Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Análise de Textos Históricos utilizando LLMs e Modelagem de Tópicos

João Pedro Lukasavicus Silva

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisor: Mateus Espadoto

São Paulo

Esta seção é opcional e fica numa página separada;
ela pode ser usada para uma dedicatória ou epígrafe.

Agradecimentos

Do. Or do not. There is no try.
— Mestre Yoda

Texto texto. Texto opcional.

Resumo

João Pedro Lukasavicus Silva. **Análise de Textos Históricos utilizando LLMs e Modelagem de Tópicos**. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo.

Elemento obrigatório, constituído de uma sequência de frases concisas e objetivas, em forma de texto. Deve apresentar os objetivos, métodos empregados, resultados e conclusões. O resumo deve ser redigido em parágrafo único, conter no máximo 500 palavras e ser seguido dos termos representativos do conteúdo do trabalho (palavras-chave). Deve ser precedido da referência do documento. Texto texto. Texto texto.

Palavras-chave: Redes neurais. LLM. Grandes modelos de linguagem. Modelagem de tópicos. Textos históricos. Isidoro de Sevilha. Etimologias.

Abstract

João Pedro Lukasavicus Silva. **Analysis of Historical Texts using LLMs and Topic Modeling: *a subtitle***. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo.

Elemento obrigatório, elaborado com as mesmas características do resumo em língua portuguesa. De acordo com o Regimento da Pós-Graduação da USP (Artigo 99), deve ser redigido em inglês para fins de divulgação. É uma boa ideia usar o sítio www.grammarly.com na preparação de textos em inglês. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text. Text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text text.

Keywords: Neural Networks. LLM. Large Language Models. Topic Modeling. Historical Texts. Isidore of Seville. Etymologies.

Lista de abreviaturas

LLM	Large Language Model
MLP	Multi-layer Perceptron
PDF	Portable Document Format
HTML	Hyper-text Markup Language
NLP	Natural Language Processing
URL	Localizador Uniforme de Recursos (*Uniform Resource Locator*)
IME	Instituto de Matemática e Estatística
USP	Universidade de São Paulo

Lista de figuras

Lista de tabelas

Lista de programas

Sumário

Capítulo 1

Introdução

Tradicionalmente, pesquisadores das áreas de humanidades como história, filosofia, entre outras, dependem da análise de grandes quantidades de fontes para a realização do seu trabalho. Esta análise tipicamente é realizada de forma manual, e com grande custo, tanto em termos de tempo quanto de recursos humanos, custo este que pode se tornar um fator limitador da produção científica desses pesquisadores.

A área de processamento de linguagem natural (NLP) possui métodos bem estabelecidos para análise semântica de textos, como por exemplo, modelagem de tópicos (Deerwester *et al.*, 1990; David M Blei *et al.*, 2003; Jelodar *et al.*, 2019), que visa agrupar textos que tratam de assuntos semelhantes. Com o surgimento dos grandes modelos de linguagem (LLM) baseados em transformers (Vaswani *et al.*, 2017; Devlin *et al.*, 2019), é possível observar um grande salto em termos de qualidade das ferramentas e métodos para análise de texto, o que pode ser atribuído à maior capacidade de mapeamento de conceitos em um espaço latente de atributos, que, por sua vez, possibilita um melhor agrupamento de textos em termos de semântica. Em todo caso, o uso destas técnicas requer certa familiaridade com o seu funcionamento e com os textos analisados, de modo que sejam feitos os ajustes necessários para a obtenção de bons resultados.

1.1 Objetivos

Neste trabalho pretendemos, por meio do uso de técnicas computacionais de processamento de texto, estudar os tópicos existentes na obra conhecida como *Etimologias*, de Isidoro de Sevilha (c.560-636). Esta obra, que é considerada como a primeira grande enciclopédia da Idade Média, é formada por 20 livros que tratam das origens das palavras agrupadas em diversos grandes temas. O autor, que viveu em uma época de grandes mudanças culturais, buscava registrar o conhecimento de escritores latinos da Antiguidade Clássica, como Varrão, Catão e Plínio o Velho, entre outros. A obra, que foi escrita em latim medieval, foi copiada exaustivamente ao longo de cerca de 700 anos, sendo utilizada como uma espécie de livro-texto básico nas instituições de ensino da época. Por conveniência, para este trabalho será utilizada uma tradução recente para a língua inglesa (Barney *et al.*, 2006).

1.2 Contribuições

A principal contribuição pretendida com este trabalho é conseguir mapear temas transversais discutidos na obra estudada, para além dos temas gerais propostos pelo autor. Adicionalmente, propomos um experimento de análise de similaridade semântica entre a obra estudada e possíveis fontes da antiguidade, para identificar potenciais citações sem atribuição.

1.3 Organização

O texto está organizado da seguinte forma: no Capítulo 2 tratamos do background histórico dos autores estudados, bem como das ferramentas utilizadas; no Capítulo 3 apresentamos os dados analisados, o *pipeline* de processamento criado para os dados, e a configuração dos experimentos realizados; no Capítulo 4 apresentamos e discutimos os resultados obtidos para cada experimento realizado. O Capítulo 5 conclui o texto.

