguagem bidirecional de duas camadas biLM (JOSHI, 2019). O modelo biLM tem duas camadas e cada camada tem duas fases (*backward pass* e *forward pass*):

– A arquitetura utiliza rede neural convolucional (CNN) para representar termos de uma sequência de texto em vetores de termos brutos;

– Esses vetores de termos brutos atuam como entradas para a primeira camada do biLM;

[1] https://allenai.org/allennlp

– A fase *forward pass* contém informações sobre um certo termo e o contexto anterior ao termo, ou seja, que ocorrem à esquerda do termo;

– A fase *backward pass* contém informações sobre o termo e o contexto posterior, ou seja, termos que ocorrem à direita do termo;

– O par de informações extraído das fases *backward pass* e *forward pass* é utilizado para produzir os vetores de termos intermediários;

– Os vetores de termos intermediários alimentam a camada seguinte do modelo biLM;

– A representação final (ELMo) é a soma ponderada dos vetores de termos brutos e dos vetores de termos intermediários.

Semelhante as embeddings tradicionais de termos independentes de contexto, como GloVe e Word2vec, as representações produzidas por ELMo podem ser usadas como entrada para uma rede neural para tarefas de aprendizado. O treinamento do ELMo é feito usando uma função objetivo de modelagem de linguagem, onde o objetivo é prever o próximo termo na sequência, o que pode ser feito em ambas direções (*forward pass* e *backward pass*).

A entrada para ELMo é uma sequência de termos $w_1, w_2, ..., w_i, ..., w_n$. Cada termo é convertido em word embedding, independentemente de contexto, por uma rede neural convolucional (CNN). Cada representação alimenta uma rede neural recorrente bidirecional (Long Short-Term Memory - LSTM) de duas camadas (HOCHREITER; SCHMIDHUBER, 1997). A saída da segunda camada da rede LSTM é a entrada de uma camada com o objetivo de prever os termos posterior, w_{i+1} e anterior, w_{i-1}, respectivamente, em cada posição i. Isto permite que os pesos específicos aprendidos possam ser usados posteriormente para combinar todas as camadas no modelo ELMo na posição i e formar a representação de w_1 específica para uma tarefa de aprendizado. Peters et al. (2018) mostram que diferentes camadas deste modelo aprendem diferentes aspectos de um termo, as camadas inferiores aprendem mais recursos sintáticos enquanto as camadas superiores aprendem os aspectos contextuais do termo.

Capítulo 4

Biclustering e Co-clustering

Este capítulo tem o propósito de contextualizar o leitor sobre biclustering e co-clustering, assim como apresentar tipos, esclarecer diferenças e apresentar algoritmos relacionados. Este capítulo está organizado da seguinte forma: Na seção 4.1 é apresentada uma introdução sobre os termos biclustering e co-clustering; na seção 4.2 é apresentada uma formal sobre os as tarefas de biclustering e co-clustering, considerando suas diferenças; e na seção 4.3 são apresentados alguns dos principais trabalhos encontrados na literatura e utilizados nesta tese, como co-clustering baseado na teoria da informação e co-clustering baseado em modelo espectral.

4.1 Introdução

O *biclustering* e o *co-clustering* (também referenciado na literatura como biagrupamento e coagrupamento (MADEIRA, 2011), respectivamente) são estratégias de *clustering* onde critérios de similaridade são aplicados de forma simultânea às linhas e às colunas de matrizes de dados. O objetivo é agrupar simultaneamente os objetos e atributos ou características de um conjuntos de dados. São comumente utilizadas para agrupar objetos, principalmente na área da genética e no processamento de linguagem natural.

A motivação para utilizar o *co-clustering*, segundo Dhillon, Mallela e Modha (2003), é que a estratégia resulta implicitamente em uma redução de dimensionalidade adaptativa em cada iteração. Além disso sua aplicação utiliza menos parâmetros quando comparada à abordagem de *clustering* tradicional. Isto acontece porque, com a utilização do *co-clustering*, são utilizados *clusters* de termos e não apenas termos individualmente. Isso provoca uma redução implícita e adaptativa de dimensionalidade, além da remoção de ruído, resultando em *clusters* melhores.

Na literatura existem aplicações que utilizam *biclustering* e *co-clustering* para pesquisa patrocinada, exibição de anúncios dos anunciantes mais relevantes, inclusão de anunciantes em pesquisas, isolamento de submercados, construção de taxonomias, aplicações genéticas e *clustering* simultâneo de termos e documentos (HENRIQUES; MADEIRA, 2021; HENRIQUES; MADEIRA, 2018; NEVES et al., 2021).

A fim de exemplificar, explicar e apresentar as diferenças entre *clustering*, *biclustering* e *co-clustering*, considere o seguinte exemplo adaptado de Diaz e Peres (2019): Como entrada, um conjunto de documentos de texto D é formado por 5 artigos de notícias ($D = \{d_1, d_2, ..., d_5\}$) e 12 termos $W = \{w_1, w_2, ..., w_{12}\}$. Para definir as relações existentes entre os documentos e termos é criada uma matriz M de valores binários, ou seja, no caso da ocorrência do termo no documento, a posição correspondente na matriz a intersecção documento/termo tem valor 1, enquanto na ausência do termos no documento a intersecção correspondente da matriz tem valor 0.

A partir dos dados de entrada, as tarefas de *clustering*, *biclustering* e *co-clustering* têm como objetivo encontrar similaridade nos dados, como temas ou contextos similares. Assim, o que difere cada tarefa é a forma como os dados de entrada são processados, gerando resultados diferentes em cada um dos casos.

Considerando que o exemplo mencionado seja composto por um conjunto fictício de dados formado a partir de três notícias cujo tema seja tecnologia e duas notícias cujo tema seja cultura e considerando análises por tarefas de *clustering*, *biclustering* e *co-clustering*, os resultados são:

– Para a tarefa de *clustering*: dois *clusters*; o primeiro formado por duas notícias do tema "tecnologia" e o segundo formado por três notícias do tema "cultura", como ilustrado na Figura 12.

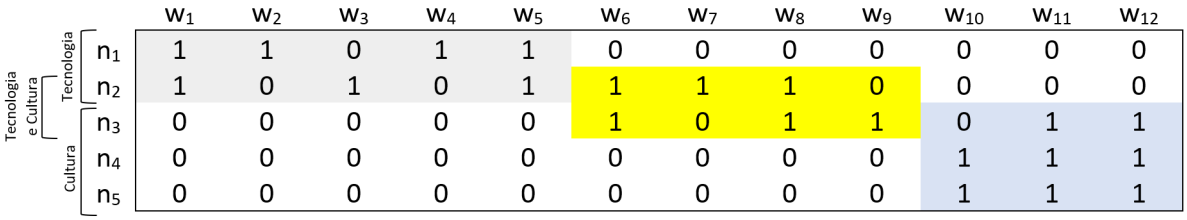
Figura 12 – Exemplo de *clustering* em matriz

		w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}
Tecnologia	n_1	1	1	0	1	1	0	0	0	0	0	0	0
	n_2	1	0	1	0	1	1	1	1	0	0	0	0
Cultura	n_3	0	0	0	0	0	1	0	1	1	0	1	1
	n_4	0	0	0	0	0	0	0	0	0	1	1	1
	n_5	0	0	0	0	0	0	0	0	0	1	1	1

Fonte: Diaz e Peres (2019)

– Para a tarefa de *biclustering*: três *clusters*; o primeiro formado por duas notícias do tema "tecnologia", o segundo formado por três notícias do tema "cultura" e o terceiro formado por duas notícias do que abordam o tema "tecnologia-cultura", como ilustrado na Figura 13.

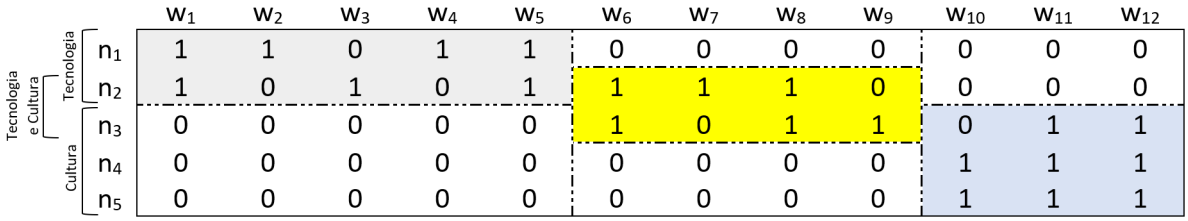
Figura 13 – Exemplo de *biclustering* em matriz



Fonte: Adaptado de Diaz e Peres (2019)

– Para a tarefa de *co-clustering*: três *clusters*; o primeiro formado por duas notícias do tema "tecnologia", o segundo formado por três notícias do tema "cultura" e o terceiro formado por duas notícias do que abordam o tema "tecnologia-cultura", como ilustrado na Figura 14.

Figura 14 – Exemplo de *co-clustering* em matriz



Fonte: Adaptado de Diaz e Peres (2019)

Percebe-se nos exemplos de *biclustering* e *co-clustering* que os termos também são agrupados e *clusters* de termos são associados a *clusters* de documentos, formando os *biclusters* e *co-clusters*.

Embora a aplicação da tarefa de *clustering* unidimensional seja prática, apresente bons resultados e com eficácia, o resultado da tarefa não traz informação sobre os termos, no caso de *clustering* de documentos. Em alguns casos, considerando o contexto, é necessário que os resultados sejam mais específicos. Nestes casos tanto a tarefa de *biclustering* como *co-clustering* são indicadas, pois, na prática, seus resultados são diferenciados em relação ao *clustering* unidimensional.

Considerando o exemplo mencionado acima e os resultados obtidos, existe um *cluster* de notícias em que o tema principal abordado é tanto "tecnologia" quanto "cultura". Um exemplo de notícia que pertencente a este *cluster* pode estar relacionada ao uso da tecnologia visando promover a cultura, tema recorrente na atualidade. Sendo assim, considerando as tarefas de *biclustering* e *co-clustering*, é natural que exista um *cluster* que englobe os dois temas e este tipo específico de notícia.

Como observado na Figura 13 e Figura 14, o resultado apresentado pelo *biclustering* e o resultado apresentado pelo *co-clustering* são equivalentes. Porém, a tarefa de *co-clustering* permite, considerando este exemplo, obter outros quatro *co-clusters*, pois permite a sobreposição de colunas. Assim, toda linha ou coluna da matriz deve fazer parte de algum *co-cluster*.

A diferença entre os resultados obtidos pelas tarefas de *biclustering* e *co-clustering* se torna mais clara com o seguinte exemplo, adaptado de Papadimitriou e Sun (2008). Considere uma matriz M com dimensão 4×5 que é composta por um conjunto de quatro linhas $E = \{e_1, e_2, e_3, e_4\}$ que representam profissionais e um conjunto de colunas S (mecânica, engenharia, programação, automação e robótica) que representam as competências dos profissionais. É possível construir a matriz M utilizando valores binários, ou seja, caso o profissional possua tal competência, a intersecção da matriz M recebe valor 1; caso contrário, recebe valor 0. A Figura 15 ilustra uma representação da matriz M:

Figura 15 – Matriz M para representação de *biclustering* e *co-clustering*

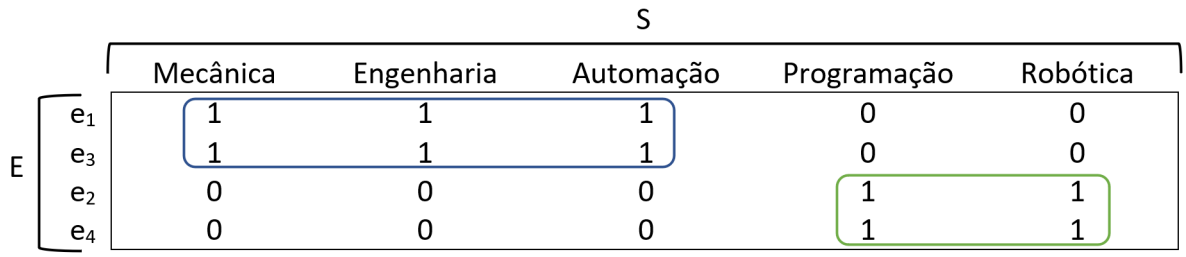
		S				
		Mecânica	Engenharia	Programação	Automação	Robótica
E	e_1	1	1	0	1	0
	e_2	0	0	1	0	1
	e_3	1	1	0	1	0
	e_4	0	0	1	0	1

Fonte: Papadimitriou e Sun (2008)

Visualmente é possível notar que a matriz M pode ser dividida em dois *clusters* diferentes. Pode-se notar a existência de um *cluster* formado pelos profissionais e_1 e e_3 e as suas competências, mecânica, engenharia e automação. O outro *cluster* formado a partir da matriz M é composto pelos profissionais e_2 e e_4 e suas competências, programação e robótica. Reorganizando a matriz M e observando pela perspectiva de um resultado da tarefa de *biclustering*, pode-se ver claramente o que é ilustrado na Figura 16:

De acordo com a reorganização da matriz M podemos ter dois resultados diferentes considerando a aplicação das tarefas de *biclustering* e *co-clustering*. No caso do *biclustering* seria a divisão dos conjuntos de dados em dois *clusters*, assim como vemos na Figura 16. Porém, considerando os resultados obtidos pela tarefa de *co-clustering*, nenhum elemento seria desconsiderado de algum *cluster*, tanto de linhas quanto de colunas. Essa é a principal diferença entre as tarefas de *biclustering* e *co-clustering*, ou seja, *biclustering* produz *clusters* considerando apenas objetos e características relacionadas, já o *co-clustering* produziria algo como *clusters* de engenheiros que não possuem habilidades em programação ou robótica. Neste caso, dois novos *clusters* devem ser considerados em relação a Figura 16, totalizando quatro *co-clusters*, dois *clusters* de linhas e dois *clus-*

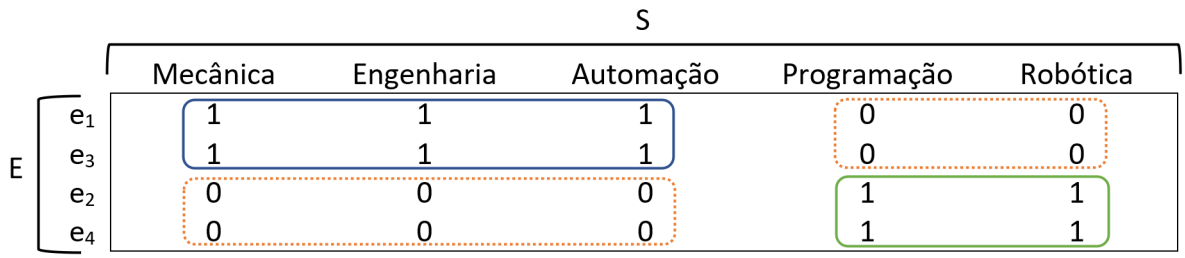
Figura 16 – Reorganização da Matriz M pela perspectiva de *biclustering*



Fonte: Papadimitriou e Sun (2008)

ters de colunas: um novo *cluster* seria formado pelos empregados que não apresentam competência em mecânica, engenharia e automação e o outro *cluster* seria formado pelos empregados que não apresentam competência em programação e robótica. Dessa forma, todas as linhas e colunas da matriz *M* fazem parte de pelo menos um *cluster*. O resultado pode ser visualmente analisado como ilustra a Figura 17:

Figura 17 – Reorganização da Matriz M pela perspectiva de *co-clustering*



Fonte: Papadimitriou e Sun (2008)

Pela definição apresentada por Madeira e Oliveira (2004), podemos afirmar que:

– O *biclustering* é uma estratégia de *clustering* que procura por *biclusters* que podem ser sobrepostos ou não sobrepostos em relação a matriz de dados. Um *bicluster* é formado por um conjunto de objetos e também um conjunto de atributos associados àqueles objetos e, dessa forma, cada objeto no *bicluster* mantém uma forte relação com todos os objetos do mesmo *bicluster* e tendo pouca ou nenhuma relação com objetos de *biclusters* diferentes.

– O *co-clustering* é uma estratégia de *clustering* que procura por um número pré definido de *co-clusters*, a partir de co-partições, onde os objetos pertencentes a uma co-partição apresentam uma forte relação a uma partição de atributos. Cada objeto pertencente a um *co-cluster* de objetos está fortemente relacionado a todos os outros objetos que formam o mesmo *co-cluster*, além de estar fortemente relacionado ao subconjunto de

atributos em comum àquele *co-cluster*. Também é possível que os objetos de um *co-cluster* mantém uma relação considerando o restante dos atributos.

4.2 Caracterização formal

No decorrer desta seção são apresentadas definições formais das duas estratégias de *clustering* e outros conceitos básicos relacionados a estas definições.

4.2.1 Biclustering

De acordo com Madeira e Oliveira (2004), o *biclustering* é uma estratégia de *clustering* de objetos e características que procura por *biclusters* que podem ser sobrepostos ou não sobrepostos, com o objetivo de encontrar a melhor divisão em matriz de dados. Para representação, seja uma matriz A de dimensões $n \times m$, formada por um conjunto de linhas X, sendo que $X = \{x_1, ..., x_i, ..., x_n\}$ e um conjunto de colunas Y, sendo que $Y = \{y_1, ..., y_j, ..., y_m\}$. Podemos representar cada elemento da matriz A por a_{ij}, onde $i = \{1, ..., n\}$, representando o índice da linha e $j = \{1, ..., m\}$, representando o índice da coluna. O par i e j corresponde a uma intersecção na matriz e contém um valor representando a relação entre a linha i e a coluna j. Tal representação é ilustrada na Figura 18:

Figura 18 – Matriz A

$$A = \begin{array}{c|cccccc} & y_1 & & y_j & & y_m \\ x_1 & a_{11} & \cdots & a_{1j} & \cdots & a_{1m} \\ \vdots & \vdots & & \vdots & & \vdots \\ x_i & a_{i1} & \cdots & a_{ij} & \cdots & a_{im} \\ \vdots & \vdots & & \vdots & & \vdots \\ x_n & a_{n1} & \cdots & a_{nj} & \cdots & a_{nm} \end{array}$$

Fonte: Madeira e Oliveira (2004)

Considerando que I é um subconjunto de linhas e J é um subconjunto de colunas da matriz A, podemos afirmar que $I \subseteq X$ e $J \subseteq Y$ e, assim, $A_{IJ} = (I; J)$ é uma submatriz da matriz A composta pelos elementos que fazem parte da intersecção entre as linhas I e J, ou seja, uma submatriz formada pelo conjunto de linhas I e o conjunto de colunas J. Dizemos que o *bicluster* $A_{IJ} = (I; J)$ é um subconjunto de linhas e um subconjunto de colunas (submatriz), sendo que $I = \{i_1, ..., i_p\}$ representa um subconjunto de linhas

onde ($I \subseteq X$ e $p \leq n$), e $J = \{j_1, ..., j_s\}$ representa um subconjunto de colunas onde ($J \subseteq Y$ e $s \leq m$). Dessa forma podemos dizer que o *bicluster* $(I; J)$ é definido como uma submatriz de dimensão $p \times s$ da matriz de dados A. Desse modo, pode-se afirmar que o problema de *biclustering* busca identificar um conjunto de *biclusters* $B_{IJ} = (I;\ J)$.

4.2.1.1 Tipos De Biclusters

A representação apresentada nesta subseção segue a definição apresentada por Madeira e Oliveira (2004). Considere os valores presentes nas Tabela 12, 13, 14, 15 e 16 como valores meramente ilustrativos visando representar padrões em *biclusters*. Os *biclusters* são classificados da seguinte forma:

a) *Biclusters* com valores constantes: São aqueles em que os valores não se alteram dentro do *bicluster*, considerando linhas e colunas. Neste exemplo, o valor 1 em todas as células da representa não existe nenhuma alteração nos valores do *bicluster*, ou seja, todos os elementos, sejam genes em linhas ou condições experimentais em colunas, por exemplo, possuem uma expressão constante e igual a 1 em todas as situações consideradas. A Tabela 12 ilustra a representação deste tipo de *bicluster*.

Tabela 12 – Biclusters com valores constantes.

	c_1	c_2	c_3	c_4
l_1	1	1	1	1
l_2	1	1	1	1
l_3	1	1	1	1
l_4	1	1	1	1

Fonte: Produzido pelo autor.

b) *Biclusters* com valores constantes nas linhas ou nas colunas: Esse tipo de *bicluster* é formado por linhas com valores constantes ou por colunas com valores constantes. Se refere a um padrão onde linhas e/ou colunas da matriz de dados têm valores constantes em todas as condições. Tal *bicluster* é representado na Tabela 13 e na Tabela 14:

Tabela 13 – Biclusters com valores constantes nas linhas.

	c_1	c_2	c_3	c_4
l_1	1	1	1	1
l_2	2	2	2	2
l_3	3	3	3	3
l_4	4	4	4	4

Fonte: Produzido pelo autor.

c) *Biclusters* que apresentam valores coerentes: Neste tipo de *bicluster* pode-se notar uma consistência entre os valores, tanto nas linhas quanto em colunas. Normal-

Tabela 14 – Biclusters com valores constantes nas colunas.

	c_1	c_2	c_3	c_4
l_1	1	2	3	4
l_2	1	2	3	4
l_3	1	2	3	4
l_4	1	2	3	4

Fonte: Produzido pelo autor.

mente os valores de tal tipo de *bicluster* evoluem considerando operações como multiplicação, adição ou subtração. Na Tabela 15 é representado este tipo de *bicluster* considerando uma adição em toda linha, somando 3 para a próxima coluna, ou seja, se a intersecção A_{11} da matriz contém o valor 3, a intersecção A_{12} terá 6.

Tabela 15 – Biclusters que apresentam valores coerentes.

	c_1	c_2	c_3	c_4
l_1	3	6	9	12
l_2	4	7	10	13
l_3	5	8	11	14
l_4	6	9	12	15

Fonte: Produzido pelo autor.

d) *Biclusters* formados a partir de evoluções coerentes: No caso deste tipo de *bicluster* pode ser considerada a evolução coerente como alterações coerentes ao longo dos valores pertencentes ao *bicluster*. No exemplo representado na Tabela 16, podemos notar de que existe uma certa coerência nas linhas, onde os valores diminuem entre c_1 e c_2, aumentam entre c_2 e c_3, e diminuem novamente entre c_3 e c_4.

Tabela 16 – Biclusters formados a partir de evoluções coerentes.

	c_1	c_2	c_3	c_4
l_1	80	15	40	10
l_2	26	7	18	3
l_3	55	25	31	12
l_4	19	9	12	5

Fonte: Produzido pelo autor.

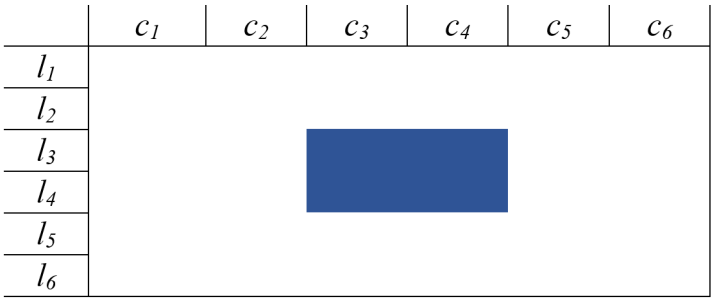
4.2.1.2 Estruturas de bicluster

Além dos tipos de *biclusters* apresentados na seção anterior, outra característica fundamental para este contexto é a estrutura do *bicluster*. A estrutura compreende o comportamento do *bicluster* em relação aos demais *biclusters* e à matriz de dados. Na sequência

serão apresentadas as estruturas de *biclusters*, considerando as definições propostas por Madeira e Oliveira (2004).

– *Bicluster* único: Quando, considerando toda a matriz de dados, é encontrado apenas um *bicluster*. A Figura 19 representa um a estrutura de um *bicluster* único.

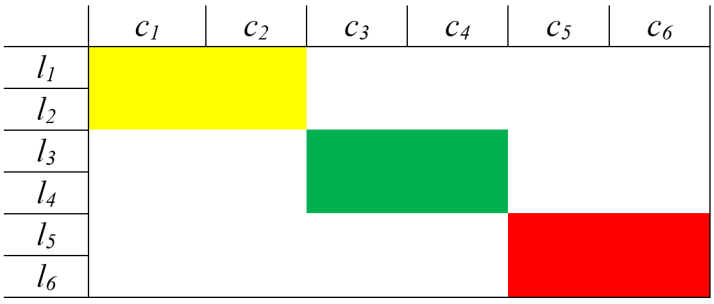
Figura 19 – Bicluster com valor constante



Fonte: Madeira e Oliveira (2004)

– Linhas e colunas exclusivas: Estrutura onde todo *bicluster* é formado por linhas e colunas exclusivas, ou seja, tanto as linhas quanto as colunas que formam o *bicluster* não podem fazer parte de nenhum outro *bicluster*. Nesta estrutura as linhas que compõe o *bicluster* devem apresentar comportamento ou características semelhantes em relação as colunas do *biclusters*, exclusivamente. A estrutura de *biclusters* com linhas e colunas exclusivas é ilustrada na Figura 20.

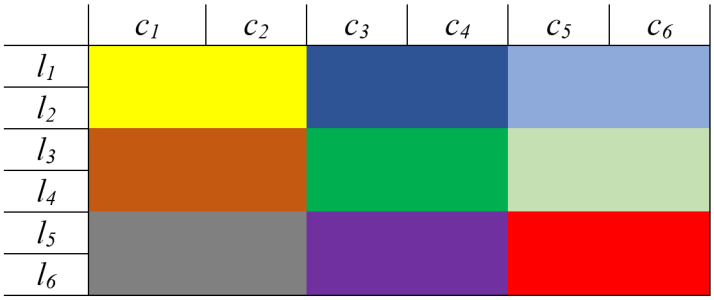
Figura 20 – Representação de linhas e colunas exclusivas.



Fonte: Madeira e Oliveira (2004)

– Tabuleiro de xadrez: O tabuleiro de xadrez é formado quando os dados são divididos em vários *biclusters* não exclusivos, ou seja, que compartilham linhas e/ou colunas, mas que não sejam sobrepostos. Quando os *biclusters* formados geram a estrutura tabuleiro de xadrez e todas as linhas ou colunas da matriz de dados fazem parte de pelo menos um *cluster*, podemos dizer que tal *biclusters* é um *co-cluster*. Um exemplo deste tipo de estrutura é ilustrado na Figura 21.

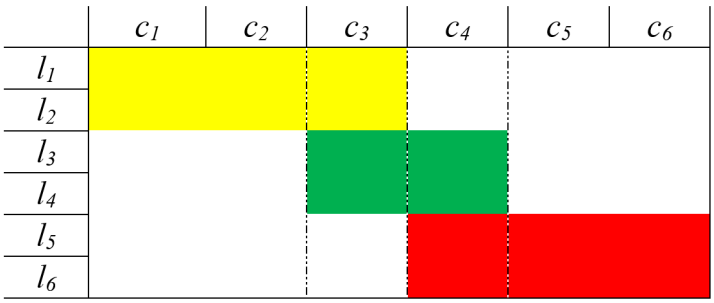
Figura 21 – Representação de tabuleiro de xadrez



Fonte: Madeira e Oliveira (2004)

– Linhas exclusivas: Nesta estrutura de *bicluster*, cada linha da matriz de dados deve pertencer a apenas um *bicluster*. As colunas podem pertencer a um ou mais *biclusters*. É valido mencionar que, nesta estrutura, como as colunas podem pertencer a um ou mais *biclusters*, pelo menos uma coluna deve ser sobreposta para caracterizar tal definição. No exemplo ilustrado na Figura 22 é possível ver que a coluna c_3 faz parte tanto do primeiro *bicluster* (em amarelo) quanto do segundo *bicluster* (em verde), e a coluna c_4 faz parte tanto do segundo *bicluster* quanto do terceiro *bicluster* (em vermelho). Nota-se também que as linhas fazem parte de apenas um *bicluster*.

Figura 22 – Representação de linhas exclusivas

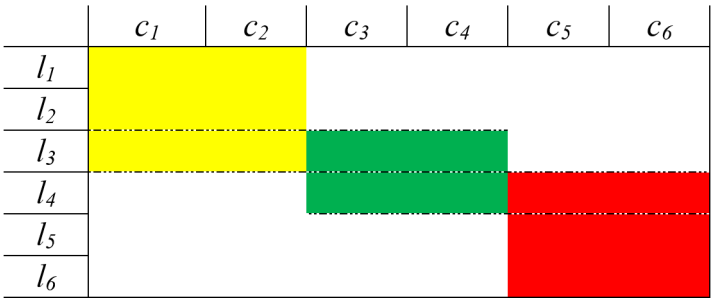


Fonte: Madeira e Oliveira (2004)

– Colunas exclusivas: Ao contrário da estrutura com linhas exclusivas, nesta estrutura as colunas só podem pertencer a um *bicluster*, exclusivamente, enquanto as linhas podem pertencer a um ou mais *biclusters*. Assim como no item anterior, é valido mencionar que, nesta estrutura, como as colunas podem fazer parte de um ou mais *biclusters*, pelo menos uma linha deve ser sobreposta para caracterizar tal definição. No exemplo ilustrado na Figura 23, podemos ver que a linha l_3 faz parte tanto do primeiro *bicluster* (em amarelo) quanto do segundo *bicluster* (em verde) e a linha l_4 faz parte tanto do segundo *bicluster* quanto do terceiro *bicluster* (em vermelho). Nota-se que as colunas

fazem parte de apenas um *bicluster*.

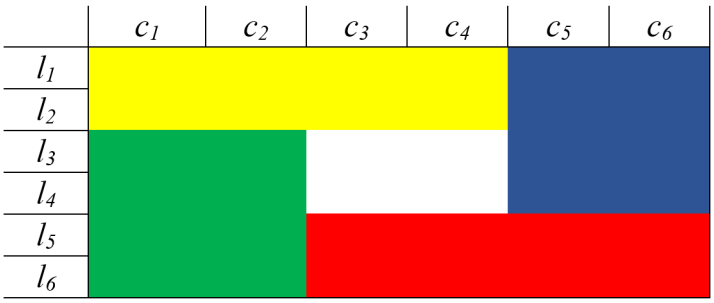
Figura 23 – Representação de colunas exclusivas



Fonte: Madeira e Oliveira (2004)

– *Biclusters* não exclusivos e sem sobreposição: Neste tipo de estrutura existem sobreposições tanto de linhas quanto de colunas, ou seja, tanto linhas quanto colunas podem pertencer a vários *biclusters*, desde que não exista a sobreposição de *biclusters*. Um exemplo desta estrutura é ilustrado na Figura 24.

Figura 24 – Representação de biclusters não exclusivos e sem sobreposição

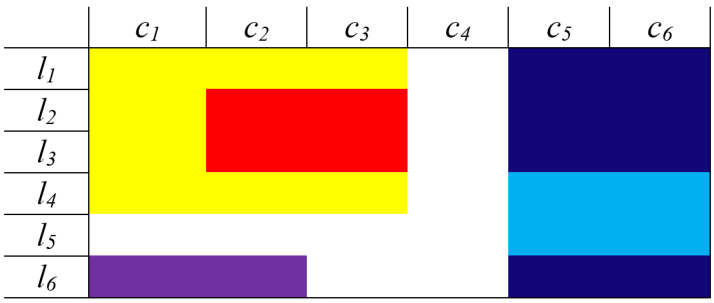


Fonte: Madeira e Oliveira (2004)

– *Biclusters* com sobreposição e com estrutura hierárquica: Neste tipo de estrutura existem *biclusters* dentro de outros *biclusters*. Este tipo de estrutura é ilustrada na Figura 25, onde podemos notar que o *bicluster* 1 (em amarelo) é formado pela submatriz com as linhas l (l_1, ..., l_4) e colunas c (c_1, ..., c_3). Podemos ver também que o *bicluster* 2 (em vermelho) é formado pela submatriz com as linhas l (l_2, l_3) e colunas c (c_2, c_3), fazendo parte do *bicluster* 1. O mesmo ocorre com os *biclusters* 3 (em azul marinho) e 4 (azul piscina). É importante mencionar que neste tipo de estrutura a sobreposição é total, ou seja, um *bicluster* está totalmente inserido em outro *bicluster*.

– *Biclusters* com sobreposição parcial: Neste tipo de estrutura existem *biclusters* sobrepostos, mas não totalmente, ou seja, linhas e colunas podem pertencer a mais de um

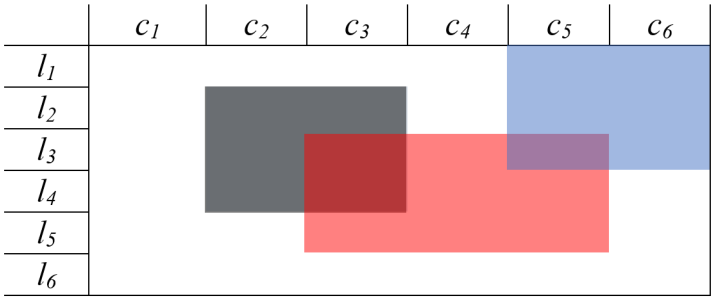
Figura 25 – Representação de biclusters com sobreposição e com estrutura hierárquica



Fonte: Madeira e Oliveira (2004)

bicluster, mas também podem ser exclusivos. Um exemplo deste tipo de estrutura é ilustrado na Figura 26.

Figura 26 – Representação de biclusters com sobreposição parcial



Fonte: Madeira e Oliveira (2004)

4.2.2 Co-clustering

O *co-clustering* é uma tarefa de *clustering* que possui conceitos semelhantes ao *biclustering*. A principal diferença entre as duas tarefas é que no *co-clustering* toda linha ou coluna da matriz deve fazer parte de algum *co-cluster* (MADEIRA; OLIVEIRA, 2004).

Segundo Pensa et al. (2010), o *co-clustering*, formalmente definido considerando uma matriz A de dimensão $n \times m$ (considerando que $A \in R^{n \times m}$), no qual x_{ij} é o elemento que corresponde a intersecção da linha i com a coluna j, e $\mathbf{x}_i$ e $\mathbf{y}_j$ indicam os vetores associados respectivamente à linha i e à coluna j.

Um *co-clustering* $C^{k \times l}$ sobre A produz simultaneamente um conjunto de $k \times l$ *co-clusters*, ou seja, C^r em k *clusters* de linhas associadas a uma partição C^c em l *clusters* de colunas, que otimizam uma determinada função objetivo. Aqui, C^r é o conjunto de *co-clusters* para as linhas, onde cada *co-cluster* C^r_p é um *cluster* de linhas, e C^c é o conjunto de *co-clusters* para as colunas, onde cada *co-cluster* C^r_q é um *cluster* de colunas. Esses

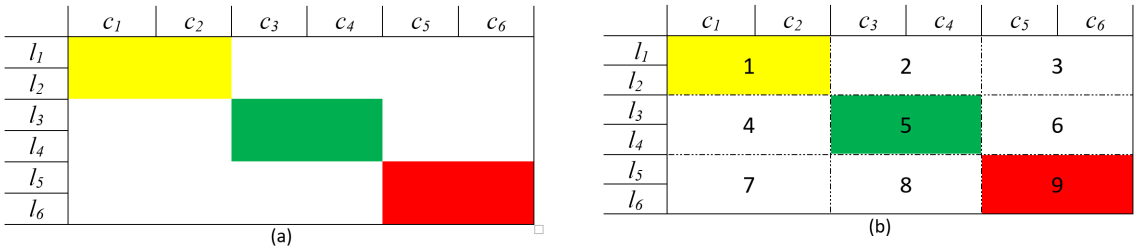
co-clusters são otimizados para maximizar ou minimizar uma função objetivo específica durante o processo de *co-clustering*.

A função objetivo em questão é formulada para encontrar k *clusters* de linhas e l *clusters* de colunas que capturem padrões significativos na matriz A. Então, *co-clustering* busca identificar padrões complexos simultaneamente em ambas as dimensões da matriz, proporcionando uma representação estruturada dos dados através dos *co-clusters*. Pela definição apresentada em Madeira e Oliveira (2004), a função objetivo busca agrupar todas as linhas e colunas da matriz A em algum *co-cluster* específico, portanto $\forall i, j : (i, j) \in C^r \times C^c$.

Sendo assim, a tarefa de *co-clustering* é capaz de identificar relações entre objetos atributos na matriz de dados. A identificação dessas relações é vista como uma maneira de gerar conhecimento sobre o conjunto de dados, indicando como diferentes objetos e atributos estão relacionados entre si.

A Figura 27 ilustra que existe uma forte relação entre as estruturas de *biclustering* e *co-clustering*. Na Figura 27a, podemos ver um exemplo de *biclustering*. Na Figura 27b é apresentado um exemplo de *co-clustering*. Nota-se que os mesmos *clusters* encontrados na tarefa de *biclustering* (Figura 27a) porém, outros *clusters* são considerados. Percebe-se que, considerando a tarefa a ser aplicada aos dados, os resultados serão diferentes.

Figura 27 – Exemplos de *biclustering* e *co-clustering*



Fonte: Madeira e Oliveira (2004)

4.2.3 Algoritmos

Nesta seção serão apresentados alguns tipos de algoritmos para a tarefa de *co-clustering*, assim como definidos pelos autores (DHILLON; MALLELA; MODHA, 2003; KLUGER et al., 2003; GOVAERT; NADIF, 2013). A escolha por apenas tipos de algoritmos para a tarefa de *co-clustering* é porque foi decidido que neste trabalho todas linhas e colunas da matriz de dados deveriam pertencer a algum cluster.

Algoritmos de *co-clustering* são muito utilizados em trabalhos cujos dados são textos, pois apresentam maior eficácia nos conjuntos de dados com alta dimensionalidade

e esparsidade, característica comum deste tipo de dado (AFFELDT; LABIOD; NADIF, 2020; AFFELDT; LABIOD; NADIF, 2021; SELOSSE; JACQUES; BIERNACKI, 2020).

Segundo Dhillon, Mallela e Modha (2003), a aplicação de algoritmos de *co-clustering* em conjuntos de dados textuais implica consideravelmente na redução de dimensionalidade, produzindo *clusters* mais objetivos e minimizando o problema de ruído naturalmente presente nos dados.

Os algoritmos de *co-clustering*, quando aplicados em dados textuais, são de alta complexidade computacional devido à dimensionalidade e esparsidade dos dados (SILVA; PERES; BOSCARIOLI, 2016). Dessa forma, a análise de diferentes métodos utilizados para implementar algoritmos de *co-clustering* é fundamental. A seguir será apresentada uma breve descrição de categorias de métodos de *co-clustering* encontrados na literatura:

– Métodos espectrais (*Spectral Co-clustering*): os algoritmos que implementam o método espectral de *co-clusteirng* consideram os dados de entrada representados na forma de um grafo bipartido. *Spectral Co-clustering* é uma técnica de que visa reduzir conjuntos de dados multidimensionais em *clusters* de dados semelhantes, agrupando dados não organizados, considerando a singularidade. A singularidade refere-se à certos valores associados à matriz que é construída a partir dos dados originais, representando as relações de similaridade entre os pontos de dados. A decomposição espectral dessa matriz envolve o cálculo dos valores e vetores associados, que fornecem informações sobre a estrutura dos dados. A singularidade pode surgir quando os pontos de dados não estão organizados em *clusters* bem definidos (LIU; HAN, 2018). Por exemplo, considerando que os dados de entrada sejam documentos e termos, a presença de um valor diferente de zero na posição a_{ij} da matriz de dados representa uma aresta ligando um documento a um termo, e então os nós são mapeados para um espaço de baixa dimensão. Alguns dos mais famosos algoritmos encontrados na literatura que aplicam métodos espectrais de *co-clusteirng* foram propostos por Dhillon (2001), precursor da aplicação da estratégia, e o algoritmo *"Spectral Biclustering"*, proposto por Kluger et al. (2003), voltado para aplicações genéticas.

– Métodos baseados em modelo probabilístico (*Model Based Co-clustering*): Métodos baseados em modelo probabilístico buscam modelar conjuntamente as linhas e colunas de uma matriz de dados, considerando a possibilidade de que diferentes *co-clusters* de linhas e colunas possam existir. Esses *co-clusters* são identificados pelo algoritmo de *co-clustering*. O ajuste do modelo é feito através da maximização de uma função que mede a probabilidade dos dados observados sob o modelo. Com relação aos métodos de *co-clusteirng* probabilístico, podemos citar o proposto por Bhatia, Iovleff e Govaert (2017), chamado *"blockcluster"*. O algoritmo utiliza modelo de blocos latentes (*Latent Block Models*) e maximização de expectativa.

– Métodos baseados em fatoração de matriz (*Matrix Factorization Co-clustering*): Neste método, a matriz de dados original é aproximada por meio da decomposição em dois

conjuntos menores de matrizes, uma para as linhas e outra para as colunas. Cada uma dessas matrizes é associada a um conjunto de *co-clusters*, representando padrões específicos de *co-clustering*. A técnica busca encontrar fatores que expliquem a estrutura da matriz original, considerando simultaneamente padrões nas linhas e colunas. Métodos baseados em fatoração de matriz como, por exemplo, o algoritmo apresentado por Nie et al. (2020), que aplica aproximações para o problema de decomposição de matrizes. Aproximações são utilizadas porque encontrar uma solução exata para a decomposição de matrizes pode ser computacionalmente desafiador. Nie et al. (2020) aplica corte normalizado, uma medida que considera a qualidade dos *clusters* formados definida como a soma das conexões entre os elementos do mesmo *cluster* dividida pela soma total de conexões.

– Métodos baseados na teoria da informação (*Information Theoretic Co-clustering*): Information Theoretic Co-clustering baseia-se em princípios da teoria da informação para identificar padrões de *co-clusters* em conjuntos de dados. A ideia central é utilizar medidas de entropia e informação mútua para avaliar as relações entre linhas e colunas em uma matriz de dados, sendo que informação mútua é uma medida da dependência estatística entre duas variáveis aleatórias. A teoria da informação fornece métricas que quantificam a dependência estatística entre variáveis aleatórias, e essas métricas são aplicadas nas linhas e colunas da matriz para identificar padrões de *co-clusters*. A estratégia busca agrupar linhas e colunas de modo que a informação mútua entre os *clusters* seja maximizada, indicando assim a presença de padrões significativos. No contexto de *co-clustering*, maximizar a informação mútua entre linhas e colunas significa agrupar essas linhas e colunas de modo que o conhecimento sobre uma forneça a máxima informação possível sobre a outra, revelando assim padrões na matriz de dados (DHILLON; MALLELA; MODHA, 2003; GOVAERT; NADIF, 2013).

– Métodos baseados na modularidade *(Modularity Based Co-clustering)*: Modularity Based Co-clustering tem como ideia central a maximização da modularidade em uma rede formada pelas linhas e colunas de uma matriz de dados. A modularidade é uma medida que avalia a qualidade da divisão de uma rede em comunidades, indicando quão densas são as conexões dentro das comunidades em comparação com as conexões entre elas. A matriz de dados é tratada como uma rede bipartida, onde as linhas e colunas formam os dois conjuntos de vértices. O objetivo é dividir tanto as linhas quanto as colunas em comunidades de modo a maximizar a modularidade global da rede bipartida. Um algoritmo que utiliza este método foi proposto por Labiod e Nadif (2011).

A escolha das abordagens de *co-clustering* apresentadas se deve a:

– Spectral Co-clustering: poder lidar com diferentes formas e tamanhos de *co-clusters*.

– Model Based Co-clustering: flexibilidade para incorporar diferentes distribuições para modelar dados.

– Matrix Factorization Co-clustering: poder lidar eficientemente com dados esparsos.

– Information Theoretic Co-clustering: poder ser aplicado em uma variedade de domínios, especialmente quando há uma dependência estatística significativa entre linhas e colunas.

– Modularity Based Co-clustering: poder ser eficaz em casos nos quais a estrutura dos dados é essencial.

4.3 Trabalhos Relacionados

Segundo Dhillon (2001), o *clustering* é uma ferramenta fundamental no aprendizado não supervisionado, utilizada para agrupar objetos semelhantes dispostos em tabelas de co-ocorrência ou tabelas de contingência, e que a maioria dos algoritmos de *clustering* disponíveis na literatura aplica *clustering* unidimensional (como K-means e DBSCAN, por exemplo), ou seja, *clustering* de uma única dimensão da tabela mediante a semelhanças, considerando a segunda dimensão.

Em algumas situações é desejável que ambas as dimensões de uma tabela (no caso, documentos e termos) sejam co-agrupados ou agrupados simultaneamente, explorando a dualidade entre documentos e termos (ou linhas e colunas). Dessa forma, considerando como exemplo, o algoritmo visa encontrar documentos semelhantes considerando a interação de tais documentos com um determinado *cluster* de termos.

Considerando as características do algoritmo de *co-clustering*, é possível notas algumas características importantes:

– O *co-clustering* diminui monotonicamente a perda de informações mútuas, convergindo para um mínimo local;

– Pode ser aplicado em dados multidimensionais;

– A redução da dimensionalidade em cada etapa ajuda a superar problemas comuns em estratégias de *clustering* unilateral, como esparsidade e alta dimensionalidade;

– Apresenta eficiência computacional.

Pensando na aplicação de tarefas de co-clustering para organizar documentos de texto, alguns trabalhos foram estudados com o objetivo de encontrar algoritmos e estratégias, considerando os desafios do *clustering* de texto, a esparsidade e alta dimensionalidade (DHILLON, 2001; XUAN et al., 2017; HUANG; XU; LV, 2018; FRANÇA, 2016; HONDA et al., 2016; AILEM; ROLE; NADIF, 2017; SALAH; ROGOVSCHI; NADIF, 2016; MEI et al., 2016; AILEM; ROLE; NADIF, 2015; AILEM; ROLE; NADIF, 2016; AFFELDT; LABIOD; NADIF, 2020; LACLAU; NADIF, 2016).

4.3.1 Co-clustering baseado na teoria da informação

De forma sucinta, Dhillon (2001) diz que o *clustering* de termos induz o *clustering* de documentos enquanto que o *clustering* de documentos induz *clustering* de termos, definindo *co-clustering* como a tarefa de agrupar simultaneamente documentos e palavras de modo que as interações entre os *clusters* sejam maximizadas. Em outras palavras, o objetivo é encontrar conjuntos de documentos e palavras que exibam uma forte coexistência ou coocorrência, otimizando a função de perda. Pode-se afirmar que o algoritmo apresentado por Dhillon (2001) se difere do *clustering* tradicional unidimensional porque, em todos os estágios, protótipos do *clustering* de linha incorporam informações de *clustering* de coluna e vice-versa.

A abordagem utiliza um grafo bipartido para representar as relações entre documentos e palavras. Nesse grafo, os nós são divididos em dois conjuntos: um conjunto representa os documentos e outro conjunto representa as palavras. As arestas do grafo bipartido conectam documentos a palavras, indicando a frequência ou a relação entre a presença de uma palavra em um documento. O particionamento espectral desse grafo bipartido é então utilizado para encontrar *clusters* coesos de documentos e palavras. O particionamento espectral envolve a análise dos autovetores associados aos menores autovalores da matriz de Laplace normalizada do grafo.

Os resultados apresentados por Dhillon (2001) mostram que o *co-clustering* é significativamente melhor do que o *clustering* unidimensional. Com a aplicação do *Information Theoretic Co-clustering* é possível notar desempenho superior em comparação ao caso em que o *clustering* de documentos é executado sem nenhum *clustering* de termos. Dhillon (2001) cita também que o motivo é que o *Information Theoretic Co-clustering* implicitamente executa uma redução de dimensionalidade adaptativa em cada iteração e estima menos parâmetros quando comparado a abordagens de *clustering* unidimensional.

4.3.2 Co-clustering baseado em modelo de aprendizagem de co-ajuste

Zhang et al. (2021) apresenta um algoritmo chamado *Co-adjustment Learning for Co-clustering* (CALCC) que pode ser usado simultaneamente em situações de aprendizagem não supervisionada, semi-supervisionada e supervisionada. CALCC implementa um modelo aprendizagem de co-ajuste (CAL) utilizado para extrair representações significativas tanto no espaço dos objetos quanto no espaço de características para produzir *co-clusters*. O desenvolvimento do CALCC foi inspirado em dois aspectos: utilização de pequenos conjuntos de objetos como informação pode ser eficaz quando incorporado no processo de aprendizagem do modelo CP e; o uso simultâneo de pequenos conjuntos de objetos e pequenos conjuntos de características pode ser útil para obter melhor entendimento dos dados.

4.3.3 Co-clustering baseado em método espectral

Huang et al. (2020) apresenta um algoritmo de *co-clustering* baseado em grafos bipartidos que utiliza a dualidade entre objetos e características de dados. O algoritmo constrói um grafo bipartido de modo que a estrutura simultânea de dados possa ser extraída. Huang et al. (2020) cita que a questão principal da utilização de grafos bipartidos para *co-clustering* é integrar os grafos bipartidos para obter um consenso ótimo. Assim é possível aprender um peso ótimo para cada grafo bipartido automaticamente, sem introduzir nenhum parâmetro adicional.

4.3.4 Co-clustering baseado em fatorização de matriz

Salah, Ailem e Nadif (2018) cita que um aspecto importante ao lidar com dados de texto é capturar as relações semânticas entre os termos, uma vez que documentos que tratam do mesmo assunto podem não necessariamente usar exatamente o mesmo vocabulário. Para confirmar a hipótese de que termos que coocorrem com frequência no mesmo contexto, por exemplo, um documento ou frase, provavelmente apresentam apresentam significados semelhantes, Salah, Ailem e Nadif (2018) propõe um modelo baseado na tri-fatorização de matriz não negativa (NMTF) que mapeia termos com co-ocorrência frequente visando encontrar relações entre eles.

4.3.5 Co-clustering baseado em modularidade

Liu, Chen e Chao (2018) apresenta um algoritmo *co-clustering fuzzy* bloco-diagonal (chamado MMFCC) que aplica maximização da modularidade. MMFCC utiliza a medida de modularidade em sua função objetivo como critério para fazer o *co-clustering* de dados representados em matrizes.

Liu, Chen e Chao (2018) cita que MMFCC oferece algumas vantagens, como a produção direta de uma matriz bloco-diagonal e uma descrição interpretável dos *co-clusters* resultantes, determinando automaticamente o número apropriado de *co-clusters*. A técnica *co-clustering fuzzy* também aumenta a precisão do resultado final do *clustering*, introduzindo os graus de pertinência *fuzzy*.

4.3.6 Métodos que aplicam co-clustering fuzzy

Segundo Kummamuru, Dhawale e Krishnapuram (2003), algoritmos de *clustering fuzzy* unidimensionais costumam apresentar alguns problemas quando os dados são esparsos, podendo ser solucionados com a utilização do *co-clustering*. Dessa forma, Kummamuru, Dhawale e Krishnapuram (2003) apresenta uma modificação do algoritmo *Fuzzy Clustering for Categorical Multivariate Data* (FCCM) (OH; HONDA; ICHIHASHI, 2001),

chamado de *Fuzzy Co-Clustering of Documents and Keyword* (Fuzzy-CoDoK) visando o *clustering* simultâneo de documentos e palavras-chave em grandes *corpora* de texto.

Para avaliar a qualidade do algoritmo Fuzzy-CoDoK, Kummamuru, Dhawale e Krishnapuram (2003) utilizam 3 conjuntos de dados: 20Newsgroups, Yahoo K1 e Classic3. Os dados do 20Newsgroups foram divididos em 3 amostras aleatórias de três subconjuntos de dados, compostos da seguinte forma: Binary1, Binary2 e Binary3, com tamanhos dos vocabulários de 4024, 4058 e 4271 termos, respectivamente; Multi51, Multi52 e Multi53, com tamanhos de vocabulários de 3249, 3215 e 3256, respectivamente; e Multi101, Multi102 e Multi103, com tamanhos de vocabulários de 2804, 2828 e 2903, respectivamente. O conjunto Yahoo K1 contém 2340 documentos divididos em 6 categorias, da seguinte forma: 494 sobre saúde; 1389 sobre lazer; 141 sobre esportes; 114 sobre política; 60 sobre tecnologia e; 142 sobre negócios, totalizando 12015 termos. O conjunto de dados Classic3 contém 1400 documentos sobre sistemas aeroespaciais, 1033 documentos sobre medicina e 1460 documentos sobre recuperação de informações.

Antes da avaliação, os dados são submetidos por duas etapas principais de pré-processamento. Na primeira etapa, um conjunto de termos (chamado de vocabulário do documento) é extraído e cada documento é representado de acordo com seu vocabulário particular. Nesta primeira etapa, para encontrar o vocabulário de cada documento são realizadas as tarefas de eliminação de *stopwords*, aplicação da lematização e, por fim, eliminação das termos que ocorrem em menos de 3 documentos. Na segunda etapa, os documentos são representados pelo vetor de frequência no qual cada dimensão reflete o número de ocorrências de um termo no documento. Em avaliações, o algoritmo Fuzzy CoDoK foi comparado com outros 4 algoritmos: FCCM, *Fuzzy Simultaneous Keyword Identification and Clustering* (FSKWIC), *Spherical K-means* (SKM) e *Spherical Fuzzy C-Means* (SFCM).

Nas configurações para os experimentos, para todos os outros algoritmos foram atribuídos graus de pertinência dos objetos aos *clusters* iniciais aleatoriamente, exceto para FSKWIC, em que os *clusters* iniciais foram obtidos a partir do *Fuzzy* C-Means. Como critérios de parada, para FCCM e *Fuzzy* CoDoK, o algoritmo termina se a alteração máxima em uc for menor que o erro (e) ou se o algoritmo atingir 50 iterações. Para os documentos foram atribuídos o rótulo do *cluster* com o número máximo de membros, valores de Tu, T e o erro (e), no *Fuzzy CoDoK*, foram ajustados para 0,00001, 1,5 e 0,00001, respectivamente. No FCCM, foi definido $Tu = 0{,}02$, $Tv = 1{,}8$ e e = 0,00001 e, no FCM, o fator de fuzzificação m como 1,015. Os resultados mostram que o *Fuzzy* CoDoK apresenta um desempenho melhor que o algoritmo FSKWIC e que o algoritmo FCCM. *Fuzzy* CoDoK funciona melhor que os algoritmos SFCM e que SKM no conjunto de dados Multi10. Segundo Kummamuru, Dhawale e Krishnapuram (2003), esse desempenho se deve quando as bases de dados têm *clusters* sobrepostos.

É possível observar que uma das principais vantagens do algoritmo *Fuzzy CoDoK* é a

possibilidade de escalabilidade e *clustering* de documentos com grande dimensionalidade de características, resolvendo um dos problemas apresentados no algoritmo FCCM. Outra vantagem é que os resultados são relevantes. Apesar de alguns resultados inferiores aos obtidos pelos algoritmos de comparação, na maioria dos casos, *Fuzzy CoDoK* apresenta um melhor desempenho, principalmente quando existem *clusters* sobrepostos.

Segundo Yan, Chen e Tjhi (2013), uma das partes principais para o avanço da tecnologia no que diz respeito ao aprendizado ou descoberta de conhecimento é o desenvolvimento de estratégias que visam a análise, compreensão e organização de grandes coleções de dados em forma de documentos de texto. A mineração de dados que sejam relevantes para o usuário visa encontrar padrões desejados. A tarefa de *clustering* é uma das principais estratégias e mais utilizadas atualmente para encontrar automaticamente *clusters* naturais nos dados.

Nos últimos anos, em aplicações de vários domínios, como *clustering* de documentos de texto, construção de taxonomias, aplicações em dados de mapeamento genético, entre outras, abordagens de *co-clustering* têm sido apresentadas motivadas pela dualidade entre objetos e características. Quando o problema se trata do *clustering* de documentos, segundo Yan, Chen e Tjhi (2013), o *clustering* de características semelhantes é tão importante quanto o *clustering* de documentos semelhantes.

Partindo dessa premissa, a motivação para Yan, Chen e Tjhi (2013) é incorporar conhecimento prévio ao processo de *co-clustering* visando facilitar/direcionar o aprendizado de dados com o objetivo de melhorar o desempenho da categorização dos objetos. É proposto um método chamado de *Semi-Supervised Technique in the Fuzzy Co-Clustering Framework* (ou SS-FCC). No método é incorporado conhecimento de domínio na forma de restrições através da adição de um pequeno número de objetos previamente rotulados ao conjunto de dados. O objetivo da adição do conhecimento é orientar o *clustering* com o propósito de aumentar a precisão e reduzir a sensibilidade dos parâmetros de fuzzificação, visando atingir melhores resultados.

As restrições utilizadas por Yan, Chen e Tjhi (2013) são do tipo *"must-link"* (ou "deve ligar") e *"cannot-link"* (ou "não deve ligar"). Os dados são dispostos no modelo espaço vetorial, sendo que o conjunto de dados consiste em N objetos representados por M características, respectivamente, representado cada objeto. Portanto, o conjunto de dados pode ser representado usando uma matriz $N \times M$ onde cada linha representa um dos objetos de dados da mesma forma que cada coluna representa uma característica. O objetivo do *clustering* é particionar esses N objetos em C *clusters* significativos.

Em experimentos, o desempenho do algoritmo SS-FCC foi testado em 14 conjuntos de dados de *benchmark* do 20Newsgroups. Em 10 destes conjuntos o número de *clusters* varia entre 2 e 20, o número de objetos varia entre 500 e 4199 e a dimensionalidade da representação destes documentos no modelo espaço vetorial varia de cerca de 2000 a cerca de 10000 dimensões. Os dados foram pré-processados da seguinte maneira: termos que

ocorrem menos de 0,5% ou mais de 99,5% em relação ao número de documentos foram removidos, remoção de *stopwords* e lematização. Como medidas de avaliação foram adotadas F-Score e Acurácia. As configurações do algoritmo SS-FCC são as seguintes: as inicializações são aleatórias em todos os testes e os resultados representam uma média de 20 simulações de teste. O limiar de parada foi definido em $\in = 10^{-5}$, assim como o limite máximo de iterações foi definido em 200. Os resultados do algoritmo SS-FCC foram comparados com o algoritmo *Fuzzy* CoDoK (KUMMAMURU; DHAWALE; KRISHNAPURAM, 2003). Os experimentos foram feitos comparando *Fuzzy* CoDoK sem objetos rotulados e SS-FCC com um *cluster* de restrições de pares construídos em determinada porcentagem de objetos: com 5%, 10%, 15% e 20%. Em todos os conjuntos de dados utilizados, em todos os experimentos, SS-FCC supera o *Fuzzy* CoDoK, além de que o SS-FCC é capaz de fazer melhorias significativas aprendendo mais rapidamente utilizando restrições. Além disso, quando a porcentagem dos objetos rotulados aumenta continuamente, não apenas o desempenho do *clustering* aumenta considerando as medidas de F-Score e Acuráci, mas também a estabilidade dos resultados.

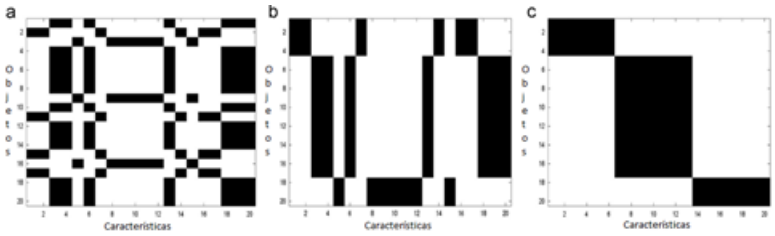
Levando em consideração os benefícios e o desempenho obtido, pode-se observar o que o algoritmo SS-FCC proposto em Yan, Chen e Tjhi (2013) atinge bons resultados combinando a estratégia de *co-clustering* com restrições que guiam o algoritmo no sentido de agrupar objetos considerando conhecimento prévio.

Laclau e Nadif (2016) afirmam que a estratégia de *co-clustering* apresenta vantagens em relação a estratégia de *clustering* unidimensional, como menor esforço computacional e representação simplificada da matriz de dados.

Com os benefícios apresentados pela aplicação de tal estratégia, Laclau e Nadif (2016) propõe dois algoritmos de *co-clustering* inspirados no algoritmo *Double K-means*, o algoritmo *Hard Diagonal Double K-means* (DDKM) e o algoritmo *Fuzzy Diagonal Double K-means* (F-DDKM), que visam o *co-clustering* de termos e documentos simultaneamente. Ambos algoritmos visam o *co-clustering* diagonal com o número de *clusters* de linhas (objetos) igual ao número de *clusters* de colunas (características). Em ambos algoritmos, os dados originais (Figura 28a) são reorganizados gerando *co-clusters* de linhas e colunas diagonalmente (Figura 28c) baseados na minimização de medidas de heterogeneidade dos *clusters* e na variância intra-cluster.