Capítulo 2

Background

Neste trabalho, utilizaremos ferramentas de modelagem de tópicos e grandes modelos de linguagem com o objetivo de demonstrar sua utilidade no estudo de textos históricos. Nas seções a seguir detalhamos as ferramentas utilizadas e as obras a serem estudadas.

2.1 Redes Neurais e Grandes Modelos de Linguagem

Desde o surgimento dos primeiros computadores digitais no período logo após a Segunda Guerra Mundial (1939 - 1945), pesquisadores trabalharam para desenvolver ferramentas que pudessem executar tarefas que até então eram realizadas apenas por humanos. Cientistas como Warren McCulloch (1898 - 1969) e Walter Pitts (1923 - 1969), um médico e um lógico, estão entre os primeiros a discutirem modelos do cérebro humano, e como estes poderiam ser implementados computacionalmente. Em 1943 publicaram um artigo (McCulloch e Pitts, 1943) considerado seminal na área, aonde propuseram um modelo matemático do cérebro representado como uma rede de elementos simples interconectados. Um elemento particular recebe sinais dos elementos conectados à sua entrada, e produzem uma soma ponderada como sinal de saída, que, dependendo de um limiar pré-determinado, é enviado para elementos conectados à sua saída como um sinal ativo (valor 1) ou inativo (valor 0). Os autores demonstram no artigo que é possível calcular funções lógicas utilizando diferentes configurações de conexões entre elementos. Posteriormente, estes elementos passaram a ser conhecidos como neurônios artificiais, que serviram de base para o desenvolvimento das redes neurais artificiais, anos mais tarde.

Alan Turing (1912 - 1954), matemático e teórico da computação que ajudou a estabelecer as bases da ciência da computação como disciplina, foi um dos primeiros a se dedicar ao que veio a ser chamado posteriormente de inteligência artificial, de forma mais abrangente. Turing apresenta (Turing, 1950) um conjunto de questões e definições que seriam essenciais para o desenvolvimento da área. Em primeiro lugar, ele pergunta se "máquinas podem pensar", o que traz o primeiro problema importante, o de falta de definições claras sobre o que é "pensar" e, naquele momento, sobre o que seria a "máquina" em questão. Como definições do que é pensar, e de forma mais abrangente, do que é inteligência, representam problemas filosóficos importantes e sem resposta objetiva, Turing propôs

uma abordagem prática para isso: no lugar de "máquinas podem pensar", ele passa a perguntar se "máquinas podem agir de forma indistinguível de um humano". Em outras palavras, ele ignora o processo interno pelo qual uma ação é produzida, e se preocupa apenas com o efeito da ação por aqueles que a percebem. Para demonstrar esta ideia, Turing propôs um experimento chamado de Jogo da Imitação. Na versão mais simples do jogo, há três participantes: um humano e um computador que devem responder a perguntas de um juiz humano que só pode fazer perguntas e receber respostas escritas em papel, sem contato direto com os outros dois. Se o juiz, após uma série de perguntas, não for capaz de distinguir o humano da máquina, a máquina venceu o jogo, ou seja, conseguiu se passar por um humano. Apesar de ser uma definição prática, simples, e que serve aos propósitos do autor, esta ideia de que basta ser percebido como humano para ser considerado inteligente ou pensante é problemática para pensadores de outras áreas, que tratam de aspectos metafísicos da mente e do pensamento.

Em 1980, o filósofo John Searle (1932 -) propôs o argumento do Quarto Chinês (Searle, 1980). Neste experimento, o autor imagina uma pessoa que não fala chinês isolada em um quarto com um livro contendo instruções sobre como interpretar símbolos chineses. Se alguém colocar um texto em chinês por debaixo da porta, por exemplo, a pessoa poderia seguir as instruções do livro para produzir símbolos que representem uma resposta coerente para falantes de chinês. No entanto, esta pessoa estaria apenas seguindo regras sintáticas sem nenhuma compreensão semântica do texto que está produzindo. O autor busca com isso se contrapor a ideias funcionalistas, como a de Turing, de que a mente é apenas um sistema de processamento de informações. Ou seja, para Searle, não basta ser capaz de se comportar como se entendesse uma conversa para uma entidade ser considerada como pensante ou inteligente.

Apesar das críticas, a busca pela inteligência artificial seguiu com maior ou menor intensidade ao longo da segunda metade do século XX. Por exemplo, partindo da ideia de neurônios artificiais de McCulloch e Pitts em 1943, Frank Rosenblatt (1928 - 1971), psicólogo, desenvolveu o Perceptron (Rosenblatt, 1958) em 1957, que é considerada a primeira rede neural artificial. A ideia do Perceptron foi desenvolvida ao longo dos anos, passando por diversos altos e baixos, e com a evolução do hardware existente, cada vez mais capaz, culminou na criação do que passou a ser chamado de Deep Learning (Krizhevsky *et al.*, 2012), por volta de 2012 com o trabalho de Alex Krizhevsky (1986 -), que consiste no uso de grandes redes neurais com milhões e até bilhões de elementos aplicado a problemas de processamento de texto e imagem. Mais recentemente, por volta de 2017, a partir do trabalho de Ashish Vaswani (1986 -) e outros foram criadas novas arquiteturas de conexão entre os neurônios artificiais, batizadas de Transformers (Vaswani *et al.*, 2017), com o objetivo de permitir que mais dados pudessem ser armazenados e processados em conjunto, desta forma possibilitando fornecer mais informações de contexto para um problema computacional de processamento de texto ou imagem. A ideia dos transformers é que possibilitou o surgimento de modelos mais recentes como ChatGPT e Gemini, que são exemplares do que é considerado inteligência artificial hoje em dia.

Modelos baseados em transformers possuem grande capacidade de mapeamento semântico de dados e de geração de texto verossímil, que podem ser consideradas como suas características mais salientes. Estas capacidades são obtidas com base na observação e processamento de grande quantidade de dados de exemplo, processo que é chamado

de “treinamento” no jargão da área, realizados com o objetivo de se identificar padrões existentes nos dados para poder determinar, por exemplo, quais palavras que aparecem comumente próximas a outras em certos contextos. Concluída esta etapa de treinamento, o modelo é capaz de produzir texto em resposta a uma questão: primeiro é feito o mapeamento semântico da questão para encontrar termos e sentenças com significado parecido na memória do modelo, e segundo, com base nos termos e sentenças encontrados, o modelo faz a síntese de um texto que tenha verossimilhança. Note o uso do termo verossimilhança no lugar de corretude: como os modelos são criados de acordo com a definição de Turing, basta que forneçam respostas indistinguíveis das de um humano, e cabe a quem coloca a questão avaliar a corretude das respostas fornecidas.

2.2 Evolução das redes neurais em PLN

Hoje, modelos de redes neurais artificiais são prevalentes em diversas áreas de aplicação de machine learning, como classificação e geração de imagens, Processamento de Linguagem Natural (PLN), reconhecimento de fala, etc. A seguir, apresentaremos uma breve (e incompleta) história da evolução desses modelos em algumas sub-áreas de PLN, desde desenvolvimentos anteriores à adoção de redes neurais, até o advento dos Transformers.

Harris (1954) e Firth (1957) argumentam que o sentido de uma palavra pode ser deduzido, em parte, a partir dos contextos onde ela é comumente utilizada - "conhecerás uma palavra pela compania que ela mantém". Essa ideia, que podemos chamar de Hipótese Distribucional, direta ou indiretamente, guiou inúmeros avanços em diferentes campos na área de PLN. Deerwester *et al.* (1990), com seu modelo de Análise Semântica Latente (LSA), foi um dos pioneiros em operacionalizar essa ideia, obtendo resultados promissores na área de indexação e recuperação de informação. Neste modelo, aplica-se uma técnica de álgebra linear, denominada Decomposição em Valores Singulares (SVD), a uma matriz de frequência (ou alguma outra métrica, como TF-IDF) termo-documento, para se obter uma representação mais densa e de menor dimensionalidade desta matriz. O resultado é uma representação vetorial que consegue capturar relações semânticas entre termos e documentos, mapeando itens com significados similares a lugares comuns em um espaço latente, configurando uma forma primitiva do conceito que hoje conhecemos como *word embeddings*.

Bengio *et al.* (2003) propôs o uso de redes neurais arificiais para modelar distribuições de probabilidade conjunta de sequências de palavras em um determinado idioma. Na arquitetura proposta, uma rede neural é treinada para executar duas tarefas simultaneamente: Associar cada palavra de um vocabulário a um vetor de $\mathbb{R}^m$ (*embedding*), e determinar a distribuição de probabilidade condicional de sequências de palavras do vocabulário (isto é, dada uma sequência de palavras, determinar a distribuição de probabilidade da próxima palavra), expressas em termos de suas representações vetoriais. Embora poderosa, essa abordagem ainda tinha uma grande complexidade computacional, inviabilizando o seu uso em grandes conjuntos de dados. Um grande avanço na área, que possibilitou a criação de modelos de embedding com um número bem maior de dimensões, e o uso de volumes muito maiores de dados no treinamento desses modelos, foi a chegada dos modelos *Continuous Skip-gram* e *Continuous Bag-of-Words* (CBOW) (Mikolov, Chen *et al.*, 2013). Estas arquiteturas dispensaram o uso de camadas ocultas, e tinham como objetivo