Laclau e Nadif (2016) afirmam que ambos os algoritmos DDKM e F-DDKM são eficientes em relação a qualidade dos *co-clusters* resultantes, além da capacidade de lidar com conjuntos de dados esparsos e de alta dimensionalidade.

Figura 28 – Reorganização de dados de acordo com linhas e colunas



Fonte: Laclau e Nadif (2016)

Capítulo 5

Abordagem para Co-clustering de Documentos com Avaliação de co-clusters

Neste capítulo é apresentada a abordagem para organização de documentos proposta e avaliada nesta tese. Este capítulo está organizado da seguinte forma: Na seção 5.1 é introduzida uma visão geral da abordagem, os objetivos e as principais contribuições dessa proposta; Na seção 5.2 são descritas as etapas do processo de organização de documentos proposto, desde o tratamento das coleções de textos, a construção das matrizes documentos × clusters_de_termos, aplicação de algoritmos de co-clustering, avaliação dos clusters de documentos, avaliação dos clusters de termos e avaliação de associação entre clusters de termos e clusters de documentos; Na seção 5.3 são apresentados os recursos, técnicas, ferramentas e algoritmos utilizados neste projeto, com o objetivo de contextualizar o leitor sobre o porquê de tais escolhas.

5.1 Descrição geral da proposta

Os métodos de *co-clustering* têm se mostrado adequados para agrupar documentos representados no formato de matriz em que cada atributo é um termo que ocorre no documento, uma vez que, por encontrarem *clusters* de documentos e *clusters* de termos simultaneamente, obtém-se como resultado informações sobre quais termos estão associados a cada *cluster* de documentos. Isso vai de encontro a uma tarefa que permanece no foco das pesquisas da mineração de textos, que é a descoberta de tópicos que caracterizam o conteúdo do documento. Porém, as representações estruturadas de documentos na

forma de vetorização de contagem, seja por Bag-of-words ou TF-IDF, são esparsas e de alta dimensionalidade. A representação de termos como vetores numéricos, conhecidos como word embeddings, ampliou as linhas de pesquisa sobre representação de documentos possibilitando sua utilização de forma mais efetiva por métodos de aprendizado de máquina.

Nesse sentido, neste trabalho é utilizada a representação do vocabulário de uma coleção de documentos na forma de word embeddings para encontrar *clusters* de termos e reduzir a dimensionalidade e esparsidade da matriz documentos × termos, ou seja, aplicar um método prévio de *clustering* de termos e utilizar o resultado para aplicar *co-clustering* na matriz documento × cluster_de_termos. Para essa finalidade, foi usado o algoritmo K-means e os termos foram agrupados em diferentes números de *clusters*. O *clustering* prévio de termos leva à necessidade de mudanças no cálculo da medida TF-IDF para criar representações vetoriais por contagem dos documentos da coleção. Neste trabalho é feita também uma proposta simples de cálculo da medida TF-IDF para *clusters* de termos, usando a soma do TF-IDF de termos individuais. Com essas transformações, cada coluna da matriz TF-IDF representa agora um *cluster* de termos e não mais cada termo individualmente. As matrizes geradas por essa forma são então apresentadas como entrada para algoritmos de *co-clustering* e, após a execução dos mesmos, os *clusters* obtidos são avaliados.

Conforme observado por Role, Morbieu e Nadif (2018), a maioria dos trabalhos avaliam o desempenho dos algoritmos de *co-clustering* apenas considerando os *clusters* de documentos, como é feito em um contexto de *clustering* unidimensional. Essa abordagem não é suficiente, uma vez que o *co-clustering* visa encontrar associações entre *clusters* de documentos e *clusters* de termos. Não existem trabalhos na literatura disponível que busquem solucionar este problema, ou seja, não existem maneiras suficientes para avaliar, de forma completa, algoritmos de *co-clustering*.

Buscando uma forma de apresentar avaliações mais completas e informativas e buscando solucionar tal deficiência observada, com o objetivo de explorar o resultado fornecido por algoritmos de *co-clustering*, foram conduzidos três tipos de avaliações: avaliações de *clusters* de documentos separadamente, avaliações de *clusters* de termos separadamente e avaliações das associações entre os *clusters* de documentos e *clusters* de termos. Para o primeiro tipo de avaliação foram utilizadas métricas de avaliação externa de *clustering* comumente utilizadas na literatura. Para o segundo tipo de avaliação, além de duas métricas internas de avaliação de *clustering* consolidadas na literatura, foi também utilizada uma adaptação da métrica sugerida por Role, Morbieu e Nadif (2018), proposta original deste trabalho. Para o terceiro tipo de avaliação foram utilizadas duas métricas originais propostas neste trabalho, também como adaptações de métricas encontradas na literatura (ROLE; MORBIEU; NADIF, 2018; SELOSSE; JACQUES; BIERNACKI, 2020).

De maneira resumida, o esforço apresentado neste trabalho se concentra em: dadas

coleções de documentos de texto, o método proposto busca agrupar documentos e *clusters* de termos utilizando algoritmos de *co-clustering*, avaliá-los e analisá-los utilizando métricas inovadoras e originais.

Inicialmente, as coleções de documentos de texto são pré-processadas e utilizadas para produzir vetores de embeddings, os quais são utilizados para construir matrizes termo × embeddings correspondentes. Tais vetores de embeddings são agrupados utilizando K-means.

Tal sequência de processos produz *clusters* de termos. Os termos mais relevantes são avaliados e submetidos para a construção de matrizes documento × cluster_de_termos. Por fim, as matrizes documento × cluster_de_termos são submetidas aos algoritmos de *co-clustering* utilizados nesta proposta, produzindo os *co-clusters* resultantes. Os *co-clusters* resultantes são submetidos às métricas de avaliação propostas e posteriores análises.

5.2 Descrição das etapas da abordagem proposta

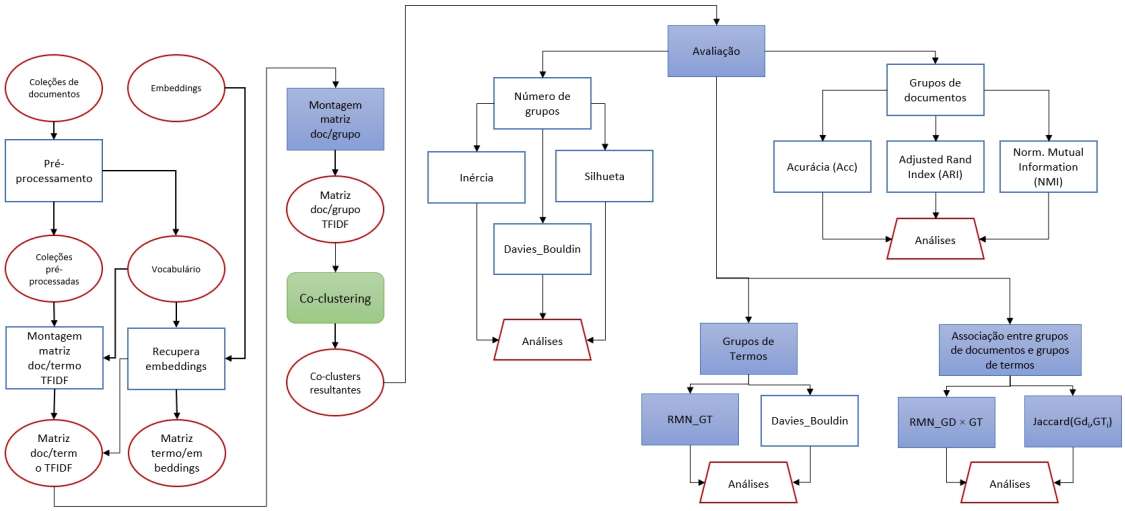
Nesta seção são descritas as etapas que compõem a abordagem proposta para organizar documentos e termos usando *co-clustering*. As etapas descritas nesta seção, que podem ser acompanhadas na Figura 29, são apresentadas com o objetivo de orientar o leitor sobre o fluxo de informações pelo sistema, considerando passos apresentados neste capítulo, que incluem o pré-processamento da coleção de documentos, a recuperação dos word embeddings dos termos que compõem o vocabulário da coleção e o *clustering* dos termos desse vocabulário, a construção da matriz documentos × clusters_de_termos com o cálculo da medida TF-IDF modificada, o *co-clustering* das coleções de documentos com base na matriz calculada na etapa anterior e as avaliações dos resultados obtidos usando métricas já conhecidas e as propostas neste trabalho.

5.2.1 Conjuntos de dados

Para este projeto, a escolha dos conjuntos de dados utilizados nos experimentos adotou o critério de diversidade, ou seja, buscando compreender diferentes aspectos, como, por exemplo, a variação no número de documentos, variação no número de classes, variação número de termos, variação na quantidade de termos por documento e variação no aspecto formal dos textos, com variações de textos populares, como *reviews*, e textos formais, provenientes de notícias ou artigos científicos. Visando abranger todos estes aspectos, foram escolhidos seis conjuntos de dados. Tais conjuntos são:

– Classic3: É um conjunto com 3891 documentos de texto usados para *benchmarking* em muitas pesquisas relacionadas à mineração de texto. É composto pelas subcoleções Medline (MED), Cisi (CISI) e Cranfield (CRAN). Medline consiste em 1033 resumos

Figura 29 – Fluxo de informações e tarefas



Fonte: Produzido pelo autor.

de revistas médicas, Cisi consiste em 1460 resumos de artigos sobre o tema recuperação de informação e Cranfield consiste em 1388 resumos de sistemas aerodinâmicos. O conjunto apresenta esparsidade de 99% (TUNALI, 2010).

– Classic4: O conjunto de dados Classic4 é composto 7095 resumos de artigos científicos separados por quatro coleções de documentos diferentes: CACM (títulos e resumos da revista Communications of the ACM), CISI (informações documentos de recuperação), CRANFIELD (documentos do sistema aeronáutico) e MEDLINE (periódicos médicos) (TUNALI, 2010). O conjunto de dados Classic3 é um subconjunto do conjunto Classic4.

– CSTR: O conjunto de dados CSTR (*Computer Science Technical Reports*) é composto por 300 documentos pertencentes a 4 áreas da ciência da computação: PLN, Robótica/Visão, Sistemas e Teoria. O conjunto é formado por resumos e relatórios técnicos publicados no Departamento de Ciência da Computação da Universidade de Rochester, no período de 1991 a 2007.

– Reuters8: Neste projeto foi utilizado um subconjunto da coleção Reuters-21578 (LEWIS et al., 2004) formado por documentos da agência de notícias Reuters, em 1987. Em 1990, os documentos foram disponibilizados pela Reuters para fins de pesquisa para o Laboratório de Recuperação de Informação do Departamento de Ciência da Informação e Computação da Universidade de Massachusetts, em Amherst. O conjunto R8 é dividido em oito categorias: *grain*, *ship*, *interest*, *money*, *trade*, *crude*, *acq* e *earn*.

– Sports: o conjunto de dados Sports (HAJJ; RIZK; AWAD, 2019) é formado por notícias esportivas divididas entre os assuntos beisebol, basquete, ciclismo, boxe, futebol, golfe e hóquei, entre outros, divididos em duas categorias; objetividade e subjetividade.

– Newsgroup5 (NG5): Neste projeto foi utilizado um subconjunto do 20Newsgroups (JOACHIMS, 1996), o conjunto NG5, contendo 4702 documentos, divididos em 5 categorias. O conjunto de artigos 20Newsgroups é um conjunto de dados de 20 grupos de notícias com 20000 documentos aproximadamente. A coleção é popular em aplicações de aprendizado de máquina, como classificação e *clustering* de texto.

Segundo Rossi et al. (2013), as coleções CSTR e Reuters8 são bem separadas, porém desbalanceadas, ou seja, o número de documentos de uma classe é muito maior que o número de documentos de outra classe. Segundo Laclau e Nadif (2016), CSTR, Classic3 e Classic4 são coleções de documentos sem *clusters* sobrepostos. Já em Newsgroup5, os *clusters* de documentos não são bem separados. Hajj, Rizk e Awad (2019) cita que coleção de Sports é desbalanceada e composta por artigos de várias fontes e autores e, portanto, apresenta dificuldades para a tarefa de *clustering*.

5.2.2 Pré-processamento dos conjuntos de dados

As 6 coleções de documentos de texto utilizadas foram previamente formatadas e cada documento alinhado com seu rótulo correspondente, tendo em vista padronizar o formato de todas as coleções para o processo de pré-processamento.

A partir deste ponto, para cada coleção de documentos de texto, foi aplicado(a):

– Remoção de números, símbolos, caracteres especiais e pontuação;

– Padronização da altura para caixa baixa; e

– Remoção de *stopwords*.

Vale mencionar que não foi aplicado o processo de lematização nas coleções de documentos. Segundo Moral et al. (2014), quando o objetivo é agrupar termos (passo descrito na Seção 5.2.3) lematizar os termos pode prejudicar o desempenho do *clustering*, já que manter as flexões de termos favorecem o desempenho do algoritmo de *clustering*.

A saída do pré-processamento é um conjunto para cada coleção de documentos contendo um documento por linha. As informações de cada coleção de documentos utilizados neste projeto, após pré-processamento, estão descritas na Tabela 17.

5.2.3 Recuperação de embeddings e clustering de termos

A representação de termos por vetores numéricos (word embeddings), além de prover informação mais rica sobre cada termo, por incorporar significado do termo no contexto da coleção de documentos, abre possibilidades para o uso de sua representação vetorial de formas que, utilizando o termo como uma informação atômica, não seriam possíveis. Neste trabalho incluímos o passo de *clustering* de termos do vocabulário de cada coleção antes de aplicar o algoritmo de *co-clustering*.

Tabela 17 – Dimensões das coleções de documentos após pré-processamento.

Coleção	**Nº de documentos**	**Nº de termos**	**Nº de classes**
Classic3	3891	4620	3
Classic4	7092	4685	4
CSTR	299	4971	4
NG5	4702	1892	5
Reuters8	2189	9062	8
Sports	1000	21138	2

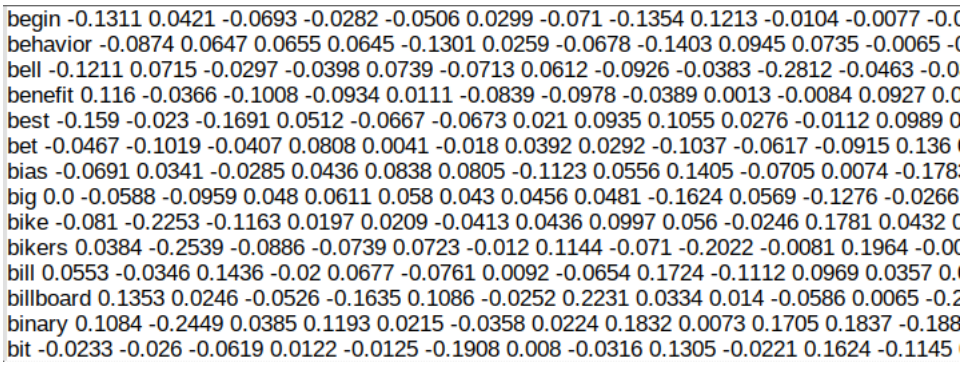
Fonte: Produzido pelo autor.

A extração do vocabulário de uma coleção de documentos consiste em organizar todos os termos que ocorrem no documento. Nesta etapa do projeto, o vocabulário extraído de cada coleção de documentos é utilizado para recuperar seus vetores de word embeddings correspondentes.

Cada termo presente no vocabulário é buscado na coleção de word embeddings. Para este trabalho foi utilizada uma coleção do fastText[1]. Quando encontrado, o vetor é extraído e adicionado ao vocabulário de embeddings. Ao final do processo cada termo presente no vocabulário está associado ao vetor correspondente.

Um exemplo do vocabulário de word embeddings é ilustrado na Figura 30, extraído da coleção Classic3:

Figura 30 – Exemplo de vocabulário de vetores de word embeddings



Fonte: Produzido pelo autor.

O *clustering* de termos consiste em agrupar os termos semelhantes de uma coleção de documentos de texto com base em word embeddings, ou seja, agrupar termos considerando representações vetoriais semelhantes.

[1] FastText é uma biblioteca gratuita e de código aberto voltada para o aprendizado de representações e classificadores de texto. Disponível em: https://fasttext.cc/

Este processo produz *clusters* de termos semelhantes que são utilizados para a construção dos dados de entrada dos algoritmos de *co-clustering*. Como mencionado, o *co-clustering* produz *co-clusters*, ou seja, considerando a aplicação neste projeto, *clusters* de documentos relacionados a *clusters* de termos. Os objetivos de agrupar termos como uma etapa prévia à aplicação do *co-clustering* são:

– Analisar o impacto que o *clustering* prévio de termos causa no desempenho de algoritmos de *co-clustering*;

– Reduzir a dimensionalidade dos dados de entrada dos algoritmos de *co-clustering*;

– Aumentar a assertividade dos algoritmos de *co-clustering*; e

– Extrair tópicos mais representativos de cada *co-cluster* com base nos termos mais relevantes de cada *cluster*.

O vocabulário de vetores de word embeddings mencionado anteriormente é utilizado como conjunto de dados de entrada para o *clustering* de termos. Desse modo, o *clustering* de termos é realizado com base no vetor de embeddings que representa cada termo, através da distância entre pares de vetores.

O método de *clustering* utilizado para essa finalidade neste trabalho, o algoritmo K-means, recebe como parâmetro o número de *clusters*. Com base em experimentos prévios, optamos por agrupar os termos usando 5 valores distintos: número ideal de *clusters*, valor resultante da aplicação do método do cotovelo (*elbow method*) (THORNDIKE, 1953), 5%, 10%, 25% e 50% do total de termos.

Por exemplo, a coleção de documentos R8, após o pré-processamento, é formada por 9062 termos. A aplicação do método do cotovelo sugeriu que o número ideal de *clusters* para a coleção de dados é 23 (em torno de 0,25%). Para os outros casos, 5%, 453 *clusters*, 10%, 906 *clusters*, 25%, 2265 *clusters* e para 50%, 4531 *clusters*.

Na Tabela 18 são mostrados 10 *clusters* formados apenas por ilustração do processo, extraídos da coleção Classic3, que tem um total de 4620 termos. A Tabela 18 representa o *clustering* de termos para 462 *clusters* formados após este processo.

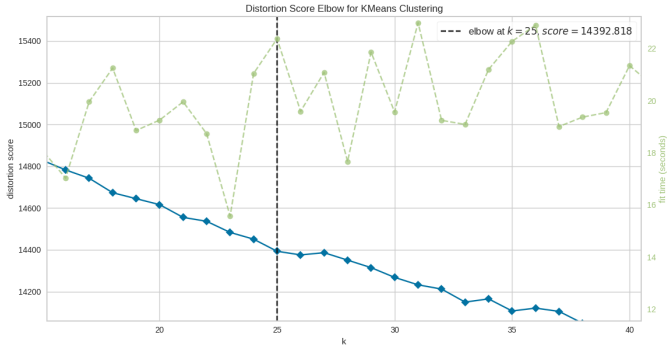
Nas Figuras 31, 32, 33, 34, 35 e 36 é mostrada a avaliação do método do cotovelo para as coleções de documentos utilizadas neste trabalho. O método do cotovelo é uma técnica utilizada para determinar o número ideal de *clusters* em um conjunto de dados. O objetivo do método do cotovelo é avaliar o desempenho do algoritmo de *clustering* para diferentes valores de k *clusters* e observar como a variação da medida de validação interna se comporta à medida que k aumenta. Nos gráficos, a linha verde designa a quantidade de tempo para treinar o modelo de *cluster* k, a linha azul representa a variação da medida de validação interna (soma dos quadrados intra-*cluster* - WCSS) para diferentes valores de k, e a linha vertical preta tracejada mostra o ponto onde a linha azul mostra um "cotovelo", que corresponde ao ponto de melhor inflexão, ou seja, ao valor ideal de k.

Tabela 18 – Dimensões das coleções de documentos após pré-processamento.

Clusters	**Termos**
Cluster_7	allied attributable closely related similar unrelated
Cluster_8	arginine cysteine histidine
Cluster_9	pericardial
Cluster_10	baby infant neonate newborn
Cluster_11	mitochondrion organelle
Cluster_12	check checked confirm log verify
Cluster_13	glycolytic isozyme isozymes palmitate pyruvate sphingomyelin sphingosine thymidine
Cluster_14	amplitude attenuation excitation harmonic modulation oscillattion oscillatory
Cluster_15	career
Cluster_16	accordance carefully correctly fit ill perfectly poorly properly reasonably satisfactorily

Fonte: Produzido pelo autor.

Figura 31 – Classic3 - Gráfico do método do cotovelo



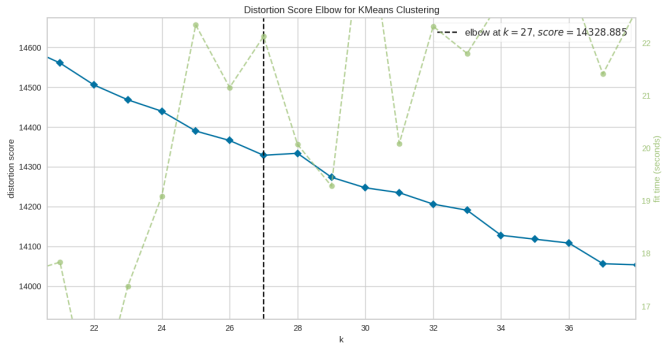
Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

5.2.3.1 Avaliação do clustering de termos

O *clustering* de termos foi avaliado como forma de subsidiar as escolhas dos números de *clusters* e também para confrontar a qualidade com as avaliações dos *co-clusters* obtidos ao final do processo. As medidas de avaliação utilizadas são todas de avaliação interna de *clustering*, uma vez que não há informação disponível quanto aos *clusters* corretos a que cada termo deve pertencer. Essas medidas são:

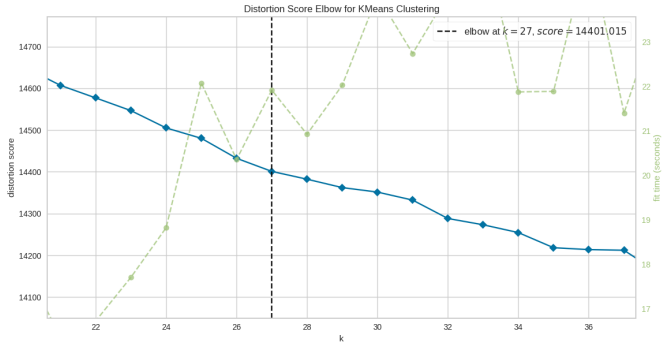
Silhueta: *Silhouette* (ROUSSEEUW, 1987) é uma medida interna de avaliação utilizada para calcular a eficácia das estratégias de *clustering*, auxiliando na interpretação e validação de consistência interna de *clusters* de objetos, ou seja, medindo quão semelhante um objeto é considerando seu próprio *cluster* (coesão) em comparação com outros *clusters* (separação). Usa compactação de *clusters* individuais (distância *intra-cluster*) e

Figura 32 – Classic4 - Gráfico do método do cotovelo



Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

Figura 33 – CSTR - Gráfico do método do cotovelo



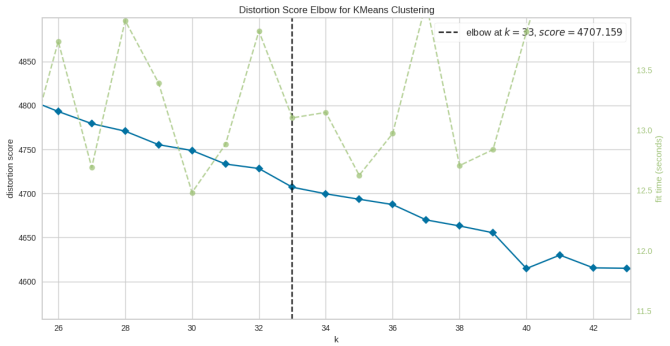
Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

separação entre *clusters* (distância *inter-cluster*) para avaliar de forma geral a qualidade do algoritmo de *clustering*. Quanto aos valores, a silhueta varia de -1 a +1, sendo que quanto mais alto o valor maior a correlação entre o objeto com seu próprio *cluster* e menor correlação com os outros *clusters*. Valores próximos de 0 indicam que os *clusters* estão sobrepostos. Tal correlação pode ser calculada utilizando uma medida de distância, como distância Euclidiana, por exemplo. A silhueta para um determinado ponto $\boldsymbol{i}$ é definida conforme Equação 19:

$$s_i = \frac{b_i - a_i}{max(b_i, a_i)} \tag{19}$$

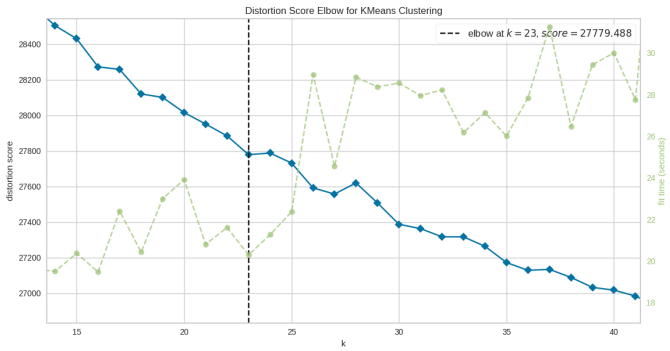
Na Equação 19, b_i representa a menor distância média do ponto i até outro *cluster* do qual i não faça parte, chamado de *cluster* vizinho de i, e a_i representa distância média do ponto i a todos os outros pontos do *cluster* do qual faz parte. b_i e a_i são calculados conforme a Equação 20 e Equação 21, respectivamente.

Figura 34 – Newsgroup5 - Gráfico do método do cotovelo



Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

Figura 35 – Reuters8 - Gráfico do método do cotovelo



Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

$$b_i = \min_{k \neq i} \frac{1}{|C_k|} \sum_{j \in C_k} dis(i,j) \tag{20}$$

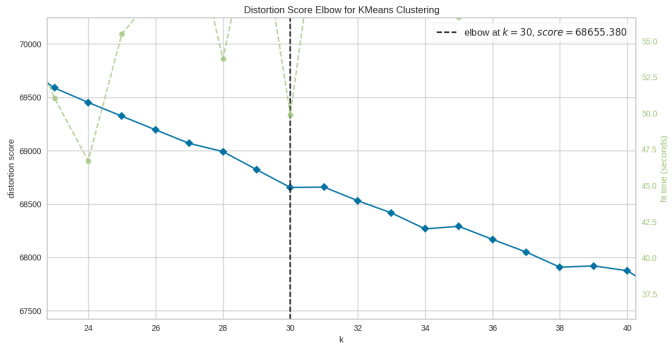
$$a_i = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} dis(i,j) \tag{21}$$

Silhueta para cada *cluster*: Silhueta do *cluster* (S_k) é a média dos índices de silhueta de todos os pontos no *cluster* C_k, como representado na Equação 22.

$$S_k = \frac{1}{|C_k|} \sum_{i \in C_k} \mathbf{s}_i \tag{22}$$

Silhueta para o *clustering* todo: A silhueta média do *clustering* (S) é a média dos índices de silhueta de todos os pontos em todos os *clusters*, como representado na Equação 23.

Figura 36 – Sports - Gráfico do método do cotovelo



Fonte: Produzido usando YellowBrick (https://www.scikit-yb.org/en/latest/).

$$S = \frac{1}{N}\sum_{i=1}^{N}\mathbf{s}_i \tag{23}$$

Onde:

– N é o número total de pontos.

– $dis(i, j)$ é a distância entre os pontos i e j.

– $|C_k|$ é o número de pontos no *cluster* C_k.

Inércia: A inércia é a própria função objetivo otimizada pelo algoritmo K-means, que pode ser utilizada como uma medida interna de avaliação para medir o quão bem um conjunto de dados foi agrupado pelo algoritmo. É calculada pela aplicação de Within-Cluster-Sum-of-Squares (WCSS). Dado um conjunto de entradas $N = (x_1, x_2, ..., x_n)$, onde cada entrada é um vetor real de dimensão d, o K-means visa particionar as N entradas em $k(\leq n)$ conjuntos $C = \{C_1, C_2, ..., Ck\}$, de modo a minimizar a soma dos quadrados das distâncias (WCSS), que é definida como na Equação 24, onde υ_i são as coordenadas do ponto i em relação ao centróide do *cluster* C_k (KRZANOWSKI; LAI, 1988).

$$\underset{S}{WCSS} = \sum_{i=1}^{k}\sum_{x\in S_i}\|x - \upsilon_i\|^2 \tag{24}$$

Os valores resultantes da aplicação da Inércia iniciam e 0 e não tem limite superior. É desejável obter o valor mais baixo possível.

Davies Bouldin: O índice Davies-Bouldin (DB) (DAVIES; BOULDIN, 1979), é uma métrica que valida a qualidade do *clustering* considerando as características do conjunto de dados. Quanto menor o valor do índice, melhor a avaliação do *clustering*. O índice de Davies-Bouldin é definido na Equação 25, onde $\delta(C_i, C_j)$ é a distância *inter-cluster* entre o *cluster* C_i e o *cluster* C_j e $\Delta(C_k)$ é a distância *intra-cluster* do *cluster* C_k.

$$DB\ index(U) = \frac{1}{k}\sum_{i=1}^{k} \underset{i \neq j}{max} \left\{ \frac{\Delta(Ci) + \Delta(Cj)}{\delta(Ci, Cj)} \right\} \tag{25}$$

Onde:

– k é o número total de *clusters*.

– C_i é o i-ésimo *cluster*.

– $\Delta(Ci)$ é a dispersão média *intra-cluster* para o *cluster* C_i.

– $\delta(Ci, Cj)$ é a distância média *inter-cluster* entre os *clusters* C_i e C_j.

– U representa o conjunto de *clusters* no qual o índice está sendo calculado.