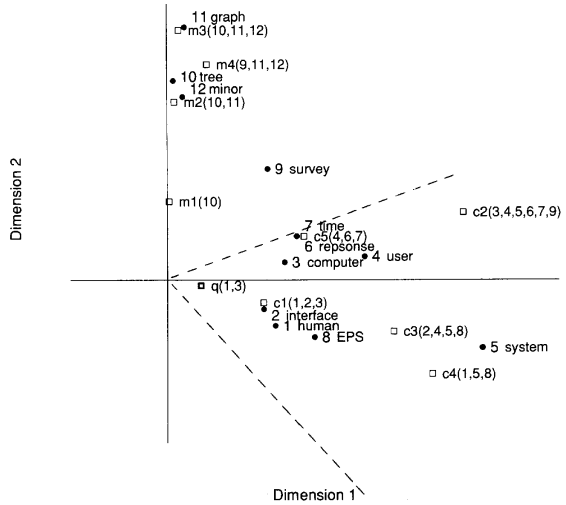


Figura 2.1: *Representação do espaço latente gerado pelo LSA.*

simplesmente prever uma palavra dadas as palavras ao seu redor (arquitetura CBOW), ou, dada uma palavra, prever as palavras ao seu redor (arquitetura Skip-gram), usando uma janela deslizante de contexto, cujo tamanho é dado como parâmetro da arquitetura. Embora, nessa arquitetura, a ordem das palavras na janela de contexto deixasse de ter importância (no caso do CBOW, usa-se a média dos vetores dessas palavras, por exemplo), os embeddings gerados conseguiam capturar bem melhor o sentido global das palavras, principalmente devido à baixa complexidade computacional do modelo, em relação ao proposto por Bengio, o que possibilitou o uso de um volume muito grande de dados em seu treinamento, e a criação de embeddings de dimensionalidades muito maiores (até 1000 dimensões, comparados com as 50-100 do modelo proposto por Bengio). Porém, vale dizer que este modelo, diferentemente do anterior, não é um modelo de linguagem probabilístico, no sentido de que o seu objetivo não é de modelar distribuições de probabilidades de sequências de palavras, mas de aprender representações semânticas eficientes. Posteriormente, usando técnicas como subamostragem e amostragem negativa, Mikolov mostrou como acelerar o processo de aprendizado e inferência dos modelos, e criar melhores representações (Mikolov, Sutskever *et al.*, 2013).

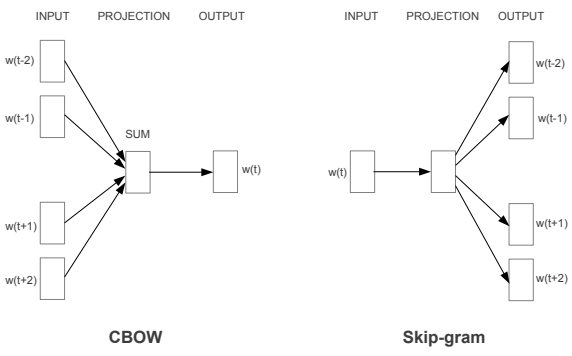


Figura 2.2: *Arquiteturas CBOW e Skip-gram.*

Na área de tradução automática de texto, um trabalho pioneiro propôs o uso de uma arquitetura do tipo *encoder-decoder* para traduzir sequências de palavras do Inglês para o Francês (Sutskever *et al.*, 2014). O modelo era composto por duas redes do tipo *Long Short-Term Memory*, ou LSTM, que é um tipo de rede neural recorrente (RNN) dotada de mecanismos de "controle de memória", para lidar com problemas de dependência de longo alcance em sequências. O modelo funcionava em duas etapas: primeiro, o *encoder*, consistindo de uma RNN mapeava sequências de palavras de tamanho variado para um vetor de dimensão fixa, denominado *estado oculto* (também chamado de *vetor de contexto*) iterativamente. Depois, o *decoder* usa o último estado oculto do *encoder* para gerar a sequência de saída. Assim, a predição de cada palavra na saída da rede é influenciada tanto por todas as palavras na entrada, quanto pelas previsões anteriores.

Uma evolução significativa dessa última abordagem foi introduzida no trabalho de Bahdanau *et al.* (2016), onde foi introduzida uma versão precursora dos mecanismos de atenção modernos. No modelo anterior, sequências de palavras eram mapeadas para um vetor de dimensão fixa, o que fazia com que a performance do modelo deteriorasse rapidamente para sequências longas (Cho *et al.*, 2014). O novo modelo proposto buscava livrar o *encoder* de ter que comprimir todo o contexto em um único vetor de tamanho fixo, gerando uma sequência de estados ocultos que o *decoder* poderia acessar. Além disso, o modelo dispunha de um mecanismo para "procurar", entre os estados ocultos, informações relevantes para a predição de uma determinada palavra dentro da sequência, e *selecionar* quais palavras deveriam influenciar a palavra atual. O resultado foi uma melhora significativa na transdução de sequências longas.