5.2.4 Geração da matriz documentos × clusters_de_termos

Os dados encontrados em *corpora* disponíveis, seja na *web* ou em conjuntos de dados criados para auxiliar o desenvolvimento de estudos e pesquisas científicas, geralmente estão disponíveis em linguagem natural, ou seja, de forma que são de fácil compreensão humana, porém de difícil compreensão ou manipulação computacional. Sendo assim, para que os dados possam ser reconhecidos por máquinas ou sistemas, é necessário que sejam processados e transformados de forma que sejam compreendidos e possam ser interpretados.

A representação textual pode ser feita de várias formas, geralmente concentrando na extração de *features* que representam um documento de texto, considerando os atributos mais relevantes àquele documento. É comum que antes deste processo seja utilizado um método para reduzir a dimensionalidade das entradas.

Em documentos de texto disponíveis em língua natural normalmente existem termos irrelevantes, ou seja, que não apresentam representatividade ao significado do conteúdo. Geralmente há uma quantidade grande de dados de entrada, mas pouca informação útil. Por isso, muitos processos de representação de dados textuais visam a redução da dimensionalidade das características de tais objetos.

O principal objetivo do *clustering* prévio de termos é construir uma matriz de entrada para o algoritmo de *co-clustering* com dimensionalidade reduzida, de forma que cada coluna da matriz represente um *cluster* de termos e não mais apenas um termo, como acontece na representação tradicional de documentos com a medida TF-IDF.

Com base nos *clusters* de termos produzidos pela etapa de *clustering* de termos descrita na Seção 5.2.2 foi gerada a matriz documentos × clusters_de_termos, entrada principal para os algoritmos de *co-clustering*. A matriz contém dimensões $d \times g$, sendo d o número de documentos do conjunto de dados e g o número de *clusters* escolhido para o *clustering* prévio de termos. A matriz documento × clusters_de_termos é construída partir dos *clusters* de termos produzidos a partir do processo descrito na seção 5.2.2 e da matriz TF-IDF tradicional.

Neste trabalho é proposta a definição do cálculo da medida TF-IDF de um *cluster* de termos em um documento pela soma dos valores TF-IDF (calculados da forma tradicional como definido na seção 3.2.2) de todos os termos do *cluster* que aparecem no documento. Com isso geramos uma matriz da forma documentos × clusters_de_termos, que chamamos de matriz TF-IDF modificada. A Equação 26 define como calcular o TF-IDF da matriz documentos × clusters_de_termos, onde d representa um documento, g representa um *cluster* de termos e t_i representa os termos que pertencem ao *cluster* g.

$$TF\text{-}IDF(d, g) = \sum_{t_i \in g} TF\text{-}IDF(d, t_i) \tag{26}$$

Por exemplo, considere um documento de texto contendo a sentença *"month blood muslim blood nt yet email voice germany fax"* e considere que para a coleção de documentos em que este documento aparece foi escolhido agrupar os termos em 33 *clusters*. Com base nos *clusters* de termos e nos valores TF-IDF de cada termo no documento de exemplo, é gerado um conjunto de triplas contendo o termo, o valor TF-IDF e o *cluster* que o termo pertence. Um exemplo deste conjunto de triplas é ilustrado na Tabela 19.

Tabela 19 – Triplas (Termo,TF-IDF,Cluster).

Termo	**Valor TF-IDF**	**Cluster**
month	0.2748691873989251	Cluster_25
blood	0.6215683058529244	Cluster_5
muslim	0.2585299509199396	Cluster_31
nt	0.1197727827067213	Cluster_2
yet	0.2445845772321716	Cluster_14
email	0.2380415913423156	Cluster_15
voice	0.3403352872060712	Cluster_15
germany	0.3351141549021327	Cluster_23
fax	0.3351141549021327	Cluster_15

Fonte: Produzido pelo autor.

A partir do conjunto de triplas, o valor que representa um *cluster* de termos em um documento é calculado através da soma do valor TF-IDF de todos os termos que fazem parte do mesmo *cluster*.

5.2.5 Aplicação dos algoritmos de co-clustering

Nesta seção são apresentados de forma breve os algoritmos utilizados no trabalho.

A escolha do algoritmo para a tarefa proposta requer algumas considerações, como características dos dados, estrutura de *clusters*, possibilidade de *clusters* em sobreposição e métricas de avaliação. Quatro algoritmos, considerando as necessidades apresentadas na proposta, foram escolhidos para implementação, teste e avaliação. A escolha dos algoritmos foi baseada nos seguintes critérios:

– Algoritmos de *co-clustering* utilizados em dados textuais capazes de lidar com alta dimensionalidade de dados e esparsidade da matriz de dados;

– Algoritmos que apresentem a capacidade de produzir resultados com estruturas variadas, considerando *co-clusters* com: estrutura tabuleiro de xadrez, estrutura diagonal, sobreposição de linhas e/ou colunas e considerando uma matriz de dados esparsa;

– Algoritmos com diferentes estratégias, como: algoritmos com estrutura bloco-diagonal, algoritmos que implementam métodos espectrais, algoritmos baseados na teoria da informação e algoritmos *fuzzy*.

Para Govaert e Nadif (2013), outra forma utilizada geralmente para caracterizar algoritmos de *co-clustering* é através da partição produzida, ou seja, se o algoritmo produz partições gerais (na forma do tabuleiro de xadrez) ou se o algoritmo particiona a matriz de dados em uma estrutura diagonal (também chamado de algoritmo bloco-diagonal).

No caso de partições na forma do tabuleiro de xadrez, o número de *clusters* de linhas pode ser diferente do número de *clusters* de coluna, o que torna o *clustering* restritivo. Já no algoritmo de bloco-diagonal, o número de *clusters* de linhas e colunas precisa ser igual, já que cada *cluster* de linhas está associado a apenas um *cluster* de colunas, e vice-versa (GOVAERT; NADIF, 2013).

Quando os métodos são aplicados para mineração de texto, ambas abordagens, tabuleiro de xadrez e bloco-diagonal, são interessantes. A abordagem bloco-diagonal é uma abordagem teoricamente mais simples, já que cada conjunto de documentos é automaticamente rotulado por apenas um conjunto de termos. Já a abordagem tabuleiro de xadrez é utilizada quando é necessário associar um conjunto de documentos a vários conjuntos de termos, caso em que o método de *co-clustering* bloco-diagonal não se aplica (GOVAERT; NADIF, 2013).

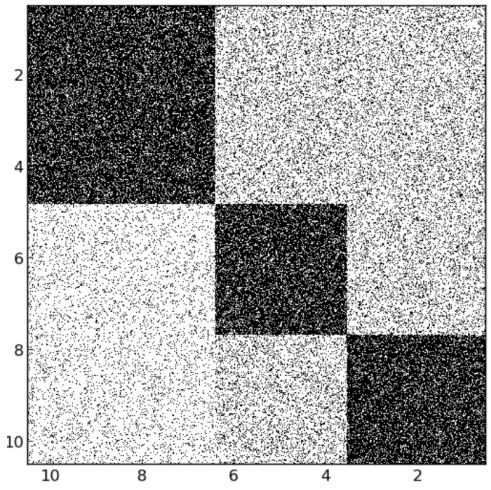
A Figura 37 mostra ilustrações de possíveis estruturas formadas a partir de algoritmos bloco-diagonais. Podemos observar que, considerando algoritmos bloco-diagonais, por atribuírem cada linha e coluna a exatamente um *co-cluster*, existe uma reorganização de linhas e colunas possibilitando que os *co-clusters* encontrados e reorganizados gerem *clusters* em uma diagonal.

Para este trabalho, como mencionado anteriormente, quatro algoritmos foram selecionados, sendo que dois algoritmos executam *co-clustering* maximizando a modularidade de grafos bipartidos e dois algoritmos utilizam e informação mútua para definir os *co-clusters*.

5.2.5.1 Co-clustering baseado na teoria da informação

No decorrer desta subseção será apresentada a estratégia de *co-clustering* baseado na teoria da informação. O conteúdo foi baseado nos trabalhos de Role, Morbieu e Nadif (2019) e Govaert e Nadif (2013).

Figura 37 – Exemplos de reorganização de dados a partir da aplicação de co-clustering bloco-diagonal



Fonte: Role, Morbieu e Nadif (2019)

Na estratégia de *co-clustering* baseado na teoria da informação, a tabela de contingência (matriz de dados com duas variáveis aleatórias) é considerada como distribuição de probabilidade empírica conjunta das variáveis aleatórias discretas (linhas e colunas da matriz ou, neste caso, documentos e termos, respectivamente). Dessa forma, o problema de *co-clustering* é tido como um problema de otimização considerando a teoria da informação, ou seja, a melhor maneira de organizar os dados da matriz é aquela que maximiza a informação mútua entre as variáveis aleatórias agrupadas, ou ainda, que minimiza a perda na informação mútua entre as variáveis aleatórias originais e a informação mútua entre as variáveis aleatórias agrupadas.

Algoritmos de *co-clustering* que utilizam a estratégia da teoria da informação agrupam linhas e colunas simultaneamente. Vale mencionar também que algoritmos que implementam esta estratégia não resultam em estruturas bloco-diagonal e o número de *clusters* de linha pode ser diferente do número de *clusters* de coluna.

5.2.5.2 Block co-clustering

O algoritmo de *co-clustering* bloco-diagonal tem como objetivo encontrar *co-clusters* ideais considerando que objetos e atributos devem ter o mesmo número de *clusters*, resultando em uma matriz de bloco-diagonal (consulte a Figura 18). Como é baseado na "modularidade do grafo bipartido"[2], o algoritmo busca maximizar uma medida da concentração de arestas dentro de *co-clusters* em comparação com a distribuição aleatória de

[2] Modularidade é um critério de qualidade frequentemente usado para detectar comunidades em grafos (GOVAERT; NADIF, 2013).

arestas entre todos os nós, de forma independente aos *co-clusters* (ROLE; MORBIEU; NADIF, 2019).

O algoritmo *Block co-clustering* utilizado neste projeto visa a maximização de uma versão adaptada de uma medida de modularidade usada normalmente em redes de computadores, o tornando eficazmente capaz de fazer *co-clustering* em matrizes binárias ou matrizes de contingência. Diferentemente de métodos disponíveis na literatura (DHILLON; MODHA, 2001; CHO et al., 2004), que utilizam relaxamentos espectrais, o método utilizado no algoritmo *Block co-clustering* permite obter *co-clusters* melhores (ROLE; MORBIEU; NADIF, 2019). Quando aplicado no contexto de matrizes de documento/termo, este método de *co-clustering* tem a vantagem de produzir diretamente descrições interpretáveis dos *clusters* de documentos.

5.2.5.3 Spectral Co-clustering

Algoritmo de *co-clustering* espectral é um algoritmo de *co-clustering* bloco-diagonal que busca maximizar a medida da concentração de arestas dentro de *co-clusters* em comparação com a distribuição aleatória de arestas entre todos os nós, porém utiliza versões normalizadas da matriz e maximiza a modularidade usando uma abordagem espectral. A utilização da matriz de modularidade normalizada é motivada pela necessidade de equilibrar os tamanhos dos *clusters* de linhas e colunas (ROLE; MORBIEU; NADIF, 2019).

O objetivo do *clustering* de blocos é tentar resumir essa matriz por blocos homogêneos. A ideia básica desse método consiste em fazer permutações de objetos e atributos para traçar uma estrutura de correspondência entre esses dois conjuntos. A medida de modularidade pode ser generalizada para realizar o *co-clustering* de dados binários e pode ser relacionada também a métodos de *co-clustering* espectral (ROLE; MORBIEU; NADIF, 2019).

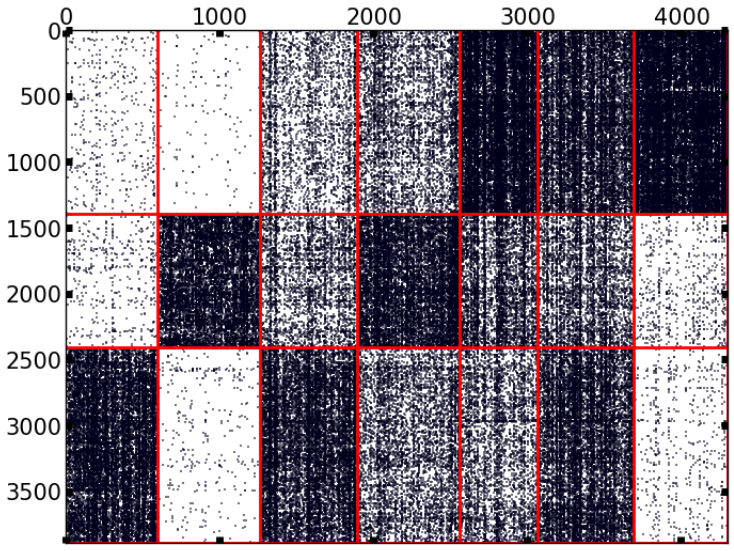
5.2.5.4 Algoritmo Information Theoretic Co-clustering

O algoritmo Information Theoretic Co-clustering (ou CoclustInfo) adota uma abordagem que utiliza teoria da informação e usa informações mútuas para definir seu critério (GOVAERT; NADIF, 2013). Outra diferença importante é que este algoritmo não busca descobrir uma estrutura de bloco diagonal como os algoritmos descritos anteriormente. O número de *clusters* de linha pode ser diferente do número de *clusters* de coluna. Um exemplo representativo do tipo de matriz obtida ao usar CoclustInfo é mostrado na Figura 38.

5.2.5.5 Algoritmo Co-clustering fuzzy

Algoritmo de *co-clustering fuzzy* é baseado na teoria da informação, sendo assim, o algoritmo busca reorganizar os dados de forma a minimizar a perda na informação mútua

Figura 38 – Clustering típico do Information Theoretic Co-clustering



Fonte: (ROLE; MORBIEU; NADIF, 2019)

entre as variáveis aleatórias originais e a informação mútua entre as variáveis aleatórias agrupadas. Nesta estratégia cada objeto/atributo pode pertencer a mais de um *cluster* (ROLE; MORBIEU; NADIF, 2019).

O algoritmo *co-clustering fuzzy* utilizado neste trabalho é o algoritmo chamado MMFCC (LIU; CHEN; CHAO, 2018), que combina modularidade com *co-clustering fuzzy*. Na função objetivo é implementado um procedimento de otimização iterativa via maximização de modularidade como o critério para o *co-clustering* de matrizes objeto-característica. Segundo (LIU; CHEN; CHAO, 2018), este algoritmo oferece algumas vantagens, como a produção direta de matriz diagonal e a descrição dos *co-clusters* resultantes, determinando automaticamente o número apropriado de *co-clusters*.

5.2.6 Avaliações dos resultados do co-clustering

As avaliações foram desenvolvidas para que possam fornecer uma boa compreensão sobre a qualidade das partições formadas tanto nos *clusters* de documentos quanto nos *clusters* de termos, bem como na associação entre *clusters* de documentos e *clusters* de termos. A avaliação da associação entre documentos e termos auxilia a avaliar a capacidade dos métodos de *co-clustering* em identificar relações significativas entre os *clusters* de documentos e os *clusters* de termos. Isso envolve a análise da correspondência entre os *clusters* de documentos e os *clusters* de termos, proporcionando uma compreensão mais profunda da estrutura dos dados.

Na avaliação dos métodos destacam-se as contribuições significativas do trabalho, especialmente nas avaliações de *clusters* de termos e na associação entre *clusters* de documentos e termos. Essas métricas não apenas oferecem uma visão abrangente da qualidade do *co-clustering*, mas também fornecem informações sobre a interpretabilidade dos resultados obtidos. Ao abordar esses aspectos, o este trabalho contribui para a validação sólida e abrangente de métodos de *co-clustering*, ampliando nossa compreensão das estruturas latentes presentes nos dados analisados.

5.2.6.1 Avaliação de clusters de documentos

Para a avaliação de *clusters* de documentos foram utilizados índices conhecidos.

Adjusted Rand Index (ARI)

O Índice Rand Ajustado (Adjusted Rand Index ou ARI) é uma das métricas amplamente utilizadas para validar o desempenho do *clustering*. Para o ARI é calculada a correspondência entre duas partições de dados e o índice avalia como objetos são classificados em uma matriz de contingência.

Para a matriz de contingência, considere que um conjunto de N objetos $X = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_i, ..., \boldsymbol{x}_N\}$ e que $U = \{u_1, u_2, ..., u_R\}$ e $V = \{v_1, v_2, ..., v_C\}$ representam duas partições diferentes dos objetos em S tais que $\cup_{i=1}^{R} {}^{u_i} = S = \cup_{j=1}^{C} {}^{v_j}$ e $u_i \cap u_{i'} = \emptyset = v_j \cap v_{j'}$ para $1 \leq i, i' \leq R$ e $1 \leq j, j' \leq C$. Dadas partições, U e V, a matriz de contingência pode ser formada para indicar sobreposição entre as partições U e V. A matriz de contingência é mostrada na Figura 39:

Figura 39 – Exemplo de matriz de contingência

		V				
	Group	v_1	v_2	$\cdots$	v_C	Total
U	u_1	t_{11}	t_{12}	$\cdots$	t_{1C}	$t_{1.}$
	u_2	t_{21}	t_{22}	$\cdots$	t_{2C}	$t_{2.}$
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	u_R	t_{R1}	t_{R2}	$\cdots$	t_{RC}	$t_{R.}$
Total		$t_{.1}$	$t_{.2}$	$\cdots$	$t_{.C}$	$t_{..} = n$

Fonte: Adaptado de Santos e Embrechts (2009)

Na matriz de contingência representada na Figura 39, a entrada t_{rc} representa o número de objetos que foram classificados. A partir do número total de combinações possíveis de pares de um determinado conjunto, podemos representar os resultados em quatro tipos diferentes de pares:

– a: O número de pares de elementos que estão no mesmo *cluster* em ambas as partições, U e V;

– b: O número de pares de elementos que estão no mesmo *cluster* na partição U e em *clusters* diferentes na partição V;

– c: O número de pares de elementos que estão em *clusters* diferentes na partição U e no mesmo *cluster* na partição V;

– d: O número de pares de elementos que estão em *clusters* diferentes em ambas as partições U e V.

Essencialmente, $a + d$ representa o número de pares de elementos que estão corretamente agrupados ou corretamente separados em ambas as partições, enquanto $a+b+c+d$ representa o número total de pares de elementos.

Através dos valores produzidos a partir dos resultados para os 4 tipos de pares é possível calcular o Rand Index (RI), como mostrado na Equação 27:

$$RI = \frac{a+d}{a+b+c+d} \tag{27}$$

Como forma de melhoria ao índice RI, Hubert e Arabie (1985) propõe o Adjusted Rand Index (ARI), índice geralmente recomendado para medir a concordância entre duas partições na análise de *clustering* com diferentes números de *clusters*.

o ARI é calculado como na Equação 28:

$$ARI = \frac{\binom{n}{2}\sum_{r=1}^{R}\sum_{c=1}^{C}\binom{t_{rc}}{2} - \left[\sum_{r=1}^{R}\binom{t_{r.}}{2}\sum_{c=1}^{C}\binom{t_{.c}}{2}\right]}{\frac{1}{2}\binom{n}{2} - \left[\sum_{r=1}^{R}\binom{t_{r.}}{2} + \sum_{c=1}^{C}\binom{t_{.c}}{2}\right] - \left[\sum_{r=1}^{R}\binom{t_{r.}}{2}\sum_{c=1}^{C}\binom{t_{.c}}{2}\right]} \tag{28}$$

Ou, apresentado na Equação 29 baseado no exemplo da Equação 27.

$$ARI = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c)+(c+b)(b+d)]}{\binom{n}{2}^2 - [(a+b)(a+c)+(c+d)(b+d)]} \tag{29}$$

Normalized Mutual Information (NMI)

A Informação Mútua Normalizada (Normalized Mutual Information ou NMI) é uma normalização da informação mútua para dimensionar os resultados entre 0 (nenhuma informação mútua) e 1 (correlação perfeita). A informação mútua é calculada como na Equação 30, onde $I(Y;C)$ é a informação mútua entre as partições Y e C e H é a entropia.

$$I(Y;C) = H(Y) - H(Y|C) \tag{30}$$

Considere como exemplo os *clusters* $C1$ e $C2$, considere também que 0 representa a classe 0 ($Y = 0$), 1 representa a classe 1 ($Y = 1$) e 2 representa a classe 2 ($Y = 2$): C1 = {0, 0, 0, 1, 1, 1, 1, 1, 1, 1} e C2 = {0, 0, 1, 1, 1, 2, 2, 2, 2, 2}

Considerando o *cluster* $C1$:

– P(Y = 0 | C = 1) = 3/10 (3 elementos 0 em $C1$)

– P(Y = 1 | C = 1) = 7/10 (7 elementos 1 em $C1$)

– P(Y = 2 | C = 1) = 0/10 (0 elementos 2 em $C1$)

O cálculo da entropia para o *cluster* $C1$ é apresentado na Equação 31, resultando na Equação 32.

$$I(Y;C=1) = -P(C=1)\sum_{y\in 1,2,3} P(Y=y|C=1)log(P(y=y)|C=1) \tag{31}$$

$$I(Y;C=1) - \frac{1}{2}\times\left[\frac{3}{10}log\left(\frac{3}{10}\right) + \frac{0}{10}log\left(\frac{0}{10}\right)\frac{7}{10}log\left(\frac{7}{10}\right)\right] = 0.4406 \tag{32}$$

Considerando o *cluster* C2:

– P(Y=0|C=2)=2/10 (2 elementos 0 em C2)

– P(Y=1|C=2)=3/10 (3 elementos 1 em C2)

– P(Y=0|C=2)=5/10 (5 elementos 2 em C3)

O cálculo da entropia para o *cluster* C2 é apresentado na Equação 33:

$$= -\frac{1}{2}\times\left[\frac{2}{10}log\left(\frac{2}{10}\right) + \frac{3}{10}log\left(\frac{3}{10}\right)\frac{5}{10}log\left(\frac{5}{10}\right)\right] = 0.7427 \tag{33}$$

Dessa forma, a informação mútua pode ser calculada como na Equação 34:

$$I(Y;C) = H(Y) - H(Y|C) = 1.5 - [0.4406 + 0.7427] = 0.3167 \tag{34}$$

E, o cálculo da NMI é apresentado na Equação 35:

$$NMI(Y,C) = \frac{2\times I(Y;C)}{[H(Y)+H(C)]} = \frac{2\times 0.3167}{[1.5+1]} = 0.2533 \tag{35}$$

NMI é uma medida externa e é uma boa forma para determinar a qualidade do *clustering*. Uma vez normalizado, é possível medir e comparar o NMI entre diferentes *clusters*.

Acurácia (ACC)

A acurácia (*accuracy*) é uma métrica externa de avaliação de um modelo, como classificação ou *clustering*, por exemplo, e é definida como a porcentagem de previsões corretas, ou seja, a razão entre o número de previsões corretas e o número total de previsões. A acurácia é calculada como na Equação 36:

$$ACC = \frac{TP+TN}{TP+TN+FP+FN} \tag{36}$$

– TP (True Positive ou verdadeiro positivo): a observação é positiva e é prevista corretamente;

– FN (False Negative ou Falso negativo): a observação é positiva, mas prevista incorretamente;

– TN (True Negative ou Verdadeiro negativo): a observação é negativa e prevista corretamente;

– FP (False Positive ou Falso positivo): a observação é negativa, mas prevista incorretamente.

5.2.6.2 Avaliação do clustering de termos

Para a avaliação do *clustering* de termos foi usado um índice proposto neste trabalho como uma adaptação de um existente (ROLE; MORBIEU; NADIF, 2018). Esta métrica é uma medida interna, já que não dispomos de *clusters* de termos como referência. As avaliações de *clustering* de termos foram feitas com base nos *clusters* de termos, sem considerar os clusters de *documentos*.

Métrica de Role, Morbieu e Nadif (2018) adaptada

Role, Morbieu e Nadif (2018) sugerem em seu trabalho uma medida de avaliação interna que utiliza vetores de word embeddings para avaliar a qualidade dos *clusters* de termos encontrados pelos algoritmos de *co-clustering*, baseada na similaridade de cosseno dos word embeddings normalizados. Nessa proposta, a métrica consiste na razão entre a similaridade *intra-cluster* média e a similaridade *inter-cluster* média, que pode ser interpretada como um cálculo similar ao da medida BetaCV[3], com a diferença de que é usada similaridade e não distância entre termos (não documentos) representados por seus embeddings. Portanto, quanto maior o valor, melhor a qualidade do *clustering*.

Neste trabalho, usamos uma forma adaptada da métrica de Role, Morbieu e Nadif (2018), para avaliação de *clusters* de termos, que chamamos de RMN_{GT}, em que a diferença com relação à métrica original é que são considerados, tanto para o cálculo de similaridade *intra-cluster* como *inter-cluster*, apenas os termos representantes dos *clusters* de termos (termo mais próximo do centroide) encontrados no *clustering* prévio. A RMN_{GT} é definida como a razão entre a similaridade *intra-clusters* e a similaridade *inter-clusters* do conjunto de *clusters* de termos obtidos por um algoritmo de *co-clustering* calculada utilizando-se os respectivos word embeddings de cada termo, como na Equação Equation 37:

$$RMN_{GT} = \frac{1\,S\_intra(GT)}{2\,S\_intra(GT)} \tag{37}$$

[3] BetaCV é uma métrica utilizada para avaliar a qualidade de partições em conjuntos de dados. Ela mede a coerência *intra-cluster* e a separação *inter-cluster*, fornecendo uma avaliação global da qualidade do *clustering* (HANDL; KNOWLES; KELL, 2005).

A similaridade *intra-clusters* e a similaridade *inter-clusters* do conjunto de *clusters* obtidos por um algoritmo de *co-clustering* são definidas, respectivamente, pela Equação 38:

$$S\_intra(GT) = \sum_{k=1}^{K} \frac{\sum_{t_i \in GT_k} \sum_{t_j \in GT_k} sim(v_i, v_j)}{|Pares\_GT_k|} \tag{38}$$

e pela Equação 39:

$$S\_inter(GT) = \sum_{k=1}^{K} \sum_{l=k+1}^{K} \frac{\sum_{t_i \in GT_k} \sum_{t_j \in GT_l} sim(v_i, v_j)}{|Pares\_GT_k\_GT_l|} \tag{39}$$

Onde:

- GT_k, $k = 1, ..., K$ é o conjunto dos K *clusters* de termos obtidos por um algoritmo de *co-clustering*;
- v_i e v_j são as representações vetoriais na forma de word embeddings dos termos t_i e t_j, respectivamente;
- $sim(v_i, v_j)$ é uma medida de similaridade entre v_i e v_j;
- $Pares\_GT_k = \{(t_i, t_j) | t_i \in GT_k, t_j \in GT_k, i < j\}$ é o conjunto de pares de termos do *cluster* GT_k.
- $Pares\_GT_k\_GT_l = \{(t_i, t_j) | t_i \in GT_k, t_j \in GT_l\}$ é o conjunto de pares de termos formados por termos dos *clusters* GT_k e GT_l.

5.2.6.3 Avaliação do ajuste entre clusters de documentos e clusters de termos

Nesta seção são descritas as métricas utilizadas para avaliar o ajuste entre os *clusters* de documentos e *clusters* de termos após aplicação do *co-clustering*. As duas métricas utilizadas são adaptações propostas neste trabalho de métricas encontradas na literatura. A primeira delas se baseia em similaridade de cosseno entre embeddings. A outra utiliza medida de Jaccard, que avalia a sobreposição entre *clusters* de termos e *clusters* de documentos.

Métrica Role, Morbieu e Nadif (2018) adaptada para avaliação de ajuste entre documentos e termos

Role, Morbieu e Nadif (2018) propõem uma métrica para avaliar a compatibilidade entre *clusters* de documentos e *clusters* de termos utilizando word embeddings. Para isso, um representante de cada tipo de *cluster* é calculado como o centroide do *cluster*. Para *clusters* de termos, o representante será o centroide dos vetores numéricos na forma de word embeddings que representam os termos do *cluster* e para *clusters* de documentos, o representante será o centroide dos representantes de cada documento que, por sua vez é o centroide dos vetores numéricos na forma de word embeddings que representam os termos

que aparecem no documento. Dessa forma, é possível calcular a similaridade de cosseno entre representantes de *clusters* de documentos e de termos.

Neste trabalho propomos e utilizamos uma versão adaptada dessa métrica, chamada de $RMN\_GD \times GT$, que tem duas diferenças com relação à versão original:

1. No cálculo do representante do *cluster* de termos são considerados apenas os representantes dos *clusters* de termos obtidos no *clustering* prévio, ou seja, os embedding dos termos da coleção que não são representantes de *clusters* não são utilizados no cálculo;

2. No cálculo do representante de documentos, os valores TF-IDF de cada *cluster* na matriz documentos × *cluster* é usado para ponderar os embedding dos representantes de *clusters* de termos que aparecem no documento.

A métrica $RMN_{GD \times GT}$ avalia o ajuste entre um *cluster* de documentos GD e um *cluster* de termos GT obtidos por um algoritmo de *co-clustering*, pelo cálculo da similaridade entre as representações vetoriais na forma de word embeddings do *cluster* de documentos e do *cluster* de termos, assim como na Equação 40:

$$RMN_{GD \times GT}(GD_i, GT_j) = sim(u_{GD_i}, v_{GT_j}) \tag{40}$$

Onde:

– GD_i é um *cluster* de documentos;

– GT_j é um *cluster* de termos;

– u_{GD_i} e v_{GT_j} são as representações vetoriais na forma de word embeddings do *cluster* de documentos GD_i e do *cluster* de termos GT_j, respectivamente;

– $sim(.,.)$ é uma medida de similaridade entre vetores.

As representações vetoriais de um *cluster* de documentos u_{GD_i} e de um *cluster* de termos v_{GT_j} são calculadas de acordo com a Equação 41, Equação 42 e a Equação 43:

$$u_{GD_i} = \frac{\sum_{d \in GD_i} u_d}{|GD_i|} \tag{41}$$

$$u_d = \frac{\sum_{t \in P_d} v_t \,.\, tf\_idf(d, g_t)}{|P_d|} \tag{42}$$

$$v_{GT_j} = \frac{\sum_{t \in GT_j} v_t}{|GT_j|} \tag{43}$$

Onde:

– u_d é a representação vetorial na forma de word embedding do documento d;

– v_t é a representação vetorial na forma de word embeddings do termo t;

– P_d é o conjunto de termos representantes dos *clusters* do *clustering* prévio de termos que ocorrem no documento d;

– g_t é o *cluster* de termos obtido no *clustering* prévio de termos do qual t é o representante.