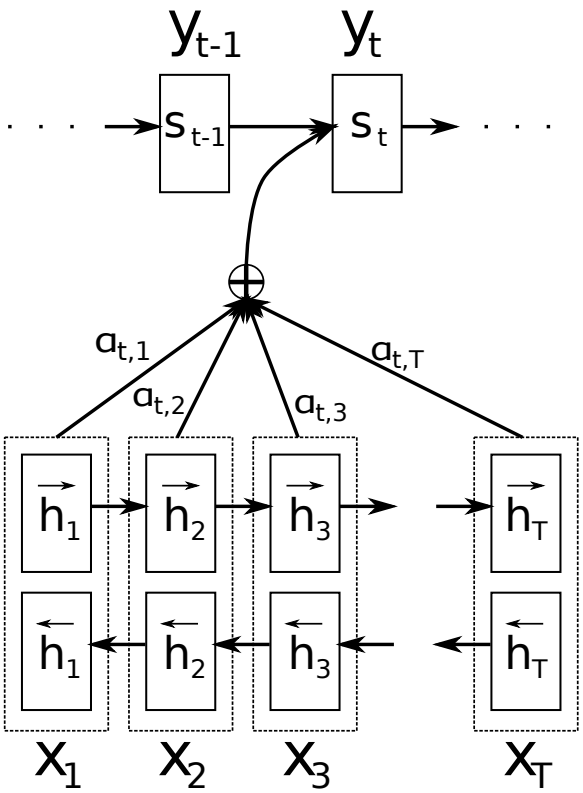


Figura 2.3: *RNNSearch.*

2.3 Transformers e desenvolvimentos subsequentes

Todos estes desenvolvimentos culminaram na criação da arquitetura batizada de Transformer (Vaswani *et al.*, 2017). Todos os modelos mencionados anteriormente dependem do processamento sequencial de tokens da entrada, devido ao emprego de redes neurais recorrentes. Na arquitetura Transformer, os tokens de entrada são processados em paralelo, o que permitiu uma enorme escalabilidade e aceleração de seus processos de treinamento e inferência. A seguir mostramos uma descrição superficial da sua arquitetura.

2.3.1 Arquitetura do modelo Transformer

Assim como os modelos de transdução já citados (Bengio *et al.*, 2003; Sutskever *et al.*, 2014; Bahdanau *et al.*, 2016), um Transformer conta com um *encoder* e um *decoder*, porém, com a diferença de não haver RNNs envolvidas.

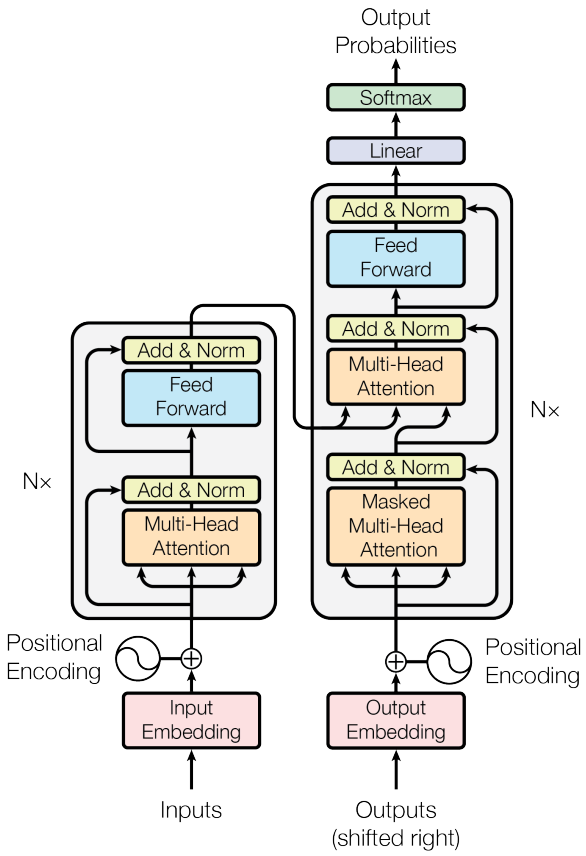


Figura 2.4: *Arquitetura de um Transformer.*

Encoder

O encoder começa com uma camada de *embedding*, juntamente com uma camada de codificação posicional, que serve para imbuir os embeddings de informação sobre suas posições, já que não estamos mais utilizando RNNs. Em cima disto, múltiplas camadas idênticas são empilhadas. Cada uma destas camadas é composta por duas sub-camadas. A

primeira implementa um mecanismo de atenção paralela (*multi-head attention*), e a segunda é uma simples rede do tipo *multi-layer perceptron* (também chamada de rede *feed-forward*).

Decoder

O decoder também é composto por várias camadas idênticas empilhadas, onde cada camada é composta por três subcamadas, duas iguais às subcamadas do encoder, e uma extra, *masked self-attention*, que consiste em uma camada de atenção modificada para evitar que uma palavra gerada pelo modelo "preste atenção" em uma palavra subsequente (isto é, seja modificada por ela).

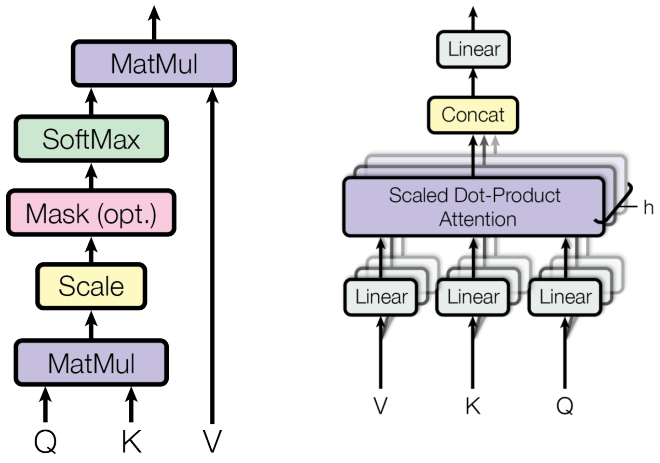
2.3.2 Atenção

Podemos entender os mecanismos de atenção presentes na arquitetura Transformer como uma maneira de palavras "informarem" umas às outras quais palavras elas podem influenciar, e de que forma. Mais especificamente, sobre cada embedding (que representa um token, na entrada ou na saída), um bloco de atenção calcula três valores, Q, K e V (*query*, *key* e *value*) usando transformações lineares simples. Para cada embedding, o seu valor de Q é comparado com o valor de K de si mesmo e todos os outros (ou dos embeddings anteriores, no caso de *masked self-attention*), usando uma função de similaridade (no caso, o produto escalar), seguida de uma normalização (*softmax*). O resultado então é usado para calcular a influência dos embeddings uns sobre os outros (efetivamente, soma-se o valor da atenção para um embedding ao valor do embedding, em conexões residuais):

$$\text{Atencao}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

onde d_k é a dimensionalidade de Q e K.

Após uma camada de atenção há uma rede *feed-forward*, com funções de ativação do tipo ReLU (*Rectified Linear Unit*).



(a) *Atenção de produto escalar.*

(b) *Multi-head attention.*

Figura 2.5: *Mecanismos de atenção nos Transformers.*

Multi-head attention

Ao invés de calcular uma única função de atenção, os valores Q, K e V são projetados linearmente h vezes em espaços de menor dimensionalidade, e vários valores de atenção são calculados em paralelo, e depois esses valores são concatenados e projetados novamente no espaço original, resultando nos valores finais de atenção. Isso permite ao modelo considerar influências entre palavras em diferentes espaços latentes, em diferentes posições.

2.3.3 BERT, SBERT e derivados

A chegada dos Transformers representou um novo paradigma para o uso de redes neurais em PLN, e em outras áreas de aplicação. Pela sua capacidade de modelagem de dependências de longo alcance, e pela sua escalabilidade, modelos baseados em Transformers logo dominaram o cenário de machine learning, impulsionados principalmente por modelos generativos como ChatGPT (Radford e Narasimhan, 2018), Gemini (Gemini Team, 2025), entre outros. Estes modelos apresentam algumas variações arquitetônicas (como uso apenas de decoders em modelos generativos) mas ainda mantêm a estrutura básica de um Transformer, com seus mecanismos de atenção. Um exemplo de modelo que conta apenas com encoders é o BERT (Devlin *et al.*, 2019). Este modelo é pré-treinado em diferentes tarefas, como predição de tokens ocultos, e predição de próxima sentença, e depois passa por um ajuste fino para tarefas específicas, como detecção de paráfrases, inferência textual, classificação de texto, etc. O seu pré-treinamento faz com que o modelo obtenha um "entendimento" básico da linguagem, que melhora consideravelmente o seu desempenho em tarefas específicas posteriores.

Apesar de alcançar bons resultados, o modelo BERT se torna computacionalmente inviável para algumas tarefas, como busca por similaridade semântica em um grande corpus, ou clustering de documentos. Para essas tarefas, é desejável um modelo capaz de gerar embeddings de frases, o que o BERT não foi projetado para fazer. Para mitigar essa limitação surgiu o SBERT (Reimers e Gurevych, 2019), que é uma modificação dos modelos BERT pré-treinados, usando redes neurais siamesas e estruturas de rede do tipo triplet para gerar boas representações semânticas de frases. Essas representações então podem ser comparadas usando distância de cosseno, por exemplo. Os embeddings usados neste trabalho são gerados por descendentes desse modelo.

2.4 Modelagem de Tópicos

Modelos de tópicos são ferramentas computacionais para descobrir automaticamente estruturas temáticas comuns em grandes quantidades de documentos, permitindo que sejam organizados e sumarizados de acordo com os assuntos presentes (David M. Blei, 2012). Modelos de tópicos são muito úteis na prospecção de documentos de interesse, em sistemas de recomendação, e em análises de séries de documentos. Existem diferentes abordagens para essa tarefa, como *Latent Dirichlet Allocation* (LDA), *Non-negative Matrix Factorization* (NMF) e modelos baseados em redes neurais. No LDA (David M Blei *et al.*, 2003), tratamos um tópico como uma distribuição de probabilidade sobre um vocabulário, e assumimos que cada documento contém uma mistura de tópicos, diferentes documentos contendo diferentes proporções dos tópicos. O modelo então supõe que os documentos

são gerados de acordo com essas distribuições, que são tratadas como variáveis ocultas, e aproxima a distribuição posterior de probabilidade, dada a distribuição observada de palavras por documento.