Avaliação utilizando coeficiente de Jaccard

O índice de similaridade Jaccard ou coeficiente de similaridade Jaccard (JACCARD, 1901) compara membros de dois conjuntos com objetivo de encontrar membros compartilhados e membros distintos. Mede a semelhança de dois conjuntos de dados, com um intervalo de pontuação entre 0 e 1, sendo 0 para populações completamente distintas e 1 para populações iguais. Quanto maior o valor, mais semelhantes são as duas populações.

O coeficiente de Jaccard entre dois conjuntos A e B é apresentado na Equação 44:

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{44}$$

Selosse, Jacques e Biernacki (2020) utilizaram o coeficiente de Jaccard como uma medida de avaliação da qualidade de *clusters* de termos isoladamente. A avaliação foi feita começando com o cálculo do coeficiente de Jaccard entre o conjunto de termos de um *co-cluster* e o conjunto de termos de cada um dos documentos da coleção toda, e em seguida calculando a média desses valores. O valor médio obtido é entendido como uma avaliação da coesão entre os termos do *co-cluster*.

Neste projeto, o coeficiente de Jaccard foi aplicado de uma forma distinta daquela proposta por Selosse, Jacques e Biernacki (2020), com o objetivo de avaliar o ajuste entre *clusters* de documentos e *clusters* de termos. Desse modo, são calculados os coeficientes de Jaccard entre o conjunto de termos de um *co-cluster* e o conjunto de termos de cada documento de um *co-cluster* e, em seguida, calculada a média desses valores. O conjunto de termos de um documento é extraído da matriz documentos $\times$ *clusters*, incluindo os termos representantes dos *clusters* que têm TF-IDF maior que zero nessa matriz. Note que, nessa métrica, ao contrário da métrica anterior, os cálculos são feitos utilizando os termos e não seus embeddings.

Considere que GD_i é um *cluster* de documentos e GT_j é um *cluster* de termos, obtidos por um algoritmo de *co-clustering*. A medida Jaccard entre GD_i e GT_j é calculada pela Equação 45:

$$Jaccard(GD_i, GT_j) = \frac{\sum_{d \in GD_i} Jaccard(P_d, GT_j^{'})}{|GD_i|} \tag{45}$$

Onde:

– P_d é o conjunto de termos representantes dos *clusters* obtidos no *clustering* prévio de termos que ocorrem no documento d;

– $GT_j^{'}$ é o conjunto de termos do *cluster* GT_j que têm valor maior que zero na matriz documentos $\times$ *clusters*.

5.3 Recursos e Ferramentas

A escolha de recursos e ferramentas computacionais adequados para implementação de um projeto é fundamental. Todos os recursos e ferramentas utilizados neste projeto foram previamente pesquisados e escolhidos de acordo com a necessidade.

5.3.1 Ferramentas Computacionais

Algumas ferramentas necessárias para o desenvolvimento deste projeto, úteis durante o pré-processamento dos arquivos, são:

– Word embeddings: O documento de word embeddings usado neste projeto foi aprendido e distribuído pelo fastText[4], biblioteca gratuita e *open source* que permite aprender representações de texto. A biblioteca disponibiliza vetores de termos pré-treinados para 157 idiomas, treinados em Common Crawl e Wikipedia. Tais modelos foram treinados para 300 dimensões utilizando CBoW e janela de tamanho 5 e 10. O algoritmo utilizado pela fastText é baseado em Bojanowski et al. (2017)

– Python: Os algoritmos criados neste projeto foram implementados utilizando a linguagem Python. A escolha da linguagem se deve às vantagens que apresenta, como facilidade de implementação e organização, disponibilidade de bibliotecas com implementações fundamentais para o projeto, como algoritmos de tratamento de texto, algoritmos de *clustering* e *co-clustering*. Em especial:

  a) Biblioteca Scikit-learn[5]: é uma biblioteca de aprendizado de máquina de código aberto, que disponibiliza algoritmos como, por exemplo, para classificação e *clustering*, além de implementações de métricas, como similaridade de cosseno (PEDREGOSA et al., 2011).

  b) Biblioteca NLTK[6]: Natural Language Toolkit (NLTK) (BIRD; KLEIN; LOPER, 2009) é uma biblioteca Python de código aberto para PLN. A biblioteca disponibiliza *corpora* e recursos lexicais.

  c) Biblioteca Coclust[7]: A biblioteca Coclust fornece um pacote com implementações de algoritmos de *co-clustering*, tanto bloco-diagonais como não diagonais (ROLE; MORBIEU; NADIF, 2019).

  d) Outras bibliotecas importantes para a elaboração deste projeto são: Gensim[8] (REHUREK; SOJKA, 2011), Numpy[9] (HARRIS et al., 2020) e Pandas[10] (TEAM, 2020).

[4] https://fasttext.cc/
[5] https://scikit-learn.org/
[6] https://www.nltk.org/
[7] https://coclust.readthedocs.io/en/v0.2.1/
[8] https://pypi.org/project/gensim/
[9] https://numpy.org/
[10] https://pandas.pydata.org/

Capítulo 6

Experimentos e Resultados

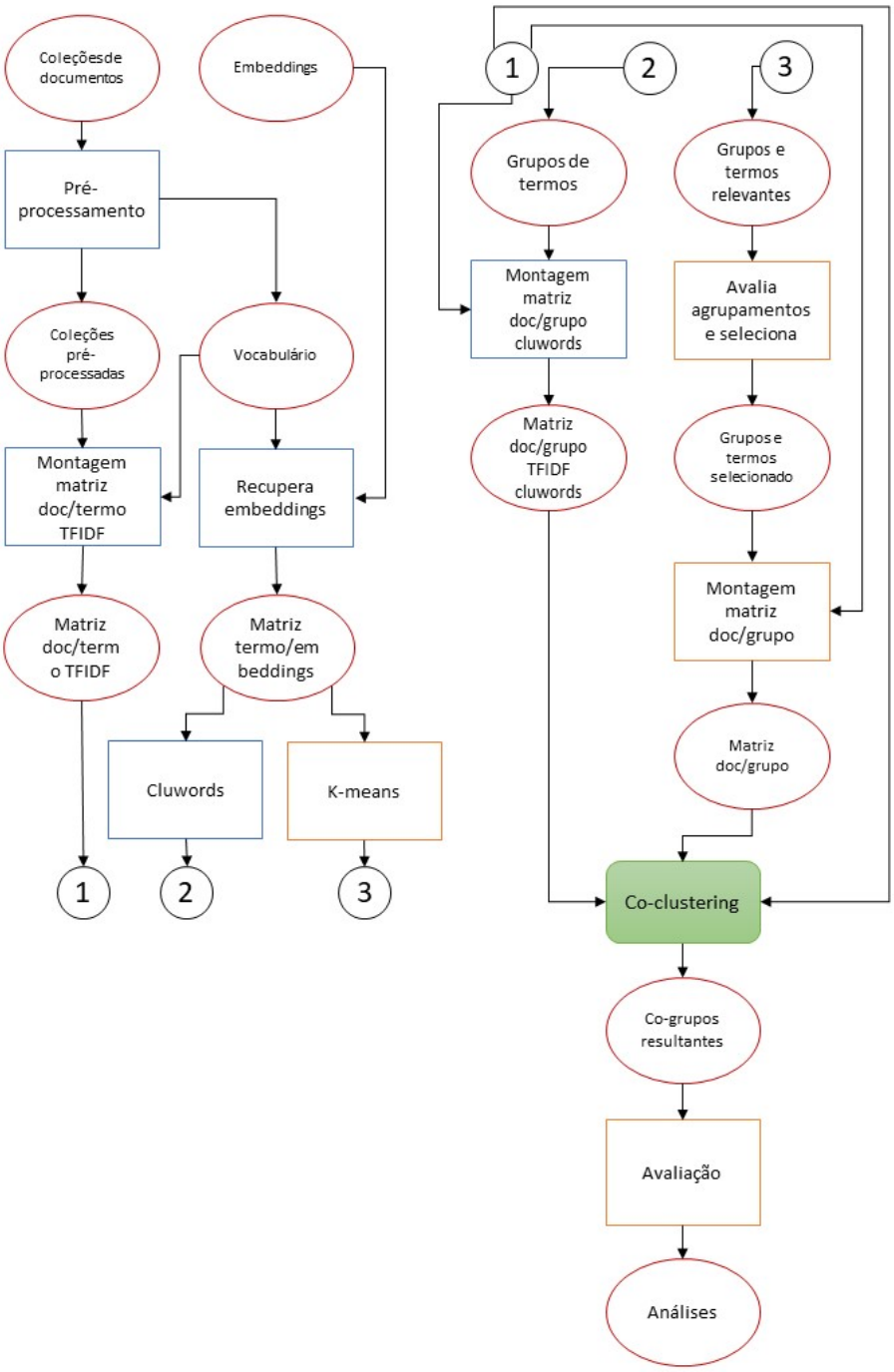
Neste capítulo são descritos os experimentos realizados e os resultados obtidos. Na Seção 6.1 é descrita a etapa de clustering de termos. Na Seção 6.2 são apresentadas as versões de matrizes TF-IDF incluídas nos experimentos para fins de comparação. Na Seção 6.3 são apresentados e analisados os resultados do primeiro grupo de experimentos, que são aqueles em que os algoritmos de co-clustering são configurados com números iguais de clusters de documentos e de termos. Na seção 6.4 são apresentados e analisados resultados do segundo grupo de experimentos, que são aqueles em que os algoritmos de co-clustering encontram número de clusters de termos diferente do número de clusters de documentos. Na Figura 40 são ilustrados os passos realizados até a avaliação.

6.1 Escolha do número de clusters para o clustering de termos

Como descrito na Seção 5.2.2, os termos foram previamente agrupados. Para tal *clustering* foram escolhidas quantidades específicas de *clusters*, sendo que, para cada coleção de documentos, o *clustering* foi feito de 5 formas diferentes utilizando o algoritmo K-means: *clustering* considerando o número ideal de *clusters*, com aplicação do método do cotovelo, e números de *clusters* igual a 5%, 10%, 25% e 50% do número total de termos. Esses números de *clusters* foram selecionados com base em experimentos preliminares que incluíram outros números de *clusters*. Com base nesses experimentos preliminares foi possível verificar que os resultados obtidos com um número elevado de *clusters* não traz informações adicionais úteis para as análises.

Com diferentes configurações pretende-se investigar o impacto do *clustering* prévio

Figura 40 – Fluxograma dos passos dos experimentos



Fonte: Produzido pelo autor

no desempenho do *co-clustering*. Os números de *clusters* de cada uma das coleções de documentos são apresentados na Tabela 20.

Para avaliar a qualidade do *clustering* de termos, como não há disponibilidade de rótulos, foram aplicadas três métricas internas de avaliação: Silhueta, Inércia e índice de Davies-Bouldin. As avaliações para cada coleção de documentos são apresentadas

Tabela 20 – Número de clusters para clustering de termos das coleções.

	Nº ideal	5%	10%	25%	50%
Classic3	25 (0,54%)	231	462	1155	2310
Classic4	27 (0,57%)	234	468	1171	2342
CSTR	27 (0,54%)	248	497	1242	2485
Newsgroup5	33 (1,74%)	94	189	473	946
Reuters8	23 (0,25%)	453	906	2265	4531
Sports	30 (0,14%)	1056	2113	5284	10569

Fonte: Produzido pelo autor.

nas Tabelas 21, 22, 23, 24, 25 e 26, em que os melhores valores de cada métrica estão destacados em negrito.

É possível notar padrões nos resultados. Considerando a avaliação de silhueta é perceptível que os valores têm um pequeno aumento conforme o número de *clusters* também aumenta, porém ainda próximo de 0. Um valor de 0 indica que amostras estão no limite de decisão entre *clusters* vizinhos, que podem ser sobrepostos. Valores negativos indicam que amostras podem ter sido atribuídas incorretamente (ROUSSEEUW, 1987). Em termos de inércia, à medida que o número de *clusters* aumenta, o valor diminui. O mesmo ocorre para o índice de Davies-Bouldin.

Tabela 21 – Classic3 - Clustering de termos.

	Nº ideal (0,54%)	5%	10%	25%	50%
Silhueta	0.009	0.009	0.007	0.027	**0.037**
Inertia	14415.883	12011.763	10704.951	7688.909	**4031.775**
DB	4.421	3.075	2.472	1.598	**1.063**

Fonte: Produzido pelo autor.

Tabela 22 – Classic4 - Clustering de termos.

	Nº ideal (0,57%)	5%	10%	25%	50%
Silhueta	0.000	0.006	0.007	0.024	**0.039**
Inertia	14338.877	12076.306	10745.056	7710.548	**4033.8064**
DB	4.530	3.146	2.434	1.609	**1.061**

Fonte: Produzido pelo autor.

Tabela 23 – CSTR - Clustering de termos.

	Nº ideal (0,54%)	5%	10%	25%	50%
Silhueta	0.001	0.003	0.022	0.059	**0.080**
Inertia	14422.873	12167.227	10712.391	7380.136	**3627.316**
DB	4.818	3.122	2.473	1.642	**1.044**

Fonte: Produzido pelo autor.

Tabela 24 – Newsgroup5 - Clustering de termos.

	Nº ideal (1,74%)	5%	10%	25%	50%
Silhueta	0.010	0.010	0.016	0.025	**0.035**
Inertia	4712.092	4271.998	3810.650	2777.313	**1482.900**
DB	3.825	3.078	2.639	1.658	**1.134**

Fonte: Produzido pelo autor.

Tabela 25 – Reuters8 - Clustering de termos.

	Nº ideal (0,25%)	5%	10%	25%	50%
Silhueta	-0.001	0.001	0.012	0.041	**0.061**
Inertia	27813.957	22180.915	19480.361	13403.551	**6510.091**
DB	4.675	2.988	2.336	1.545	**0.984**

Fonte: Produzido pelo autor.

Tabela 26 – Sports - Clustering de termos.

	Nº ideal (0,14%)	5%	10%	25%	50%
Silhueta	-0.009	0.000	0.015	0.047	**0.063**
Inertia	68663.938	52570.560	45906.325	31140.923	**15127.181**
DB	4.757	2.917	2.352	1.530	**0.972**

Fonte: Produzido pelo autor.

Com base nos valores de silhueta, podemos observar que, embora haja um discreto aumento dos valores à medida que o número de *clusters* aumenta, todas os valores estão bem próximos de zero, indicando que os *clusters* obtidos estão sobrepostos. Para as outras métricas, observa-se que *clustering* de maior qualidade são aqueles com maior número de *clusters*, o que é esperado devido às características das métricas. Esses resultados podem ser considerados como esperados, já que as diferenças entre os vetores numéricos que representam os termos são pequenas. Para os propósitos da abordagem proposta, não é necessário encontrar *clusters* de termos bem separados.

Após o *clustering* dos termos foram produzidas matrizes documento × clusters_de _termos, como mencionado na Seção 5.2.3, que representam os dados de entrada para aplicação dos algoritmos de *co-clustering*. Diferentes configurações de *clustering* produzem diferentes matrizes. Por exemplo, a matriz TF-IDF gerada a partir da coleção de documentos Classic3, composta por 3891 documentos e 4620 termos (após pré-processamento) é formada por 3891 linhas e 4620 colunas. Devido a aplicação do *clustering* prévio de termos, as matrizes documento × clusters_de_termos produzidas apresentam dimensionalidade consideravelmente menor. A matriz formada a partir dos dados do *clustering* considerando 25 clusters, por exemplo, é composta por 3891 linhas e 25 colunas, muito mais densa em comparação a matrizes TF-IDF.

As matrizes documento × clusters_de_termos mencionadas foram submetidas a 4 algoritmos de *co-clustering* citados na Seção 5.2.4.

6.2 Configurações selecionadas para comparação

Visando obter resultados que permitissem análises comparativas, foram incluídas nos experimentos, além das matrizes TF-IDF geradas a partir dos *clusters* de termos, como mencionado na seção anterior, duas formas diferentes de geração da matriz TF-IDF. Uma delas é a matriz TF-IDF gerada da forma tradicional, com o cálculo da medida TF-IDF de cada par documento-termo feito como descrito na seção 3.2.2. A outra versão de matriz TF-IDF inserida nos experimentos vem de uma proposta de método para modelagem de tópicos proposta da literatura, que também explora a informação semântica contida nos word embeddings para encontrar *clusters* de termos similares (VIEGAS et al., 2019). Nessa proposta são calculadas as similaridades entre todos os pares de termos do vocabulário usando similaridade de cosseno. Na sequência, para cada termo, são selecionados os termos que têm uma similaridade superior a um limiar definido previamente com esse termo. Dessa forma, cada termo determina um *cluster* de termos similares, que é chamado de CluWords. Viegas et al. (2019) propõem, então, uma forma modificada de gerar a matriz TF-IDF, em que as medidas tradicionais de TF e IDF de um termo t são substituídas por medidas de TF e IDF do seu respectivo CluWord, que leva em conta todos os termos similares a t.

Note que na proposta de *clustering* de termos selecionada para comparações existe uma grande sobreposição entre *clusters* de termos, já que cada termo pode compor o CluWord de vários termos. Na abordagem explorada na proposta apresentada aqui, cada termo pode pertencer a apenas um *cluster*, já que esses *clusters* foram obtidos pelo algoritmo K-means.

6.3 Experimento 1

O Experimento 1 consiste em realizar o *co-clustering* para matriz documento × clusters_de_termos de cada uma das 6 coleções de documentos utilizadas neste projeto utilizando 4 diferentes algoritmos.

Neste experimento, para todas as matrizes documento × clusters_de_termos, os algoritmos de *co-clustering* foram configurados considerando realizar *clustering* sempre com o mesmo número de conjuntos de documentos e conjuntos de termos, considerando a quantidade de rótulos de documentos que cada conjunto de documentos tem. Por exemplo, a coleção de documentos Classic3 é formada por documentos pertencentes a 3 categorias. Sendo assim, foi considerado que a aplicação do *co-clustering* agrupasse documentos em 3 *clusters* e termos em 3 *clusters*.

Esse conjunto de experimentos foi feito com os 4 algoritmos de *co-clustering* selecionados para o trabalho, com todos os 6 conjuntos de dados e com as matrizes TF-IDF geradas a partir do *clustering* das 5 formas diferentes, descritas na seção 5.2.2, além das

duas versões de matrizes TF-IDF para comparação, descritas na seção 6.2.

6.3.1 Avaliação do clustering de documentos

Inicialmente, os resultados dos algoritmos são analisados considerando-se os *clusters* de documentos, individualmente. Como neste conjunto de experimentos o número de *clusters* de termos e de documentos é o mesmo que o número de classes, foram utilizadas as métricas externas de avaliação de *clustering*: NMI, ARI e Acurácia (ACC), descritas na seção 5.2.5.1. Os resultados são apresentados em tabelas nas quais cada coluna contém as métricas referentes à execução de um algoritmo usando uma forma diferente de geração da matriz TF-IDF. Assim, as 5 primeiras colunas referem-se às matrizes com *clusters* de termos e as duas últimas colunas referem-se à matriz TF-IDF sem *clusters* e os resultados gerados pelo método CluWords (VIEGAS et al., 2019), respectivamente.

Classic3

Para a coleção de documentos Classic3, considerando as matrizes documento × clusters_de_termos, nos 4 algoritmos, o resultado obtido pelo TF-IDF sem *clusters* é melhor em todos os casos, considerando as 3 métricas de avaliação. Nota-se também que o TF-IDF CluWords apresenta bom desempenho considerando a coleção de documentos, porém inferior ao TF-IDF sem *clusters*.

Nas Tabelas 27, 28, 29 e 30 é mostrado os resultados para os 4 algoritmos.

Tabela 27 – Classic3 - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.3351	0.7611	0.7335	0.7645	0.4400	**0.9272**	0.8057
ARI	0.2697	0.8327	0.8094	0.8348	0.4184	**0.9568**	0.8507
ACC	0.6209	0.9416	0.9331	0.9408	0.6610	**0.9856**	0.9475

Fonte: Produzido pelo autor.

Tabela 28 – Classic3 - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.2726	0.6687	0.6991	0.7565	0.7801	**0.9444**	0.8654
ARI	0.2980	0.7467	0.7763	0.8302	0.8454	**0.9694**	0.9015
ACC	0.6197	0.9051	0.9195	0.9406	0.9457	**0.9897**	0.9668

Fonte: Produzido pelo autor.

A Tabela 30 mostra os resultados do algoritmo co-clustering *fuzzy*, onde é possível perceber maior estabilidade no desempenho para as matrizes documento × clusters_de_termos, com exceção do subconjunto considerando o número ideal de *clusters*.

Tabela 29 – Classic3 - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.6949	0.7459	0.4671	0.4965	0.0094	**0.8968**	0.7156
ARI	0.7509	0.7960	0.2875	0.3180	0.0002	**0.9200**	0.7223
ACC	0.9077	0.9241	0.5715	0.5828	0.3754	**0.9717**	0.8984

Fonte: Produzido pelo autor.

Tabela 30 – Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.4517	0.6944	0.6953	0.7433	0.7927	**0.9328**	0.8790
ARI	0.5154	0.7711	0.7754	0.8163	0.8579	**0.9628**	0.9148
ACC	0.8008	0.9162	0.9198	0.9342	0.9498	**0.9874**	0.9709

Fonte: Produzido pelo autor.

CSTR

Para a coleção de documentos CSTR, em 3 algoritmos, as matrizes documento × clusters_de_termos apresentaram melhor desempenho quando comparados ao TF-IDF sem *clusters* e CluWords (VIEGAS et al., 2019), especialmente considerando os algoritmos block co-clustering e spectral co-clustering, onde todos os subconjuntos do modelo proposto apresentam desempenho superior. Para o algoritmo INFO, as matrizes documento × clusters_de_termos apresentam desempenho inferior ao TF-IDF CluWords em apenas um caso.

Nas Tabelas 31, 32, 33 e 34 são apresentados os resultados para a coleção de documentos CSTR, respectivamente, para os algoritmos INFO, block co-clustering e spectral co-clustering.

Tabela 31 – CSTR - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0464	**0.0812**	0.0645	0.0604	0.0538	0.0356	0.0695
ARI	0.0204	0.0359	0.0457	0.0398	**0.0458**	0.0169	0.0309
ACC	0.3478	0.3913	**0.3946**	0.3645	0.3913	0.3277	0.3612

Fonte: Produzido pelo autor.

Tabela 32 – CSTR - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0517	0.0642	0.0666	0.0695	0.0794	0.0251	**0.0943**
ARI	0.0458	0.0324	0.0423	0.0442	0.0488	0.0060	**0.0698**
ACC	0.3913	0.3645	0.4113	**0.3980**	0.3712	0.3244	0.3314

Fonte: Produzido pelo autor.

Diferentemente do que foi observado nas avaliações da coleção de documentos Classic3 e olhando para os resultados considerando as matrizes documento × clusters_de_termos,

Tabela 33 – CSTR - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1482	0.1883	0.2534	0.2681	**0.2779**	0.0445	0.0443
ARI	0.0883	**0.1534**	0.0989	0.1063	0.1125	0.0011	-0.0069
ACC	0.5418	**0.5886**	0.4882	0.4949	0.4983	0.4314	0.3745

Fonte: Produzido pelo autor.

Tabela 34 – CSTR - Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal (0,54%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0774	0.0815	0.0632	0.2196	0.0580	**0.2301**	0.1890
ARI	0.0507	0.0673	0.0475	0.1644	0.0320	**0.1911**	0.1507
ACC	0.4013	0.3913	0.3946	0.5217	0.3612	**0.5585**	0.5185

Fonte: Produzido pelo autor.

não é possível notar um padrão de desempenho crescente quando o número de *clusters* também cresce. É possível notar que o modelo nº ideal teve um desempenho comparável aos outros casos, o que não acontece na coleção Classic3.

É válido destacar o desempenho obtido pelos modelos no algoritmo spectral co-clustering, apresentando maiores índices quando comparado a outros algoritmos e uma diferença considerável em relação ao TF-IDF sem *clusters* e TF-IDF CluWords. Este resultado é o exatamente o esperado nesta avaliação, quando utilizadas coleções de documentos consideradas difíceis para a tarefa de *clustering*.

Reuters8

Para Reuters8, no algoritmo INFO, as matrizes documento × clusters_de_termos apresentam melhor desempenho em 4 casos (5%, 10%, 25% e 50%, considerando ARI e ACC), sendo que apenas a matriz nº ideal apresenta desempenho inferior ao TF-IDF sem *clusters* e TF-IDF CluWords. Os resultados são apresentados na Tabela 35.

Tabela 35 – Reuters8 - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal (0,25%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.2521	0.3160	0.3502	0.3551	0.3712	**0.3797**	0.3050
ARI	0.1682	0.2398	0.2388	0.3239	**0.3368**	0.2219	0.2777
ACC	0.3206	0.3933	0.3864	0.4755	**0.4988**	0.3590	0.3645

Fonte: Produzido pelo autor.

Para os outros algoritmos, TF-IDF sem *clusters* e TF-IDF CluWords apresentam melhor desempenho. Isto pode ser observado nas Tabelas 36, 37 e 38.

O desempenho do algoritmo spectral co-clustering em Reuters8, assim como para a coleção de documentos CSTR, foi o melhor entre os 4 algoritmos. Os resultados são apresentados na Tabela 37.

Tabela 36 – Reuters8 - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal (0,25%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.2135	0.2885	0.3267	0.2905	0.3196	0.3336	**0.4393**
ARI	0.2538	0.3187	0.3982	0.4004	0.3816	0.4229	**0.5621**
ACC	0.4266	0.5066	0.5655	0.5386	0.5212	0.5436	**0.5998**

Fonte: Produzido pelo autor.

Tabela 37 – Reuters8 - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal (0,25%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1065	0.4284	0.4921	0.4495	0.4852	**0.6382**	0.2861
ARI	0.0547	0.4197	0.5315	0.4401	0.4731	**0.6867**	0.1379
ACC	0.4929	0.6971	0.7039	0.7327	0.7464	**0.7724**	0.3280

Fonte: Produzido pelo autor.

Tabela 38 – Reuters8 - Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal (0,25%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1800	0.2702	0.2976	0.3391	0.3574	**0.4819**	0.4546
ARI	0.1657	0.2949	0.3623	0.3103	0.4260	0.5473	**0.5494**
ACC	0.3978	0.4399	0.5363	0.4892	0.5632	**0.6509**	0.5938

Fonte: Produzido pelo autor.

Sports

Para a coleção Sports, os resultados das avaliações NMI e ARI são baixos, inclusive para TF-IDF sem *clusters*. Isto acontece porque os *clusters* são sobrepostos. Segundo Hajj, Rizk e Awad (2019), a coleção Sports é um conjunto difícil para a tarefa de *clustering*. Nestes casos, agrupar termos previamente pode melhorar o resultado. Os resultados para a coleção de documentos Sports considerando todos os algoritmos são melhores quando comparados ao TF-IDF sem *clusters* e TF-IDF CluWords. Vale destacar o equilíbrio apresentado nos resultados do algoritmos spectral co-clustering e o resultado obtido por TF-IDF CluWords no algoritmo block co-clustering.

As Tabelas 39, 40, 41 e 42 apresentam os resultados em testes com a coleção Sports considerando os algoritmos INFO, block co-clustering, spectral co-clustering e co-clustering *fuzzy*, respectivamente.

Tabela 39 – Sports - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0005	0.0017	0.0015	0.0068	**0.0177**	0.0073	0.0135
ARI	-0.0003	0.0036	0.0033	0.0071	**0.0287**	0.0052	0.0028
ACC	0.5210	0.5360	0.5350	0.5450	**0.5870**	0.5400	0.5390

Fonte: Produzido pelo autor.

Da mesma forma que a coleção CSTR, segundo Hajj, Rizk e Awad (2019), Sports

Tabela 40 – Sports - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0125	0.0343	0.0119	0.0113	0.0126	0.0030	**0.0729**
ARI	0.0154	0.0506	0.0231	0.0125	0.0169	0.0024	**0.0951**
ACC	0.5640	0.6140	0.5800	0.5580	0.5670	0.5290	**0.6550**

Fonte: Produzido pelo autor.

Tabela 41 – Sports - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	-4.0000	**0.0207**	**0.0207**	**0.0207**	0.0063	0.0013	0.0184
ARI	0.0000	**0.0029**	**0.0029**	**0.0029**	-0.0008	-0.0008	**0.0029**
ACC	0.6350	**0.6370**	**0.6370**	**0.6370**	0.6340	0.6340	0.5430

Fonte: Produzido pelo autor.

Tabela 42 – Sports - Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0158	0.0118	0.0269	0.0121	0.0103	0.0131	**0.0527**
ARI	0.0186	0.0188	0.0453	0.0242	0.0152	0.0226	**0.0769**
ACC	0.5700	0.5710	**0.6090**	0.5820	0.5640	0.5780	0.6000

Fonte: Produzido pelo autor.

é uma coleção difícil para a tarefa de *clustering*. Por esse motivo é justificável que a metodologia proposta neste trabalho obteve maiores índices em relação ao TF-IDF sem *clusters* e TF-IDF CluWords. Vale destacar o desempenho do algoritmo spectral co-clustering apresentado na Tabela 41. Esta avaliação reafirma os resultados esperados.

6.3.2 Avaliação do clustering de termos

Aqui, os resultados dos algoritmos serão analisados considerando os *clusters* de termos, também individualmente. No caso dos *clusters* de termos, não conhecemos as categorias verdadeiras dos *clusters*. Portanto, a avaliação é feita com métricas internas de avaliação de *clustering* sendo uma delas a métrica proposta por Role, Morbieu e Nadif (2018) e adaptada neste trabalho para uso com *clustering* prévio, nomeada aqui de RMN_{GT}, e uma métrica já conhecida na literatura: Davies-Bouldin, descritas na seção 5.2.5.2. Vale lembrar que esta forma de avaliação não é feita na maioria dos trabalhos que utilizam *co-clustering* para organizar coleções de documentos.

Os resultados das avaliações considerando os *clusters* de termos apresentam alguns padrões para todos os casos. A métrica RMN_{GT} apresenta pequenas variações considerando todos os subconjuntos do modelo, TF-IDF sem *clusters* e TF-IDF CluWords.