Modelos neurais de tópicos geralmente envolvem *embeddings* de palavras ou frases gerados por redes neurais, como o BERTopic, que utilizamos neste trabalho.

2.4.1 BERTopic

No BERTopic (Grootendorst, 2022), usamos agrupamentos de *embeddings* de trechos de documentos para representar tópicos subjacentes no texto.

Ele é constituído, basicamente, de 4 etapas:

1. Extração de embeddings
2. Redução de dimensionalidade
3. Clustering
4. Extração de representações dos tópicos

Primeiramente, extrai-se representações vetoriais das sentenças do corpus, utilizando algum modelo de *sentence embedding*. Utilizamos então um clustering desses embeddings como nosso modelo de tópicos. Entretanto, a alta dimensionalidade dos embeddings gerados acaba por prejudicar a performance de algoritmos e técnicas baseados em conceitos como proximidade, distância, ou vizinhos mais próximos, como diversos algoritmos de clustering (Radovanović *et al.*, 2010; Aggarwal *et al.*, 2001; Steinbach *et al.*, 2003). Técnicas de redução de dimensionalidade se demonstraram eficazes para mitigar esse efeito, e melhorar a qualidade de clusterings (Herrmann *et al.*, 2022; Allaoui *et al.*, 2020).

Por fim, depois de obter aglomerados de sentenças (ou trechos arbitrários de documentos), correspondendo aos tópicos detectados, precisamos extrair representações interpretáveis destes tópicos. Alguns modelos de tópicos baseados em embeddings utilizam termos próximos do centróide de um cluster como representação dos tópicos (Sia *et al.*, 2020; Angelov, 2020). No BERTopic, utilizamos os termos com maior poder discriminante entre os documentos de um cluster, calculando uma variante da métrica TF-IDF (*Term Frequency - Inverse Document Frequency*), denominada cTF-IDF (o 'c' vem de "classe"):

$$\text{cTF–IDF}_{t,c} = f_{t,c} \cdot \log\left(1 + \frac{A}{f_t}\right)$$

Onde $f_{t,c}$ é a frequência do termo t na classe (ou tópico, cluster) c, f_t é a frequência geral do termo t no corpus, e A é o número médio de termos por classe. As palavras com pontuações altas em um determinado tópico são justamente aquelas que, simultaneamente, aparecem muito nesse tópico, e pouco em outros tópicos.

2.4.2 UMAP

Como já vimos, aplicar técnicas de redução de dimensionalidade antes de executar tarefas de clustering tende a render melhores resultados. Existem diversas técnicas e algoritmos

para esse fim, com diferentes vantagens e aplicações, como PCA - *Principal Component Analysis* (Jolliffe e Cadima, 2016), t-SNE - *t-Stochastic Nearest Neighbors* (Maaten *et al.*, 2008), e UMAP - *Uniform Manifold Learning and Approximation* (McInnes *et al.*, 2018). Neste trabalho (assim como o autor da técnica de modelagem de tópicos usada), optamos por usar o UMAP, pela sua habilidade em preservar estruturas não-lineares nos dados, e manter um equilíbrio entre estruturas locais e globais, além de ter menores tempos de execução em relação a outras técnicas não-lineares.

A ideia principal do UMAP é: dada uma representação em grafo do dataset original, aprender uma representação, em um espaço de dimensionalidade reduzida, que seja o mais similar possível, em um certo sentido, ao grafo construído no espaço original. Simplificadamente, o algoritmo faz isso da seguinte maneira: primeiro, para cada ponto, ele calcula quais são seus k vizinhos mais próximos, onde o valor de k é um parâmetro do algoritmo. Então, para cada par de vizinhos, ele calcula uma certa "probabilidade assimétrica" de que esses pontos estejam conectados localmente, de acordo com a densidade local na região de cada ponto, isto é, a probabilidade de um ponto x_i estar conectado a outro ponto x_j, do ponto de vista de x_i, pode ser diferente da probabilidade de conexão de x_j e x_i, do ponto de vista de x_j. O grafo ponderado original é então construído, utilizando a probabilidade de ao menos uma conexão existir entre dois pontos, como os pesos para as arestas. O algoritmo então cria uma representação inicial dos pontos do dataset, porém em um espaço de menor dimensionalidade, e ajusta iterativamente as posições desses pontos para minimizar a diferença entre o grafo correspondente de menor dimensionalidade e o grafo original. Como toda técnica de redução de dimensionalidade, é claro que a projeção de menor dimensionalidade não é uma representação fiel do conjunto de dados. Em especial, informações sobre distâncias entre pontos considerados não-conectados são perdidas. Porém, os k pontos mais próximos de qualquer ponto no espaço original provavelmente também estarão perto na projeção de dimensionalidade reduzida, assim como pontos considerados não-conectados provavelmente estarão distantes. Assim, essa técnica serve como uma boa etapa de pré-processamento para tarefas subsequentes de clustering.