Excepcionalmente, quando o *clustering* é desbalanceado, ou seja, quando um *cluster* é formado por poucos termos em comparação com outros *clusters*, é possível notar picos. Exemplos de picos são mostrados na Tabela 43, com resultados da avaliação dos *clusters*

de termos no conjunto de dados Newsgroup5, para o algoritmo INFO, e na Tabela 50, com resultados da avaliação dos *clusters* de termos no conjunto de dados Sports, para o algoritmo *co-clustering fuzzy*.

Tabela 43 – Newsgroup5 - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal (1,74%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	0.4922	**1.1933**	0.5006	0.5021	0.5002	0.4999	0.5004
DB	**1.9263**	2.7404	3.6350	4.3599	4.2628	5.0519	4.6628

Fonte: Produzido pelo autor.

A seguir são apresentados os resultados nas Tabelas 44, 45 e 46 para o *clustering* de termos para a coleção de documentos Newsgroup5.

Tabela 44 – Newsgroup5 - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal (1,74%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	**0.5053**	0.5029	0.4965	0.4987	0.4995	0.5002	0.5001
DB	**2.3140**	2.7148	3.3313	4.0791	5.0742	5.0142	4.6607

Fonte: Produzido pelo autor.

Tabela 45 – Newsgroup5 - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal (1,74%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	0.4853	**0.5003**	0.4848	0.4953	0.4830	0.4360	0.4885
DB	**1.5723**	2.8648	3.4979	3.8623	5.0167	5.0546	4.6587

Fonte: Produzido pelo autor.

Tabela 46 – Newsgroup5 - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal (1,74%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	0.4989	0.5001	0.4977	**0.5004**	0.5001	0.4995	0.4999
DB	**1.5723**	2.8648	3.4979	3.8623	5.0167	5.0546	4.6587

Fonte: Produzido pelo autor.

Padrões também são notados nos resultados da métrica Davies-Bouldin, como observado na Tabela 43, onde os valores tendem a ser menores (melhores) para a matriz com o número ideal de *clusters*.

As avaliações das coleções Reuters8 e Sports apresentam valores mais altos, para o índice de DB, quando comparados a outras coleções. Isto pode indicar que o *clustering* de termos, para estas coleções de documentos, foi pior. Alguns resultados das avaliações da coleção Sports podem ser vistos nas Tabelas 47, 48, 49 e 50.

Para a avaliação do *clustering* de termos, é possível notar que a melhor abordagem, considerando a métrica Davies-Bouldin, é aquela onde o *clustering* prévio foi feito com o

Tabela 47 – Sports - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	1.9442	1.9960	1.9998	1.9998	**2.0000**	1.9997	1.9999
DB	**0.1123**	0.9037	4.3810	4.7144	4.8828	3.5054	3.6741

Fonte: Produzido pelo autor.

Tabela 48 – Sports - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	1.9934	1.9998	1.9990	1.9998	**2.0001**	1.9995	1.9995
DB	**0.1123**	0.9036	4.3810	4.7100	4.8825	3.5051	3.7983

Fonte: Produzido pelo autor.

Tabela 49 – Sports - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	1.7510	**2.5506**	1.8187	1.9620	2.1394	2.0000	2.0001
DB	**0.1116**	0.7258	4.3822	4.7168	4.8826	3.5049	3.6214

Fonte: Produzido pelo autor.

nº ideal e termos, já que para todos os casos apresentados, obteve os melhores resultados. Considerando a métrica RMN_{GT}, o melhor desempenho varia bastante, concentrando em 5% e 50% do número total de termos.

Para o *clustering* de termos é importante destacar que os modelos propostos neste trabalho obtiveram melhor desempenho considerando as métricas Davies-Bouldin e a métrica RMN_{GT} em quase todos os casos. Os resultados justificam o *clustering* prévio de termos.

6.3.3 Avaliação da associação entre clusters de documentos e clusters de termos

Nesta seção, os resultados dos algoritmos são analisados considerando simultaneamente os *clusters* de documentos e os *clusters* de termos. As métricas utilizadas baseiam-se em medidas de similaridade entre *clusters* de documentos e de termos e foram propostas neste trabalho (Seção 5.2.6.3) como adaptações de métricas encontradas na literatura, uma adaptação da métrica apresentada em Role, Morbieu e Nadif (2018), $RMN_{GD\times GT}$, e Similaridade de Jaccard (JACCARD, 1901), de forma similar ao apresentado em Se-

Tabela 50 – Sports - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal (0,14%)	5%	10%	25%	50%	Sem clusters	CluWords
RMN_{GT}	1.7510	**2.5506**	1.8187	1.9620	2.1394	2.0000	2.0001
DB	**0.1116**	0.7258	4.3822	4.7168	4.8826	3.5049	3.6214

Fonte: Produzido pelo autor.

losse, Jacques e Biernacki (2020), para que se adequassem às representações usadas nesta abordagem, em que as colunas da matriz TF-IDF representam *clusters* de termos. Assim como acontece com as avaliações da seção anterior, esta forma de avaliação não é feita na maioria dos trabalhos que utilizam *co-clustering* para organizar coleções de documentos.

Para a métrica $RMN_{GD \times GT}$ é esperado que o valor seja maior quanto maior a associação entre *clusters* de documentos e *clusters* de termos que formam um *co-cluster*. Tal padrão foi encontrado em alguns casos. Devido à quantidade elevada de resultados dos experimentos, apenas alguns mapas de calor selecionados são apresentados, sendo esses os mais representativos para fundamentar as análises e apontar os benefícios da métrica de avaliação. Esta avaliação foi configurada de modo que o número de *clusters* de documentos é igual ao número de *clusters* de termos. Para a coleção de documentos Newsgroup5 o padrão fica evidenciado. Para análise, considere, em todos os resultados, associações entre *clusters* de linhas e colunas com o mesmo nome, por exemplo, associação entre Cluster1 de linhas com Cluster1 de colunas.

Os resultados desta avaliação são apresentados na forma de mapas de calor que indicam os valores da métrica de avaliação calculada para todos os pares de *clusters* de documentos e *clusters* de termos. As Figuras 41, 42 e 43 ilustram mapas de calor onde é possível visualizar que os valores de *clusters* de termos e *clusters* de documentos são maiores em *co-clusters* associados (nas diagonais), que representam a avaliação de associação entre *clusters* de documentos com o nº ideal (33), 5% (94) e 10% (189) de *clusters* de termos, respectivamente. Os valores mostrados nas Figuras 41, 42 e 43 provam que existe maior associação em *co-clusters* compostos por *clusters* de documentos e *clusters* de termos associados.

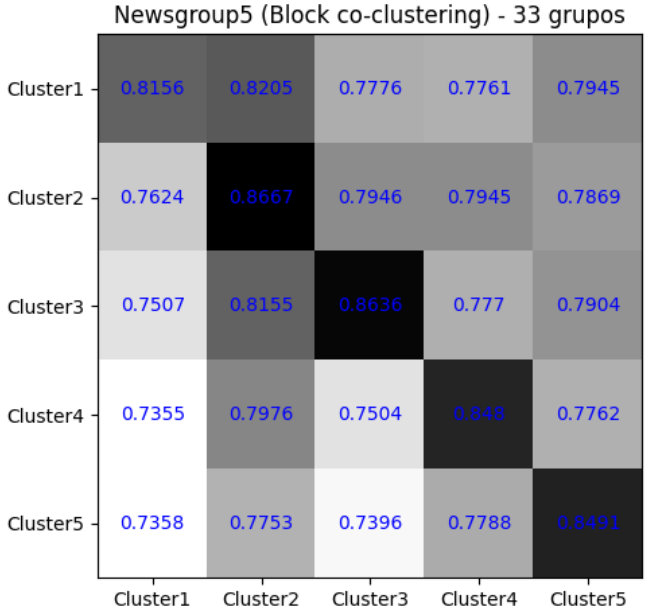
É valido mencionar que, a medida que o número de *clusters* aumenta, o padrão deixa de ser tão claro, o que pode ser visto na Figura 44 e na Figura 45, que representam a avaliação de associação entre *clusters* de documentos e termos para o algoritmo block co-clustering, com 25% (473) 50% (946), respectivamente.

Observando a Figura 45 é possível notar linhas com valores similares, ou seja, com pouca variação no resultado da métrica independentemente do *cluster* de colunas. É interessante mencionar que a avaliação de associação entre *clusters* de documentos para TF-IDF sem *clusters* apresenta o mesmo padrão, como observado na Figura 46.

Para o TF-IDF CluWords se observa padrões de linhas coerentes como é apresentado na Figura 47. Esse resultado evidencia que o uso de *clustering* prévio de termos permite obter um resultado mais significativo da métrica proposta.

Para a coleção de documentos Classic4 e o algoritmo spectral co-clustering também é possível notar padrões diagonais mas, diferentemente do que acontece com a coleção de documentos Newsgroup5 e o algoritmo block co-clustering, o padrão é mais evidenciado em subconjuntos do modelo com maior número de *clusters*. Nas Figuras 48, 49, 50 e 51 é ilustrada tal situação.

Figura 41 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - Nº ideal (33)



Fonte: Produzido pelo autor

Em alguns casos é possível notar *clusters* de termos com forte associação com vários ou todos os *clusters* de documentos. Role, Morbieu e Nadif (2018) cita que tais *clusters* podem ser ruídos. Acreditamos que estes casos ocorrem quando termos representativos do *cluster* de termos aparecem com frequência em documentos.

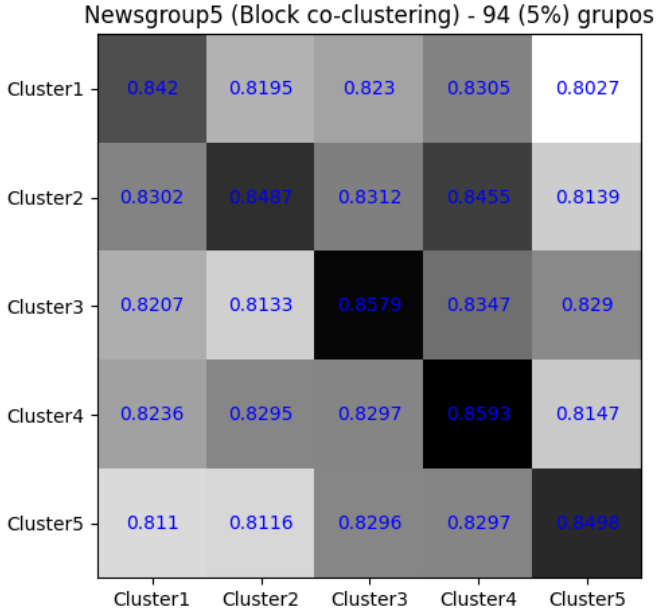
Em situações onde o *clustering* de documentos obtém melhor desempenho (na coleção Classic3) é possível notar que, além da alta associação entre *clusters* de documentos e *clusters* de termos, é maior a diferença para outras situações onde *clusters* de termos e documentos não têm associação. Isso pode ser melhor observado nas Figuras 52, 53, 54 e 55.

A aplicação do coeficiente de Jaccard nos resultados mostra tendências de que o número de ocorrência dos termos varia pouco em relação aos conjuntos de documentos, como mostrado nas Figuras 56, 57, 58, 59, 60 e 61, que apresentam avaliações para aplicação do algoritmo spectral co-clustering para a coleção CSTR. Vale destacar que as Figuras 57, 58 e 59 apresentam resultados similares.

Diferente do que foi observado para a coleção CSTR, para o conjunto Newsgroup5, em vários casos, as avaliações mostram padrões em linhas, como ilustrado nas Figura 62, Figura 63, Figura 64, Figura 65, Figura 66.

A similaridade de Jaccard não se mostrou uma métrica informativa no sentido de avaliar a associação entre *clusters* de documentos e de termos.

Figura 42 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - 5% (94)



Fonte: Produzido pelo autor

6.4 Experimento 2

Alguns testes realizados no decorrer deste trabalho mostraram que a variação do número de *clusters* de termos influencia no desempenho do algoritmo INFO co-clustering. Portanto, para o Experimento 2, foram considerados diferentes números de *clusters* de termos para o *co-clustering*. Foram utilizadas apenas as coleções de documentos Classic3 e CSTR. Por exemplo, considerando a matriz da coleção de documentos Classic3 formada a partir dos dados do *clustering* considerando 25 clusters, o número de *clusters* de termos pode variar entre 3 (número real de *clusters* de documentos) até 25.

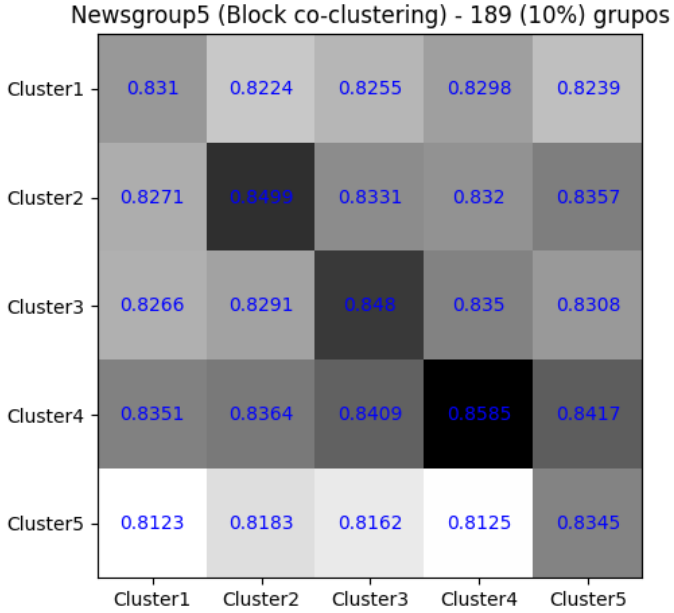
Em análises prévias foi notado que tais variações influenciam no desempenho do *co-clustering*. Os testes foram feitos empiricamente e os melhores resultados (em termos de acurácia) foram selecionados e avaliados.

6.4.1 Avaliação do clustering de documentos: Experimento 2

Nas Tabelas 51 e 52 são ilustrados os resultados considerando diferentes números de *clusters* de termos para o *co-clustering* (o número de *clusters* é mostrado entre parênteses nos cabeçalhos de colunas).

A Tabela 51 mostra que o desempenho do algoritmo INFO co-clustering melhora quando existem variações do número de *clusters* de termos em relação aos resultados obtidos pelo Experimento 1, para todos os casos. Para a matriz documento × clus-

Figura 43 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - 10% (189)



Fonte: Produzido pelo autor

ter_de_termos com número ideal de *clusters*, para a coleção Classic3, por exemplo, como mostrado na Tabela 27, o desempenho em termos de acurácia (ACC) nos resultados do Experimento 1 foi de 0.6209, enquanto que no Experimento 2, com número ideal de *clusters* para 14 *clusters* de termos, o desempenho foi de 0.9432.

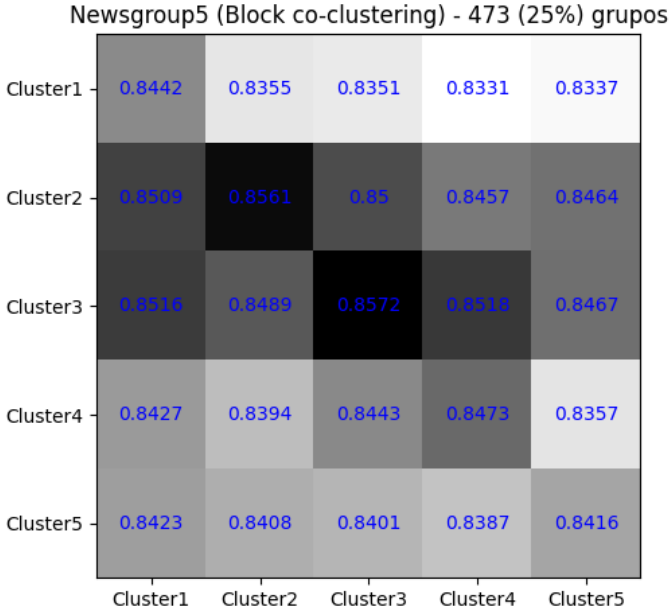
Tabela 51 – Classic3 - Avaliação de clusters de documentos - INFO co-clustering - Variação de número de clusters.

	Nº ideal (14)	5% (10)	10% (16)	25% (214)	50% (5)	Sem clusters (147)	CluWords (102)
NMI	0.7697	0.8916	0.8937	0.9082	0.9154	**0.9558**	0.8999
ARI	0.8372	0.9338	0.9375	0.9471	0.9525	**0.9761**	0.9401
ACC	0.9432	0.9773	0.9786	0.9820	0.9838	**0.9920**	0.9813

Fonte: Produzido pelo autor.

Para a coleção de documentos CSTR o mesmo se repete, ou seja, o desempenho do algoritmo INFO co-clustering melhora quando existem variações do número de *clusters* de termos, para todos os casos. Na Tabela 52 é possível notar melhora no desempenho em todos os casos, para todos os índices e para todos os números de *clusters* de termos, o que pode ser visto quando comparamos os resultados na Tabela 31 e na Tabela 52.

Figura 44 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - 25% (473)



Fonte: Produzido pelo autor

Tabela 52 – CSTR - Avaliação de clusters de documentos - INFO co-clustering - Variação de número de clusters.

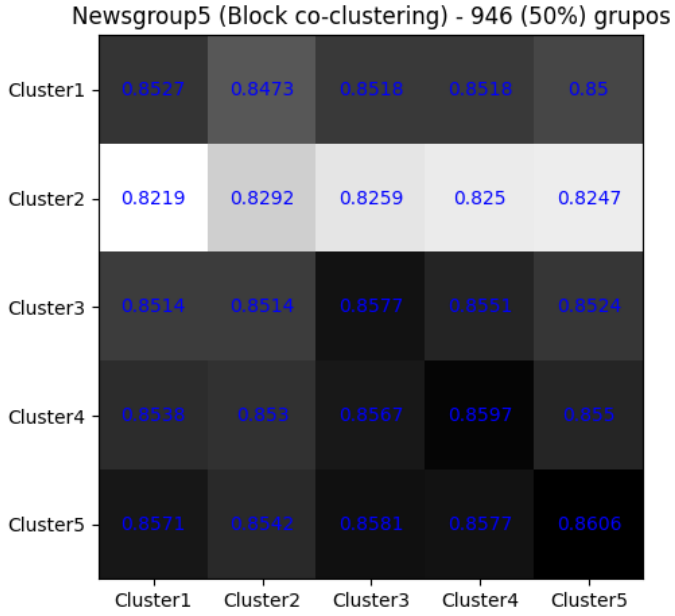
	Nº ideal (14)	5% (10)	10% (16)	25% (214)	50% (5)	Sem clusters (147)	CluWords (115)
NMI	**0.1767**	0.1314	0.1763	0.0756	0.0591	0.0646	0.0715
ARI	**0.1410**	0.1214	0.1347	0.0588	0.0682	0.0350	0.0474
ACC	0.4782	0.4882	**0.5217**	0.3979	0.4381	0.3879	0.3966

Fonte: Produzido pelo autor.

6.4.2 Avaliação do clustering de termos: Experimento 2

Para o Experimento 2, os resultados da avaliação do *clustering* de termos são apresentados nas Tabelas 53 e 54, onde os valores exibidos na primeira linha se referem ao número de *clusters* cujo experimento obteve melhor desempenho em termos das métricas RMN_{GT} e Davies-Bouldin. Por exemplo, para a coleção Classic3, para a primeira coluna da Tabela 53, o número ideal de *clusters* (*clustering* prévio), segundo o método do cotovelo, é 25. No Experimento 2, os 25 *clusters* de termos foram reagrupados em 14 *clusters*, diferentemente do Experimento 1, onde os 25 *clusters* de termos foram reagrupados em 3 *clusters*, mesma quantidade de rótulos de documentos.

Figura 45 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - 50% (946)



Fonte: Produzido pelo autor

Tabela 53 – Classic3 - Avaliação de clusters de termos - INFO co-clustering - Variação de número de clusters.

	Nº ideal (14)	5% (10)	10% (16)	25% (214)	50% (5)	Sem clusters (147)	CluWords (102)
RMN_{GT}	0.9685	0.9954	1.0002	1.0005	0.9978	1.0002	**1.0011**
DB	**0.5251**	1.0707	1.8356	1.0353	4.7109	4.2052	4.0019

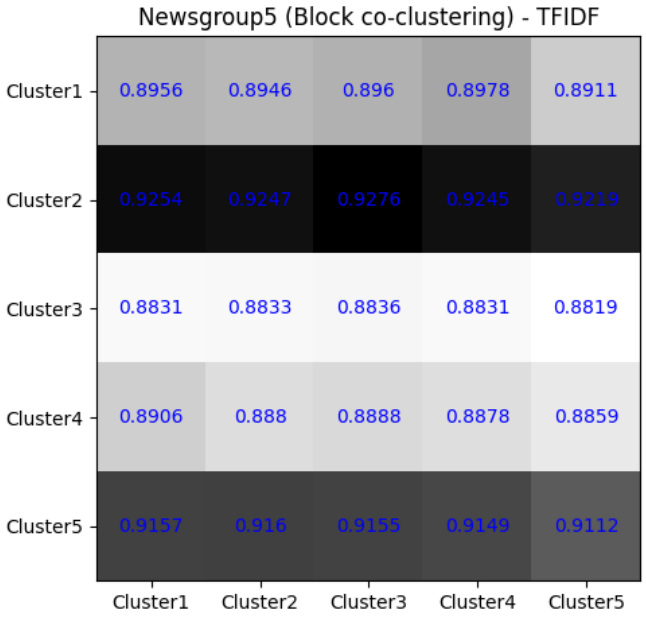
Fonte: Produzido pelo autor.

Tabela 54 – CSTR - Avaliação de clusters de termos - INFO co-clustering - Variação de número de clusters.

	Nº ideal (14)	5% (10)	10% (16)	25% (214)	50% (5)	Sem clusters (147)	CluWords (115)
RMN_{GT}	**0.6787**	0.6692	0.6684	0.6672	0.6664	0.6666	0.6667
DB	**0.7539**	2.8165	3.3477	1.8335	4.4758	3.5672	3.8882

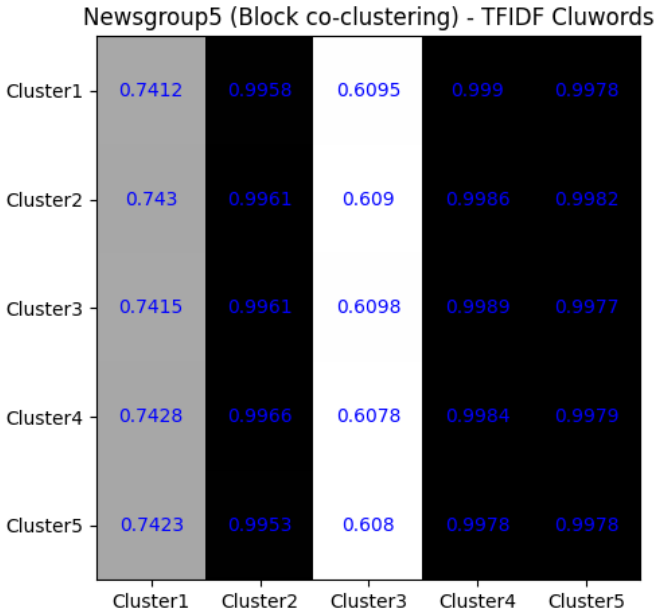
Fonte: Produzido pelo autor.

Figura 46 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - TF-IDF



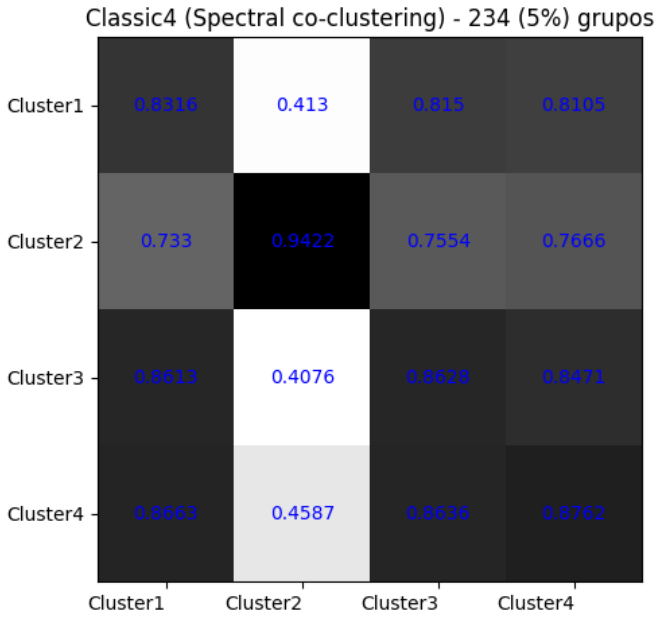
Fonte: Produzido pelo autor

Figura 47 – Avaliação de associação entre clusters de documentos e termos - NG5 Block co-clustering - TF-IDF CluWords



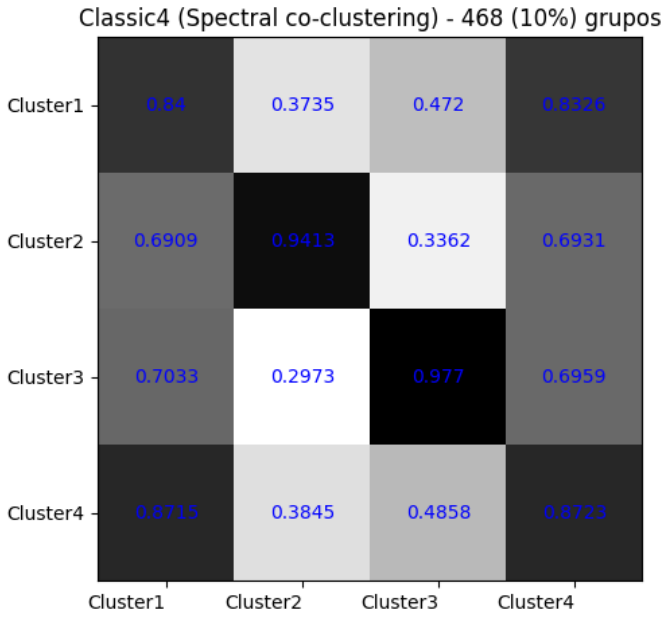
Fonte: Produzido pelo autor

Figura 48 – Avaliação de associação entre clusters de documentos e termos - Classic4 Spectral co-clustering - 5% (234)



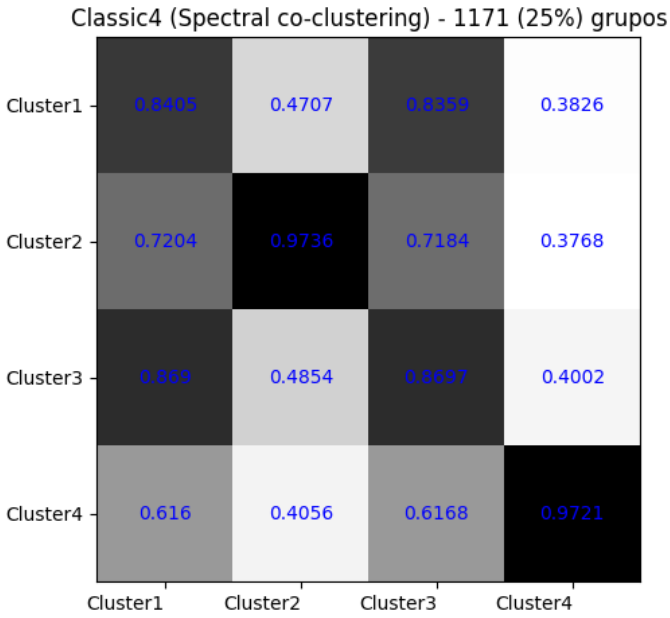
Fonte: Produzido pelo autor

Figura 49 – Avaliação de associação entre clusters de documentos e termos - Classic4 Spectral co-clustering - 10% (468)



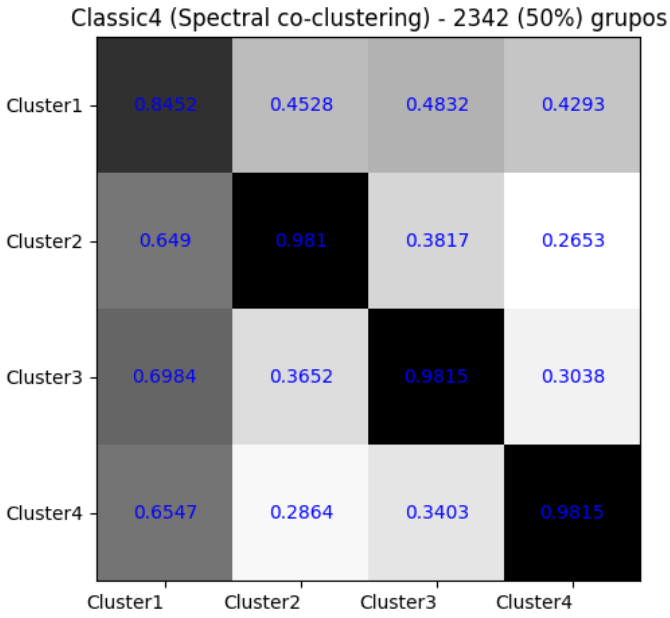
Fonte: Produzido pelo autor

Figura 50 – Avaliação de associação entre clusters de documentos e termos - Classic4 Spectral co-clustering - 25% (1171)



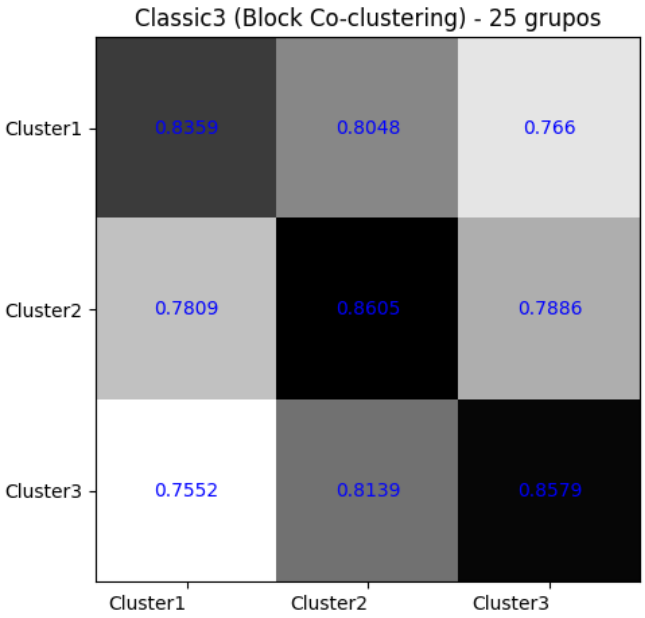
Fonte: Produzido pelo autor

Figura 51 – Avaliação de associação entre clusters de documentos e termos - Classic4 Spectral co-clustering - 50% (2342)



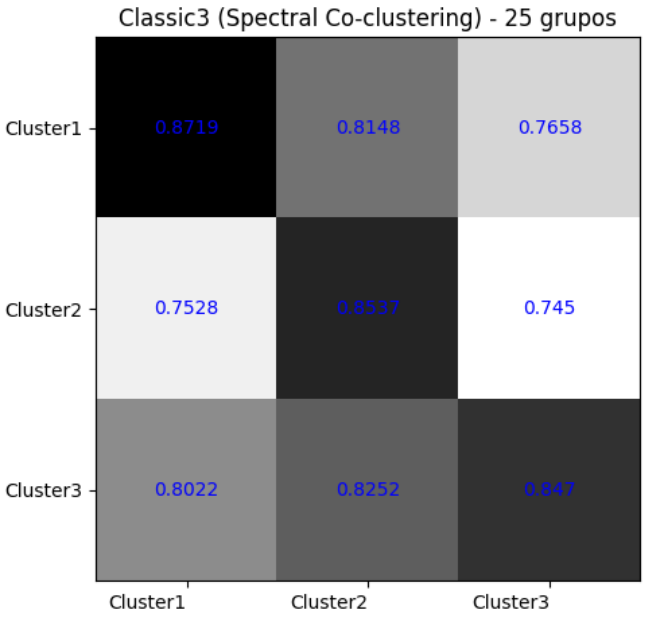
Fonte: Produzido pelo autor

Figura 52 – Avaliação de associação entre clusters de documentos e termos - Classic3 - Block co-clustering - Nº ideal (25)



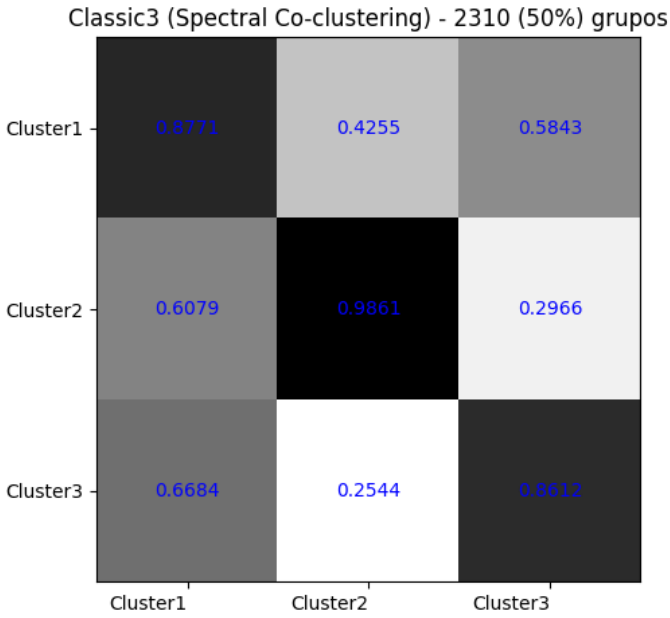
Fonte: Produzido pelo autor

Figura 53 – Avaliação de associação entre clusters de documentos e termos - Classic3 - Spectral co-clustering - Nº ideal (25)



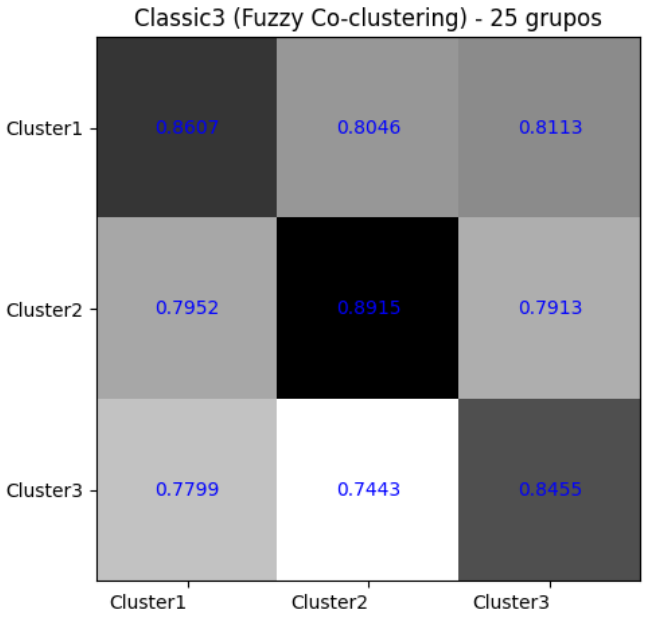
Fonte: Produzido pelo autor

Figura 54 – Avaliação de associação entre clusters de documentos e termos - Classic3 - Spectral co-clustering - 50% (2310)



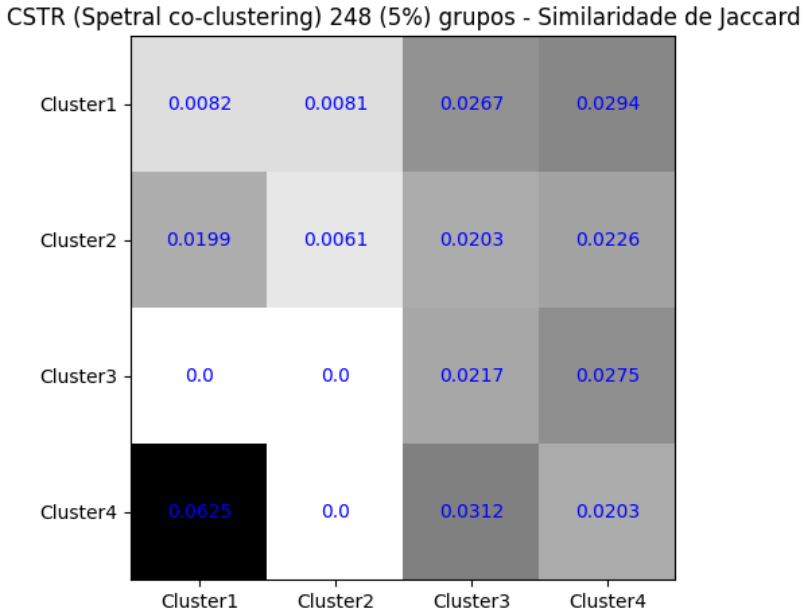
Fonte: Produzido pelo autor

Figura 55 – Avaliação de associação entre clusters de documentos e termos - Classic3 co-clustering fuzzy - Nº ideal (25)



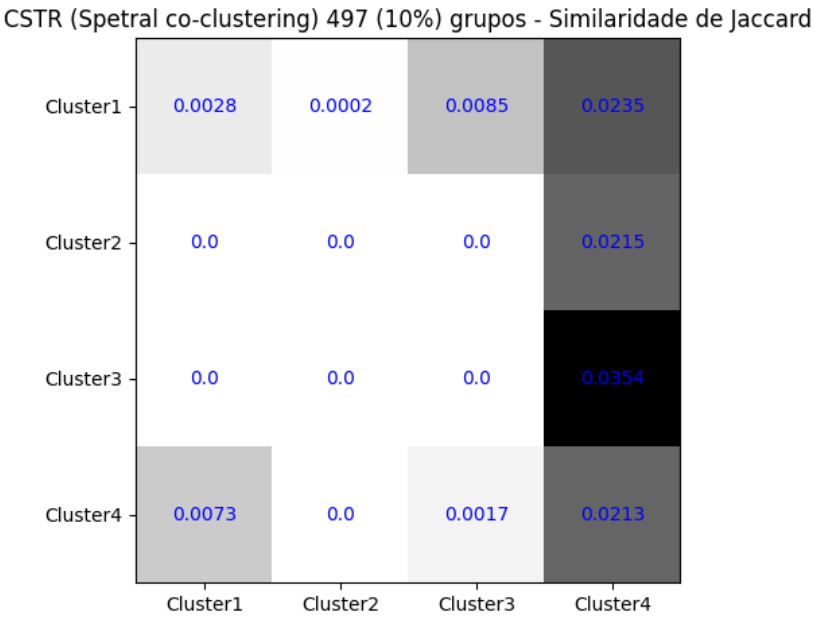
Fonte: Produzido pelo autor

Figura 56 – CSTR - Spectral co-clustering - 5% (248) - Similaridade de Jaccard



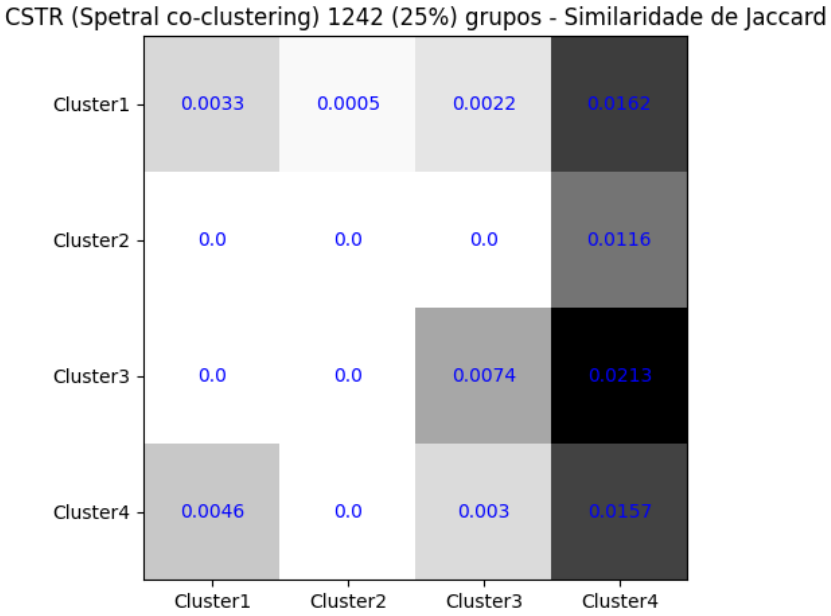
Fonte: Produzido pelo autor

Figura 57 – CSTR - Spectral co-clustering - 10% (497) - Similaridade de Jaccard



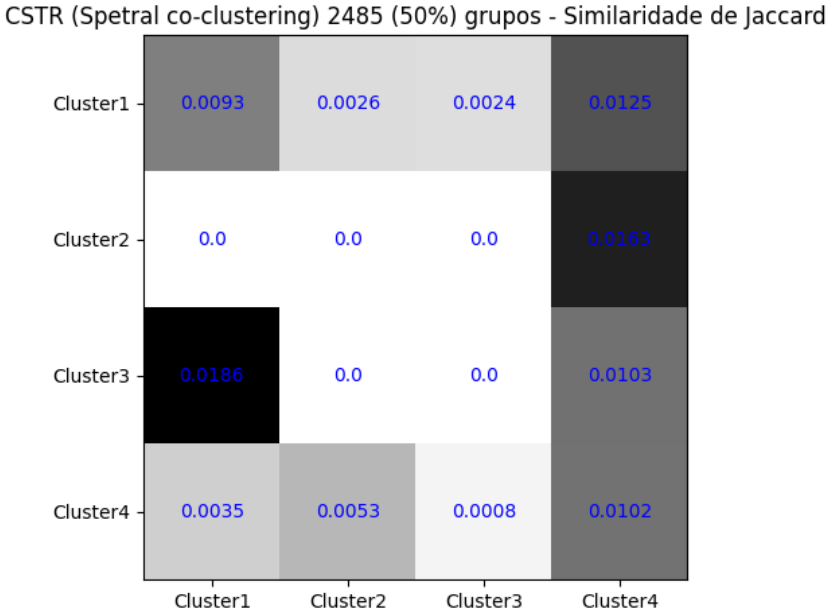
Fonte: Produzido pelo autor

Figura 58 – CSTR - Spectral co-clustering - 25% (1242) - Similaridade de Jaccard



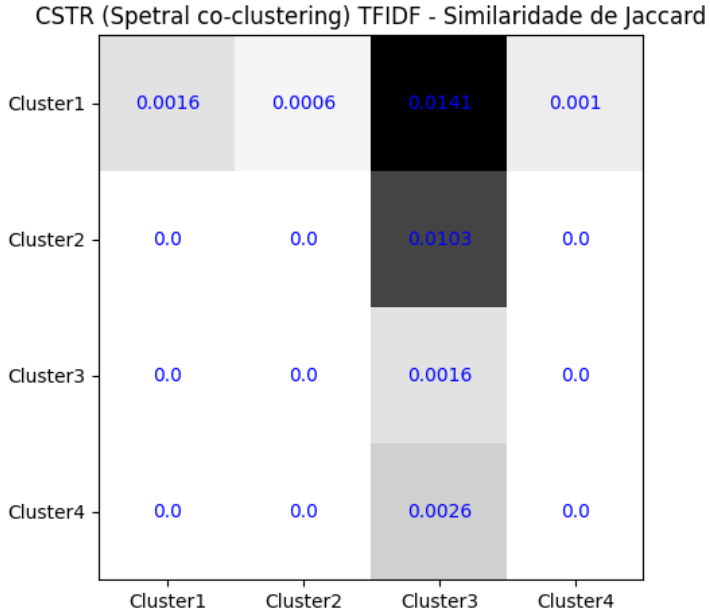
Fonte: Produzido pelo autor

Figura 59 – CSTR - Spectral co-clustering - 50% (2485) - Similaridade de Jaccard



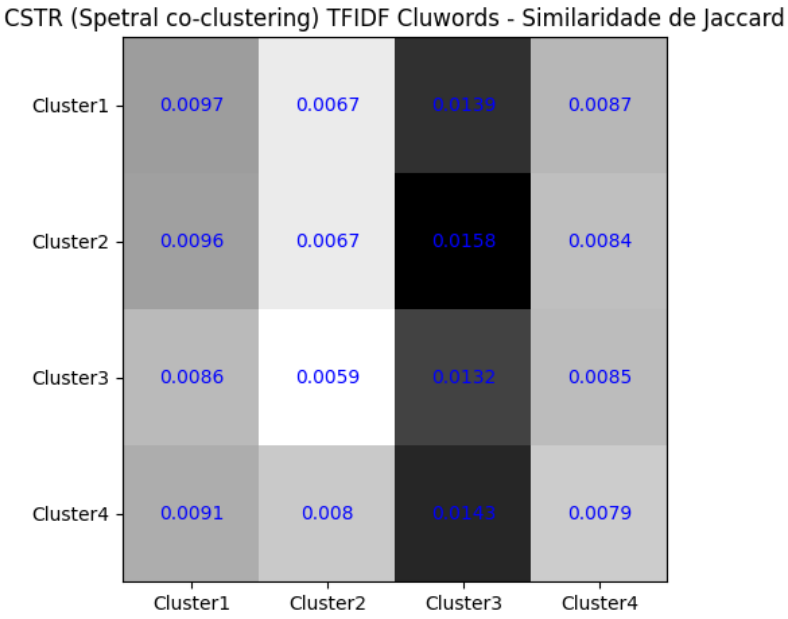
Fonte: Produzido pelo autor

Figura 60 – CSTR - Spectral co-clustering - TF-IDF - Similaridade de Jaccard



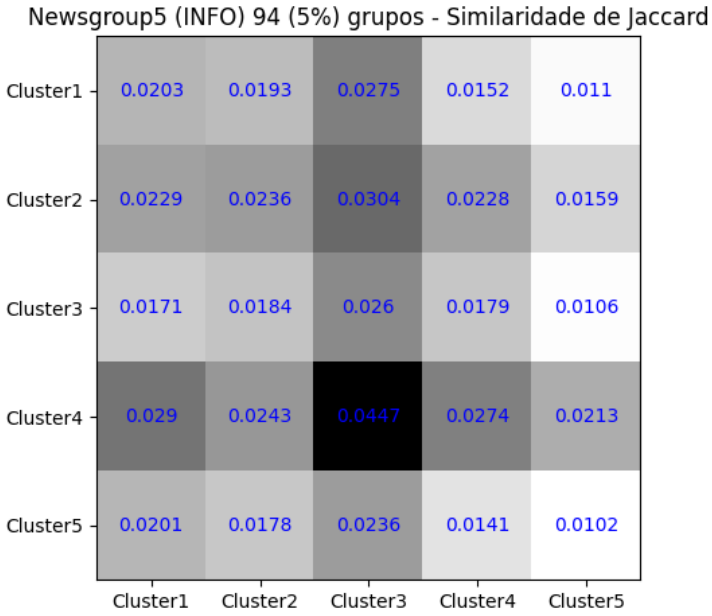
Fonte: Produzido pelo autor

Figura 61 – CSTR - Spectral co-clustering - TF-IDF CluWords - Similaridade de Jaccard



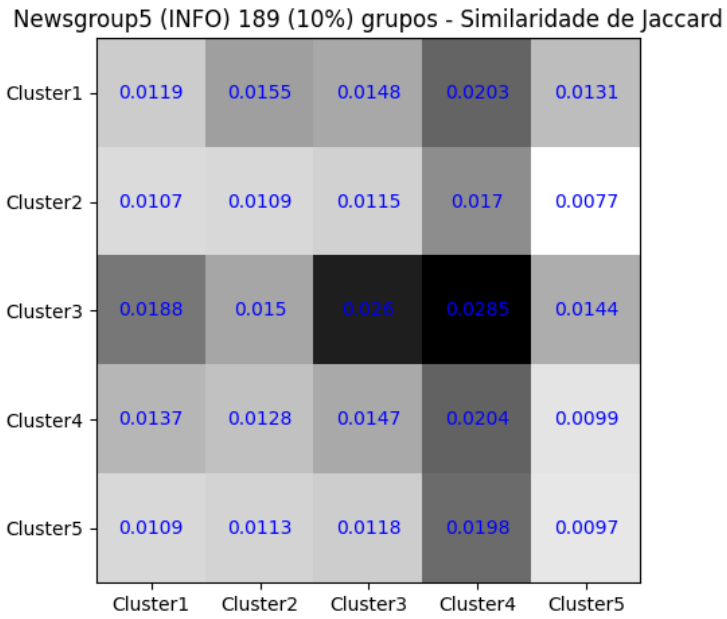
Fonte: Produzido pelo autor

Figura 62 – NG5 - INFO co-clustering - 5% (94) - Similaridade de Jaccard



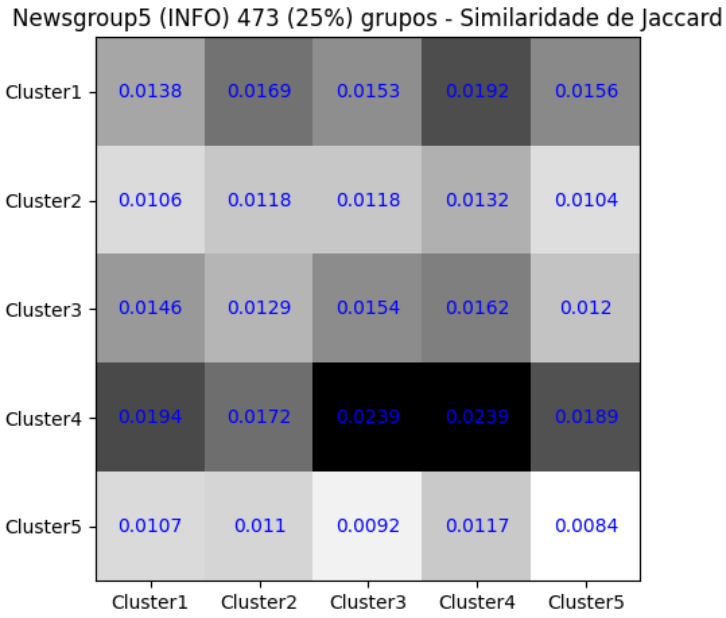
Fonte: Produzido pelo autor

Figura 63 – NG5 - INFO co-clustering - 10% (189) - Similaridade de Jaccard



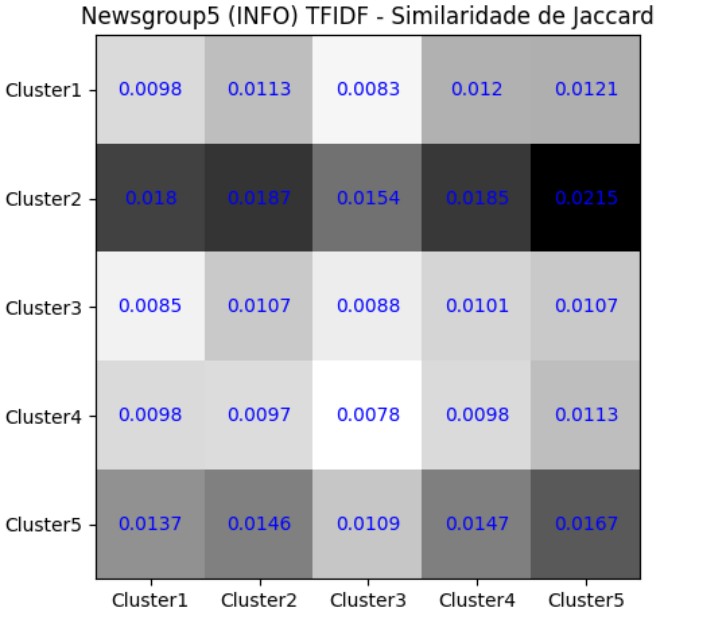
Fonte: Produzido pelo autor

Figura 64 – NG5 - INFO co-clustering - 25% (473) - Similaridade de Jaccard



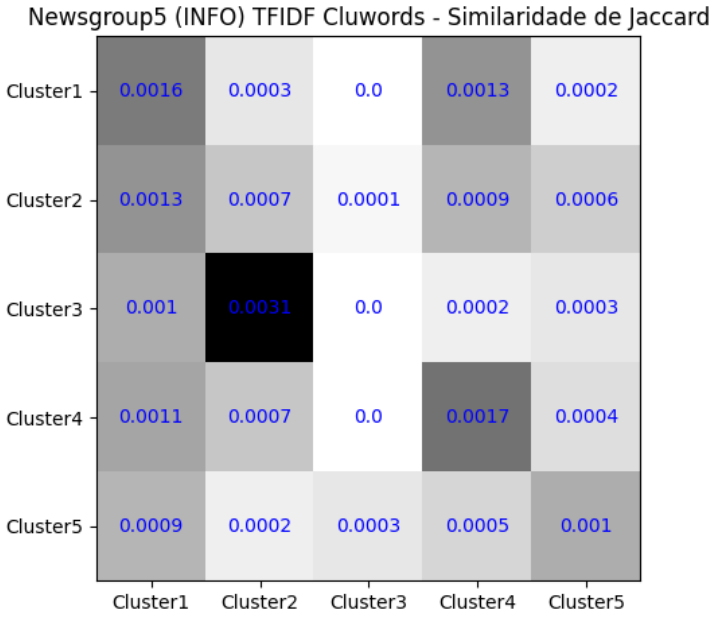
Fonte: Produzido pelo autor

Figura 65 – NG5 - INFO co-clustering - TF-IDF - Similaridade de Jaccard



Fonte: Produzido pelo autor

Figura 66 – NG5 - INFO co-clustering - TF-IDF CluWords - Similaridade de Jaccard



Fonte: Produzido pelo autor

Capítulo 7

Conclusões

Como descrito no Capítulo 1, o objetivo principal deste trabalho foi desenvolver um método eficiente para organização de documentos de texto usando *clusters* de termos e algoritmos de *co-clustering* na representação textual para criar *clusters* de documentos de conteúdo semelhante e com associação representativa com *clusters* de termos além de desenvolver medidas de avaliação eficientes para avaliar a qualidade entre *clusters* de documentos de conteúdo semelhante com associação representativa com *clusters* de termos específicos. Como objetivos específicos, buscou:

– Investigar o uso de *clustering* de termos usando embeddings como forma de diminuir a dimensionalidade e esparsidade das matrizes utilizadas para representação estruturada de documentos;

– Propor uma forma representativa para termos e documentos considerando vetores de word embeddings;

– Propor formas efetivas de avaliar o resultado de algoritmos de *co-clustering*, com ênfase na relação entre *clusters* de documentos e termos representativos desse *cluster* e foco na avaliação de *clusters* de termos e avaliação do ajuste entre *clusters* de termos e *clusters* de documentos.

Foi desenvolvida uma abordagem para organização de documentos que utiliza algoritmos de *co-clustering* após fazer o *clustering* prévio de termos usando seus *embeddings* e gerar uma matriz na forma documentos $\times$ *clusters*, com um cálculo da medida TF-IDF adaptada para *clusters* de termos.

A proposta apresentada neste trabalho permite:

– Manter informações de relações semânticas existentes entre os termos;

– Reduzir a dimensionalidade da matriz de representação;

– Reduzir a esparsidade da matriz de representação.

Entendemos que o uso de relações semânticas no processo de agrupar objetos é de grande importância para produzir *clusters* mais precisos. Para este trabalho, a representação numérica proposta se mostrou eficiente, em alguns casos, quanto à aplicação de algoritmos de *co-clustering* para agrupar simultaneamente documentos e termos.

O número de *clusters* escolhidos para o *clustering* prévio de termos também impacta nos resultados. Considerando a avaliação de *clusters* de documentos e considerando apenas as representações numéricas propostas neste trabalho, percebe-se que é mais comum que os resultados sejam melhores quando o *clustering* prévio é feito com 10%, 25% ou 50% do número de termos, ou seja, matrizes de dados esparsas com relação ao *clustering* prévio feito com o nº ideal de termos e também com 5%. Isso reafirma o que foi dito por Role, Morbieu e Nadif (2019).

Diferentemente do que acontece com os resultados das avaliações feitas nos *clusters* de documentos, para a avaliação dos *clusters* de termos, considerando o índice Davies-Bouldin, a representação numérica do número ideal de *clusters* tende sempre a ter o melhor resultado.

Neste trabalho implementamos uma variante da métrica de (ROLE; MORBIEU; NADIF, 2018), denominada RMN_{GT}, para avaliar *clusters* de termos. A RMN_{GT} é definida como a razão entre a similaridade *intra-cluster* e *inter-cluster* dos termos representativos encontrados por um algoritmo de *co-clustering*, utilizando os word embeddings correspondentes a cada termo. Os modelos propostos neste trabalho apresentaram melhor desempenho considerando a RMN_{GT} em quase todos os casos, justificando o *clustering* prévio de termos e reafirmando nossa hipótese.

Também propomos a métrica $RMN\_GD \times GT$, a qual avalia a correspondência entre um *cluster* de documentos (GD) e um *cluster* de termos (GT) obtidos por um algoritmo de *co-clustering*. A métrica considera apenas os representantes dos *clusters* de termos no cálculo, excluindo os embeddings dos termos que não são representativos. Além disso, no cálculo do representante de documentos, utiliza os valores TF-IDF de cada *cluster* na matriz documentos × *cluster* para ponderar os embeddings dos representantes de *clusters* de termos que aparecem no documento. Essa métrica visa avaliar a similaridade entre as representações vetoriais, expressas por word embeddings, dos *clusters* de documentos e termos. Para essa métrica, o valor será maior quanto maior a associação entre *clusters* de documentos e *clusters* de termos que formam um *co-cluster*, resultado encontrado em *clusters* de termos e *clusters* de documentos associados, provando que existe maior associação em *co-clusters* compostos por *clusters* de documentos e *clusters* de termos associados.

Propomos também uma abordagem utilizada para avaliar a correspondência entre *clusters* de documentos e *clusters* de termos. Para isso, são calculados os coeficientes de Jaccard entre o conjunto de termos de um *co-cluster* e o conjunto de termos de cada

documento desse *co-cluster*. A média desses valores é então calculada. O conjunto de termos de um documento é extraído da matriz documentos × *clusters*, incluindo os termos representantes dos *clusters* com TF-IDF maior que zero nessa matriz. A avaliação se mostrou pouco representativa para os experimentos realizados.

Em resumo foi observado que, considerando os experimentos em geral, o algoritmo que obteve melhor desempenho para os modelos propostos foi o Spectral Co-clustering. Com relação as coleções de documentos onde os modelos superaram de alguma forma TF-IDF sem *clusters* e TF-IDF Cluwords, como CSTR e Sports, são coleções difíceis para a tarefa de *clustering*. Isto significa que existem *clusters* com objetos sobrepostos. Para estas coleções a aplicação do *clustering* prévio de termos se mostrou vantajosa, tornando o *clustering* de documentos mais efetivo e, portanto, obtendo melhor desempenho quando comparado com TF-IDF sem *clusters* e TF-IDF Cluwords.

Considerando os experimentos, avaliações e análises realizadas neste trabalho, em suma, se conclui que os modelos propostos obtêm desempenho superior na tarefa de *clustering* quando a coleção de documentos é considerada difícil para esta tarefa, o que se deve ao *clustering* prévio de termos.

Contribuições

De acordo com os objetivos apresentados no trabalho é possível citar as seguintes contribuições:

– Proposta de geração da matriz da forma documentos × *clusters* com cálculo estendido da medida TF-IDF, com base no *clustering* prévio de termos do vocabulário da coleção usando *word embeddings*;

– Proposta de adaptações de métricas encontradas na literatura para avaliar os *clusters* de termos separadamente e o ajuste entre *clusters* de documentos e *clusters* de termos encontrados pelo *co-clustering*;

– Conjunto extensivo de experimentos em que são explorados três tipos de avaliações: qualidade de *clusters* de documentos, qualidade de *clusters* de termos e ajuste entre *clusters* de documentos e *clusters* de termos.

Trabalhos futuros

Como trabalhos futuros, pretende-se:

– Realizar novos experimentos considerando diferentes estratégias de pré-processamento. Para o trabalho apresentado aqui as mesmas estratégias de pré-processamento foram aplicadas a todos as coleções de documentos. Role, Morbieu e Nadif (2019) cita que,

diferentemente do K-means, algoritmos de *co-clustering* apresentam melhor desempenho com dados esparsos e com alta dimensionalidade. Partindo desse pressuposto, realizar experimentos considerando duas estratégias de pré-processamento: a primeira aplicando apenas a remoção de símbolos e caracteres especiais; e a segunda aplicando todas as tarefas de pré-processamento já utilizadas neste trabalho, além da lematização e a redução do vocabulário das coleções, considerando utilizar apenas termos mais relevantes;

- Como apresentado neste trabalho, para definir o TF-IDF de cada *cluster* para um documento é feita a soma dos valores TF-IDF de todos os termos daquele documento que pertence ao *cluster*. Como trabalho futuro seria interessante investigar novas formas de construir/calcular a matriz de representações numéricas de documentos e termos;
- Investigar casos que, considerando as avaliações apresentadas, os resultados foram muito abaixo em comparação ao TF-IDF e/ou TF-IDF Cluwords;
- Investigar a aplicabilidade da proposta em outras tarefas, como *clustering* e classificação.

Referências

AFFELDT, S.; LABIOD, L.; NADIF, M. Ensemble block co-clustering: a unified framework for text data. In: **Proceedings of the 29th ACM International Conference on Information & Knowledge Management**. [S.l.: s.n.], 2020. p. 5–14.

AFFELDT, S.; LABIOD, L.; NADIF, M. Regularized bi-directional co-clustering. **Statistics and Computing**, Springer, v. 31, n. 3, p. 1–17, 2021.

AGGARWAL, C. C. **Machine learning for text**. [S.l.]: Springer, 2018.

AGGARWAL, C. C.; HINNEBURG, A.; KEIM, D. A. On the surprising behavior of distance metrics in high dimensional space. In: SPRINGER. **International conference on database theory**. [S.l.], 2001. p. 420–434.

AILEM, M.; ROLE, F.; NADIF, M. Co-clustering document-term matrices by direct maximization of graph modularity. In: **Proceedings of the 24th ACM international on conference on information and knowledge management**. [S.l.: s.n.], 2015. p. 1807–1810.

AILEM, M.; ROLE, F.; NADIF, M. Graph modularity maximization as an effective method for co-clustering text data. **Knowledge-Based Systems**, Elsevier, v. 109, p. 160–173, 2016.

AILEM, M.; ROLE, F.; NADIF, M. Sparse poisson latent block model for document clustering. **IEEE Transactions on Knowledge and Data Engineering**, v. 29, n. 7, p. 1563–1576, 2017.

ALGULIYEV, R. M. et al. Cosum: Text summarization based on clustering and optimization. **Expert Systems**, Wiley Online Library, v. 36, n. 1, p. e12340, 2019.

ALMEIDA, F.; XEXÉO, G. Word embeddings: A survey. **arXiv preprint arXiv:1901.09069**, 2019.

BELLOT, P.; EL-BÈZE, M. A clustering method for information retrieval. **Technical ReportIR-0199, Laboratoire d'Informatique d'Avignon, France**, 1999.

BERTENS, P. et al. A machine-learning item recommendation system for video games. In: IEEE. **2018 IEEE Conference on Computational Intelligence and Games (CIG)**. [S.l.], 2018. p. 1–4.

BEZDEK, J. C. et al. **Fuzzy models and algorithms for pattern recognition and image processing**. [S.l.]: Springer Science & Business Media, 1999. v. 4.

BHATIA, P. S.; IOVLEFF, S.; GOVAERT, G. blockcluster: An r package for model-based co-clustering. **Journal of Statistical Software**, v. 76, n. 1, p. 1–24, 2017.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural language processing with Python: analyzing text with the natural language toolkit**. [S.l.]: "O'Reilly Media, Inc.", 2009.

BITON, D.; KALECH, M.; ROKACH, L. Fscoal—parallel simultaneous fuzzy co-clustering and learning. **International Journal of Intelligent Systems**, Wiley Online Library, v. 33, n. 7, p. 1364–1380, 2018.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 5, p. 135–146, 2017.

BÜTTCHER, S.; CLARKE, C. L.; CORMACK, G. V. **Information retrieval: Implementing and evaluating search engines**. [S.l.]: Mit Press, 2016.

CHATFIELD, C.; COLLINS, A. J. **Introduction to multivariate analysis**. [S.l.]: Routledge, 2018.

CHO, H. et al. Minimum sum-squared residue co-clustering of gene expression data. In: SIAM. **Proceedings of the 2004 SIAM international conference on data mining**. [S.l.], 2004. p. 114–125.

CRAW, S. **Manhattan Distance.** 2010.

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

DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. **arXiv preprint arXiv:1810.04805**, 2018.

DHAR, A. et al. Text categorization: past and present. **Artificial Intelligence Review**, Springer, v. 54, n. 4, p. 3007–3054, 2021.

DHILLON, I. S. Co-clustering documents and words using bipartite spectral graph partitioning. In: **Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2001. p. 269–274.

DHILLON, I. S.; MALLELA, S.; MODHA, D. S. Information-theoretic co-clustering. In: **Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2003. p. 89–98.

DHILLON, I. S.; MODHA, D. S. Concept decompositions for large sparse text data using clustering. **Machine learning**, Springer, v. 42, n. 1, p. 143–175, 2001.

DIAZ, A. K. R.; PERES, S. M. Biclustering and coclustering: concepts, algorithms and viability for text mining. **Revista de Informática Teórica e Aplicada**, v. 26, n. 2, p. 81–117, 2019.

DIENG, A. B.; RUIZ, F. J.; BLEI, D. M. Topic modeling in embedding spaces. **Transactions of the Association for Computational Linguistics**, MIT Press, v. 8, p. 439–453, 2020.

ESTER, M. et al. Density-based spatial clustering of applications with noise (dbscan). In: **Proc. of the Second International Conference on Knowledge Discovery and Data Mining**. [S.l.: s.n.], 1996. p. 226–231.

FACELI, K. et al. **Inteligência artificial: uma abordagem de aprendizado de máquina**. [S.l.]: LTC, 2011.

FRANÇA, F. O. de. A hash-based co-clustering algorithm for categorical data. **Expert Systems with Applications**, Elsevier, v. 64, p. 24–35, 2016.

GARCIA-DIAS, R. et al. Clustering analysis. In: **machine learning**. [S.l.]: Elsevier, 2020. p. 227–247.

GOVAERT, G.; NADIF, M. Clustering with block mixture models. **Pattern Recognition**, Elsevier, v. 36, n. 2, p. 463–473, 2003.

GOVAERT, G.; NADIF, M. **Co-clustering: models, algorithms and applications**. [S.l.]: John Wiley & Sons, 2013.

GUANGCE, R.; LEI, X. Knowledge discovery of news text based on artificial intelligence. **ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)**, ACM New York, NY, USA, v. 20, n. 1, p. 1–18, 2020.

HAJJ, N.; RIZK, Y.; AWAD, M. A subjectivity classification framework for sports articles using improved cortical algorithms. **Neural Computing and Applications**, Springer, v. 31, n. 11, p. 8069–8085, 2019.

HALAVAIS, A. **Search engine society**. [S.l.]: John Wiley & Sons, 2017.

HANDL, J.; KNOWLES, J.; KELL, D. B. Computational cluster validation in post-genomic data analysis. **Bioinformatics**, Oxford University Press, v. 21, n. 15, p. 3201–3212, 2005.

HARRINGTON, P. **Machine learning in action**. [S.l.]: Simon and Schuster, 2012.

HARRIS, C. R. et al. Array programming with NumPy. **Nature**, Springer Science and Business Media LLC, v. 585, n. 7825, p. 357–362, set. 2020. Disponível em: <https://doi.org/10.1038/s41586-020-2649-2>.

HARRIS, Z. S. Distributional structure. **Word**, Taylor & Francis, v. 10, n. 2-3, p. 146–162, 1954.

HENRIQUES, R.; MADEIRA, S. C. Bsig: evaluating the statistical significance of biclustering solutions. **Data Mining and Knowledge Discovery**, Springer, v. 32, n. 1, p. 124–161, 2018.

HENRIQUES, R.; MADEIRA, S. C. Flebic: Learning classifiers from high-dimensional biomedical data using discriminative biclusters with non-constant patterns. **Pattern Recognition**, Elsevier, v. 115, p. 107900, 2021.

HINTON, G. E. et al. Learning distributed representations of concepts. In: AMHERST, MA. **Proceedings of the eighth annual conference of the cognitive science society**. [S.l.], 1986. v. 1, p. 12.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HONDA, K. et al. Mmms-induced k-member co-clustering for k-anonymization of cooccurrence information. In: IEEE. **2016 International Joint Conference on Neural Networks (IJCNN)**. [S.l.], 2016. p. 2961–2966.

HUANG, S.; XU, Z.; LV, J. Adaptive local structure learning for document co-clustering. **Knowledge-Based Systems**, Elsevier, v. 148, p. 74–84, 2018.

HUANG, S. et al. Auto-weighted multi-view co-clustering with bipartite graphs. **Information Sciences**, Elsevier, v. 512, p. 18–30, 2020.

HUBERT, L.; ARABIE, P. Comparing partitions. **Journal of classification**, Springer, v. 2, n. 1, p. 193–218, 1985.

JACCARD, P. Étude comparative de la distribution florale dans une portion des alpes et des jura. **Bull Soc Vaudoise Sci Nat**, v. 37, p. 547–579, 1901.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern recognition letters**, Elsevier, v. 31, n. 8, p. 651–666, 2010.

JANG, B. et al. Bi-lstm model to increase accuracy in text classification: Combining word2vec cnn and attention mechanism. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 10, n. 17, p. 5841, 2020.

JATNIKA, D.; BIJAKSANA, M. A.; SURYANI, A. A. Word2vec model analysis for semantic similarities in english words. **Procedia Computer Science**, Elsevier, v. 157, p. 160–167, 2019.

JOACHIMS, T. **A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization.** [S.l.], 1996.

JOSHI, P. A step-by-step nlp guide to learn elmo for extracting features from text. 2019.

JOULIN, A. et al. Bag of tricks for efficient text classification. **arXiv preprint arXiv:1607.01759**, 2016.

KLUGER, Y. et al. Spectral biclustering of microarray data: coclustering genes and conditions. **Genome research**, Cold Spring Harbor Lab, v. 13, n. 4, p. 703–716, 2003.

KRZANOWSKI, W. J.; LAI, Y. A criterion for determining the number of groups in a data set using sum-of-squares clustering. **Biometrics**, JSTOR, p. 23–34, 1988.

KULKARNI, A.; SHIVANANDA, A. Converting text to features. In: **Natural language processing recipes**. [S.l.]: Springer, 2021. p. 63–106.

KUMMAMURU, K.; DHAWALE, A.; KRISHNAPURAM, R. Fuzzy co-clustering of documents and keywords. In: IEEE. **The 12th IEEE International Conference on Fuzzy Systems, 2003. FUZZ'03.** [S.l.], 2003. v. 2, p. 772–777.

KYRIAKOPOULOU, A.; KALAMBOUKIS, T. Text classification using clustering. In: **Proceedings of the Discovery Challenge Workshop at ECML/PKDD 2006**. [S.l.: s.n.], 2006. p. 28–38.

LABIOD, L.; NADIF, M. Co-clustering for binary and categorical data with maximum modularity. In: IEEE. **2011 IEEE 11th international conference on data mining**. [S.l.], 2011. p. 1140–1145.

LACLAU, C.; NADIF, M. Hard and fuzzy diagonal co-clustering for document-term partitioning. **Neurocomputing**, Elsevier, v. 193, p. 133–147, 2016.

LE, Q.; MIKOLOV, T. Distributed representations of sentences and documents. In: PMLR. **International conference on machine learning**. [S.l.], 2014. p. 1188–1196.

LEWIS, D. D. et al. Rcv1: A new benchmark collection for text categorization research. **Journal of machine learning research**, Goldsmiths, University of London, v. 5, n. Apr, p. 361–397, 2004.

LI, Y. H.; JAIN, A. K. Classification of text documents. **The Computer Journal**, Oxford University Press, v. 41, n. 8, p. 537–546, 1998.

LIU, J.; HAN, J. Spectral clustering. In: **Data Clustering**. [S.l.]: Chapman and Hall/CRC, 2018. p. 177–200.

LIU, L.; ÖZSU, M. T. **Encyclopedia of database systems**. [S.l.]: Springer, 2009. v. 6.

LIU, Y.; CHEN, J.; CHAO, H. A fuzzy co-clustering algorithm via modularity maximization. **Mathematical Problems in Engineering**, Hindawi, v. 2018, 2018.

LUHN, H. P. A statistical approach to mechanized encoding and searching of literary information. **IBM Journal of research and development**, IBM, v. 1, n. 4, p. 309–317, 1957.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. **Proceedings of the fifth Berkeley symposium on mathematical statistics and probability**. [S.l.], 1967. v. 1, n. 14, p. 281–297.

MADEIRA. **(Clustering and) Biclustering gene expression data**. 2011. <http://cw.fel.cvut.cz/wiki/_media/courses/a6m33bin/biclustering.pdf>.

MADEIRA, S. C.; OLIVEIRA, A. L. Biclustering algorithms for biological data analysis: a survey. **IEEE/ACM transactions on computational biology and bioinformatics**, IEEE, v. 1, n. 1, p. 24–45, 2004.

MEI, J.-P. et al. Large scale document categorization with fuzzy clustering. **IEEE Transactions on Fuzzy Systems**, IEEE, v. 25, n. 5, p. 1239–1251, 2016.

MENON, T. Empirical analysis of cbow and skip gram nlp models. 2020.

MIKOLOV, T. et al. Efficient estimation of word representations in vector space. **arXiv preprint arXiv:1301.3781**, 2013.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: **Advances in neural information processing systems**. [S.l.: s.n.], 2013. p. 3111–3119.

MINER, G. et al. **Practical text mining and statistical analysis for non-structured text data applications**. [S.l.]: Academic Press, 2012.

MITCHELL, T. Machine learning. McGraw hill Burr Ridge, 1997.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, Manole Ltda, v. 1, n. 1, p. 32, 2003.

MORAL, C. et al. A survey of stemming algorithms in information retrieval. **Information Research: An International Electronic Journal**, ERIC, v. 19, n. 1, p. n1, 2014.

NEVES, F. et al. Mining actionable patterns of road mobility from heterogeneous traffic data using biclustering. **IEEE Transactions on Intelligent Transportation Systems**, IEEE, 2021.

NIE, F. et al. Fast clustering with co-clustering via discrete non-negative matrix factorization for image identification. In: IEEE. **ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.], 2020. p. 2073–2077.

OH, C.-H.; HONDA, K.; ICHIHASHI, H. Fuzzy clustering for categorical multivariate data. In: IEEE. **Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference (Cat. No. 01TH8569)**. [S.l.], 2001. v. 4, p. 2154–2159.

PAPADIMITRIOU, S.; SUN, J. Disco: Distributed co-clustering with map-reduce: A case study towards petabyte-scale end-to-end mining. In: IEEE. **2008 Eighth IEEE International Conference on Data Mining**. [S.l.], 2008. p. 512–521.

PAUL, R.; HOQUE, A. S. M. L. Clustering medical data to predict the likelihood of diseases. In: IEEE. **2010 fifth international conference on digital information management (ICDIM)**. [S.l.], 2010. p. 44–49.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. **Journal of Machine Learning Research**, v. 12, n. 85, p. 2825–2830, 2011. Disponível em: <http://jmlr.org/papers/v12/pedregosa11a.html>.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: **Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)**. [S.l.: s.n.], 2014. p. 1532–1543.

PENSA, R. G. et al. Co-clustering numerical data under user-defined constraints. **Statistical Analysis and Data Mining: The ASA Data Science Journal**, Wiley Online Library, v. 3, n. 1, p. 38–55, 2010.

PETERS, M. et al. Deep contextualized word representations. arxiv 2018. **arXiv preprint arXiv:1802.05365**, v. 12, 2018.

PORTER, M. F. An algorithm for suffix stripping. **Program**, MCB UP Ltd, 1980.

REDDY, C. K.; VINZAMURI, B. A survey of partitional and hierarchical clustering algorithms. In: **Data clustering**. [S.l.]: Chapman and Hall/CRC, 2018. p. 87–110.

REHUREK, R.; SOJKA, P. Gensim–python framework for vector space modelling. **NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic**, v. 3, n. 2, 2011.

ROIGER, R. J. **Data mining: a tutorial-based primer**. [S.l.]: Chapman and Hall/CRC, 2017.

ROLE, F.; MORBIEU, S.; NADIF, M. Unsupervised evaluation of text co-clustering algorithms using neural word embeddings. In: **Proceedings of the 27th ACM International Conference on Information and Knowledge Management**. [S.l.: s.n.], 2018. p. 1827–1830.

ROLE, F.; MORBIEU, S.; NADIF, M. Coclust: a python package for co-clustering. **Journal of Statistical Software**, v. 88, n. 1, p. 1–29, 2019.

ROSSI, R. G. et al. Benchmarking text collections for classification and clustering tasks. São Carlos, SP, Brasil., 2013.

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

ROY, S.; CHAKRABARTI, A. Chapter 11 - a novel graph clustering algorithm based on discrete-time quantum random walk. In: BHATTACHARYYA, S.; MAULIK, U.; DUTTA, P. (Ed.). **Quantum Inspired Computational Intelligence**. Boston: Morgan Kaufmann, 2017. p. 361–389. ISBN 978-0-12-804409-4. Disponível em: <https://www.sciencedirect.com/science/article/pii/B9780128044094000115>.

RUSSELL, S.; NORVIG, P. Artificial intelligence: a modern approach. 2002.

SALAH, A.; AILEM, M.; NADIF, M. Word co-occurrence regularized non-negative matrix tri-factorization for text data co-clustering. In: **Thirty-Second AAAI Conference on Artificial Intelligence**. [S.l.: s.n.], 2018.

SALAH, A.; ROGOVSCHI, N.; NADIF, M. Stochastic co-clustering for document-term data. In: SIAM. **Proceedings of the 2016 SIAM International Conference on Data Mining**. [S.l.], 2016. p. 306–314.

SANTOS, J. M.; EMBRECHTS, M. On the use of the adjusted rand index as a metric for evaluating supervised classification. In: SPRINGER. **International conference on artificial neural networks**. [S.l.], 2009. p. 175–184.

SEBASTIANI, F. Machine learning in automated text categorization. **ACM computing surveys (CSUR)**, ACM New York, NY, USA, v. 34, n. 1, p. 1–47, 2002.

SELOSSE, M.; JACQUES, J.; BIERNACKI, C. Textual data summarization using the self-organized co-clustering model. **Pattern Recognition**, Elsevier, v. 103, p. 107315, 2020.

SILVA, L. d.; PERES, S.; BOSCARIOLI, C. **Introdução À Mineração de Dados-Com Aplicação Em R**. [S.l.]: São Paulo, Brasil: Elsevier Editora Ltda, 2016.

SINGH, R. H. et al. Movie recommendation system using cosine similarity and knn. **International Journal of Engineering and Advanced Technology**, v. 9, n. 5, p. 556–559, 2020.

SONG, J. **Big Data Analysis Using Machine Learning for Social Scientists and Criminologists**. [S.l.]: Cambridge Scholars Publishing, 2019.

TAN, X. et al. Multilingual neural machine translation with language clustering. **arXiv preprint arXiv:1908.09324**, 2019.

TANDEL, N. H.; PRAJAPATI, H. B.; DABHI, V. K. Voice recognition and voice comparison using machine learning techniques: A survey. In: IEEE. **2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)**. [S.l.], 2020. p. 459–465.

TAYLOR, J. **Introduction to error analysis, the study of uncertainties in physical measurements**. [S.l.: s.n.], 1997.

TEAM, T. pandas development. **pandas-dev/pandas: Pandas**. Zenodo, 2020. Disponível em: <https://doi.org/10.5281/zenodo.3509134>.

THANH, N. D.; ALI, M.; SON, L. H. A novel clustering algorithm in a neutrosophic recommender system for medical diagnosis. **Cognitive computation**, Springer, v. 9, n. 4, p. 526–544, 2017.

THONG, N. T. et al. Intuitionistic fuzzy recommender systems: an effective tool for medical diagnosis. **Knowledge-Based Systems**, Elsevier, v. 74, p. 133–150, 2015.

THORNDIKE, R. L. Who belongs in the family? **Psychometrika**, Springer, v. 18, n. 4, p. 267–276, 1953.

TUNALI, V. **Data Mining Research, Classic3 and Classic4 DataSets**. [S.l.], 2010.

VARMEDJA, D. et al. Credit card fraud detection-machine learning methods. In: IEEE. **2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)**. [S.l.], 2019. p. 1–5.

VASWANI, A. et al. Attention is all you need. **Advances in neural information processing systems**, v. 30, 2017.

VIEGAS, F. et al. Cluwords: exploiting semantic word clustering representation for enhanced topic modeling. In: **Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining**. [S.l.: s.n.], 2019. p. 753–761.

VIJAYARANI, S. et al. Preprocessing techniques for text mining-an overview. **International Journal of Computer Science & Communication Networks**, v. 5, n. 1, p. 7–16, 2015.

VINAYKUMAR, K.; RAJAVELU, S.; BLESSING, R. E. Similarity measures and text documents classfication accuracies using benchmark datasets. In: IEEE. **2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)**. [S.l.], 2020. p. 688–695.

WARTENA, C.; BRUSSEE, R. Topic detection by clustering keywords. In: IEEE. **2008 19th international workshop on database and expert systems applications**. [S.l.], 2008. p. 54–58.

WITTEN, I. H. et al. Practical machine learning tools and techniques. In: **DATA MINING**. [S.l.: s.n.], 2005. v. 2, p. 4.

XU, R.; WUNSCH, D. **Clustering**. [S.l.]: John Wiley & Sons, 2008. v. 10.

XUAN, J. et al. Doubly nonparametric sparse nonnegative matrix factorization based on dependent indian buffet processes. **IEEE transactions on neural networks and learning systems**, IEEE, v. 29, n. 5, p. 1835–1849, 2017.

YAN, Y.; CHEN, L.; TJHI, W.-C. Fuzzy semi-supervised co-clustering for text documents. **Fuzzy Sets and Systems**, Elsevier, v. 215, p. 74–89, 2013.

ZHAI, C.; MASSUNG, S. **Text data management and analysis: a practical introduction to information retrieval and text mining**. [S.l.]: Morgan & Claypool, 2016.

ZHANG, J. et al. Co-adjustment learning for co-clustering. **Cognitive Computation**, Springer, v. 13, n. 2, p. 504–517, 2021.

ZHANG, W.; LI, Y.; WANG, S. Learning document representation via topic-enhanced lstm model. **Knowledge-Based Systems**, Elsevier, v. 174, p. 194–204, 2019.

Apêndices

APÊNDICE A

Tabelas de avaliação de clustering de documentos

Tabela 55 – Classic4 - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1924	0.3119	0.3742	0.4462	0.4733	**0.6069**	0.4715
ARI	0.1272	0.2383	0.2856	0.3259	0.3403	**0.4852**	0.3300
ACC	0.4523	0.5815	0.6150	0.6470	0.6507	**0.7862**	0.6335

Fonte: Produzido pelo autor.

Tabela 56 – Classic4 - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1024	0.2277	0.3203	0.4049	0.4268	**0.5860**	0.3513
ARI	0.1297	0.1669	0.2352	0.3534	0.3641	**0.5144**	0.2601
ACC	0.4658	0.5005	0.5837	0.3534	0.7095	**0.8047**	0.5642

Fonte: Produzido pelo autor.

Tabela 57 – Classic4 - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.3899	**0.5490**	0.3560	0.3790	0.0089	0.1235	0.3937
ARI	0.3310	**0.4533**	0.1526	0.1604	0.0001	-0.0478	0.2204
ACC	0.5700	**0.7147**	0.5528	0.5571	0.4513	0.3602	0.5579

Fonte: Produzido pelo autor.

Tabela 58 – Classic4 - Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0855	0.3179	0.3765	0.2658	0.4423	**0.5647**	0.4904
ARI	0.0716	0.2406	0.3079	0.1938	0.3228	**0.4259**	0.3462
ACC	0.3790	0.5837	**0.6687**	0.5414	0.6487	0.6425	0.6051

Fonte: Produzido pelo autor.

Tabela 59 – Newsgroup5 - Avaliação de clusters de documentos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1518	0.1282	0.0730	0.1060	0.1930	**0.4694**	0.1887
ARI	0.1147	0.0909	0.0451	0.0658	0.14801	**0.4834**	0.1353
ACC	0.4030	0.3925	0.3192	0.3415	0.4487	**0.7509**	0.3906

Fonte: Produzido pelo autor.

Tabela 60 – Newsgroup5 - Avaliação de clusters de documentos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.1061	0.0644	0.1045	0.1543	0.1770	**0.4539**	0.1747
ARI	0.0974	0.0544	0.0935	0.1498	0.1736	**0.4766**	0.1062
ACC	0.3957	0.3481	0.3964	0.4415	0.4836	**0.7464**	0.3902

Fonte: Produzido pelo autor.

Tabela 61 – Newsgroup5 - Avaliação de clusters de documentos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.2907	0.1888	0.2013	0.0153	0.0139	**0.4536**	0.1241
ARI	0.1612	0.0600	0.0437	0.0077	0.0011	**0.2847**	0.0447
ACC	0.4353	0.3153	0.3407	0.2050	0.2043	**0.5957**	0.3362

Fonte: Produzido pelo autor.

Tabela 62 – Newsgroup5 - Avaliação de clusters de documentos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
NMI	0.0942	0.1143	0.1504	0.1618	0.1037	**0.5206**	0.1967
ARI	0.0904	0.0945	0.1351	0.1485	0.0831	**0.5318**	0.1295
ACC	0.3449	0.3977	0.4455	0.4459	0.3592	**0.7817**	0.4529

Fonte: Produzido pelo autor.

Apêndice B

Tabelas de avaliação de clustering de termos

Tabela 63 – Classic3 - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.9103	0.9954	0.6731	**1.0004**	0.8951	**1.0004**	1.0013
DB	1.9992	3.8281	4.3548	**5.1618**	5.1289	4.8054	4.8331

Fonte: Produzido pelo autor.

Tabela 64 – Classic3 - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.9103	0.9954	0.6731	1.0004	0.8951	**1.0005**	1.0004
DB	1.9992	3.8281	4.3548	**5.1618**	5.1289	4.8006	4.3792

Fonte: Produzido pelo autor.

Tabela 65 – Classic3 - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.9103	0.9954	0.6731	1.0004	0.8951	1.0005	**1.0012**
DB	1.9992	3.8281	4.3548	**5.1618**	5.1289	4.8025	4.6773

Fonte: Produzido pelo autor.

Tabela 66 – Classic3 - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.9627	0.9974	0.9984	0.9998	1.0002	1.0006	**1.0013**
DB	2.2512	3.9907	4.3586	**5.1830**	5.1274	4.8011	4.8177

Fonte: Produzido pelo autor.

Tabela 67 – Classic4 - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	**0.6698**	0.6674	0.6655	0.6657	0.6665	0.6665	0.6666
DB	2.2853	3.8802	4.0981	4.6629	**4.7874**	4.5006	4.7801

Fonte: Produzido pelo autor.

Tabela 68 – Classic4 - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.6621	0.6645	0.6658	0.6658	0.6664	**0.6666**	**0.6666**
DB	2.1178	3.3989	4.0518	4.6168	4.7753	4.5026	**4.7789**

Fonte: Produzido pelo autor.

Tabela 69 – Classic4 - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.6833	0.4992	**0.7302**	0.6113	0.6778	0.5871	0.6653
DB	2.1483	3.5569	4.0014	4.6634	**4.8019**	4.5037	4.7807

Fonte: Produzido pelo autor.

Tabela 70 – Classic4 - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	**0.7046**	0.6689	0.6658	0.6672	0.6673	0.6665	0.6656
DB	1.7668	3.5928	4.0698	4.6635	4.7789	4.5080	**4.7799**

Fonte: Produzido pelo autor.

Tabela 71 – CSTR - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	**0.6787**	0.6692	0.6684	0.6672	0.6664	0.6666	0.6667
DB	2.0960	3.8323	4.1215	4.4336	**4.8224**	4.5953	4.6600

Fonte: Produzido pelo autor.

Tabela 72 – CSTR - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.6410	0.6674	0.6664	**0.6679**	0.6665	0.6667	0.6663
DB	2.0855	3.5917	4.0046	4.6866	**4.8158**	4.6030	4.6683

Fonte: Produzido pelo autor.

Tabela 73 – CSTR - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.3006	0.6225	0.6387	0.6717	**0.6776**	0.6499	0.6665
DB	2.1043	3.7238	4.1413	4.7019	**4.8199**	4.5999	4.6542

Fonte: Produzido pelo autor.

Tabela 74 – CSTR - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.6414	0.6677	0.6655	0.6676	0.6665	**0.6704**	0.6668
DB	2.1350	3.6727	4.1490	4.7239	**4.8012**	4.5961	4.6377

Fonte: Produzido pelo autor.

Tabela 75 – Reuters8 - Avaliação de clusters de termos - INFO co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.1484	**0.2866**	0.2862	0.2855	0.2860	0.2856	0.2856
DB	1.1619	3.0944	3.5212	4.4235	4.4468	**4.6401**	4.3725

Fonte: Produzido pelo autor.

Tabela 76 – Reuters8 - Avaliação de clusters de termos - Block co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.2195	**0.2877**	0.2854	0.2857	0.2856	0.2858	0.2855
DB	1.1729	3.1040	4.0008	4.4555	4.5824	**4.6372**	4.3219

Fonte: Produzido pelo autor.

Tabela 77 – Reuters8 - Avaliação de clusters de termos - Spectral co-clustering.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.1234	0.2894	0.2873	**0.3300**	0.2993	0.2801	0.2857
DB	1.2011	3.1341	3.5380	3.9954	**5.0582**	4.6374	4.3701

Fonte: Produzido pelo autor.

Tabela 78 – Reuters8 - Avaliação de clusters de termos - Co-clustering fuzzy.

	Nº ideal	5%	10%	25%	50%	Sem clusters	CluWords
Role2018	0.1489	0.2839	0.2855	0.2854	**0.2856**	0.2585	0.2853
DB	1.1756	2.9819	4.0394	4.4076	**5.0294**	4.8239	4.4097

Fonte: Produzido pelo autor.