2.4.3 HDBSCAN

HDBSCAN - *Hierarchical Density-Based Spatial Clustering of Applications with Noise* é um algoritmo de clustering hierárquico baseado em densidade, capaz de extrair clusters de densidades e formas variadas, não necessariamente convexas, e ainda classificar pontos como ruído, não pertencendo a nenhuma classe (Campello *et al.*, 2013). A seguir damos uma simples descrição de seu funcionamento.

Seja $\mathbf{X} = \{x_1, \ldots, x_n\}$ um conjunto de dados, e $d(\cdot, \cdot)$ uma medida de distância. Dado um valor k, definimos uma medida de distância central $\text{core}_k(x)$, igual à maior distância entre x e um de seus k vizinhos mais próximos, para estimar a densidade ao redor de um ponto x. Definimos então outra métrica, distância de alcance mútuo, como:

$$d_{\text{mreach}-k}(x_i, x_j) = \max\{\text{core}_k(x_i), \text{core}_k(x_j), d(x_i, x_j)\}$$

Então, construímos um grafo completo, onde cada vértice corresponde a um ponto no dataset, e o peso p de uma aresta é dado por $p(x_i, x_j) = d_{\text{mreach}-k}(x_i, x_j)$. Durante a execução do algoritmo, usamos esse grafo para representar clusters: ao retirar um conjunto de arestas,

os nossos clusters são representados pelos vértices de componentes conexas do grafo.

Depois, obtemos a árvore geradora mínima deste grafo. Podemos fazer isso eficientemente utilizando o algoritmo de Prim (ou outros algoritmos, dependendo do espaço em que estamos trabalhando). Ao ordenar as arestas dessa matriz em ordem decrescente e ir retirando as arestas uma a uma, o efeito que obtemos é como um clustering divisivo: a cada aresta removida, dividimos uma componente conexa em duas. Assim podemos obter uma sequência de partições do nosso conjunto de dados, ou uma hierarquia de clusters, onde os clusters vão sendo divididos em clusters menores, usando ligação única (*single linkage*) como distância entre clusters. Podemos representar essa hierarquia usando uma árvore binária, que efetivamente é o dendrograma da clusterização. Um detalhe importante dessa construção, é que ao construir essa árvore, definimos "divisões verdadeiras" de clusters quando os clusters resultantes são maiores do que um tamanho mínimo, dado como parâmetro. Nesse caso, ambos os clusters são classificados como clusters reais. Quando um cluster resultante de uma divisão é menor que o tamanho mínimo, paramos o processo de divisão desse cluster, e classificamos os seus pontos como ruído.

Para extrair um conjunto de clusters dessa hierarquia, é definida uma métrica de estabilidade para cada cluster, que mede, informalmente, o quanto um cluster resiste ao processo de divisão. Selecionamos então os clusters cuja soma das estabilidades seja máxima, sujeito à restrição de que não podemos selecionar um cluster e algum de seus descendentes na hierarquia.

2.5 Obras e autores estudados

Neste trabalho utilizaremos edições em inglês das seguintes obras: as Etimologias (*Etymologiae*), de Isidoro de Sevilha, Sobre Agricultura (*De Agri Cultura*), de Catão, e Sobre Ciência Agrícola (*De Re Rustica*), de Varrão. A escolha destas obras não é por acaso: estudiosos da obra de Isidoro consideram que ele se baseou em diversas obras da antiguidade para escrever as Etimologias, particularmente as obras de Catão e Varrão sobre agricultura. Sendo assim, tentaremos encontrar similaridades de ideas entre os textos a partir do uso de modelos computacionais. A seguir apresentaremos detalhes sobre os autores e seu contexto.

Isidoro de Sevilha (c. 560 - 636) foi um clérigo, teólogo e pensador da alta idade média, que é considerado um dos intelectuais mais importantes do seu tempo, e cuja influência foi sentida por muitos séculos após a sua morte. Nasceu em Cartagena, que à época era parte do Reino Visigótico, estado que ocupou as regiões da Península Ibérica e atual sul da França no período seguinte ao fim do Império Romano do ocidente, e fazia parte de uma família que percentia à elite hispano-romana. Seus três irmãos ocuparam funções importantes na igreja, com destaque para seu irmão mais velho, Leandro de Sevilha (c. 534 - c. 600) foi Bispo de Sevilha, cargo que Isidoro ocuparia após a morte de Leandro. Todos os quatro irmãos são venerados como santos pela Igreja Católica.

Como Bispo de Sevilha, Isidoro exerceu grande influência no seu tempo, presidindo sínodos e concílios importantes, como os de Sevilha e Toledo, protegendo os monastérios, e ainda se envolveu na conversão dos reis Visigodos do Arianismo, uma doutrina cristã

não-trinitária, para o cristianismo Calcedoniano, que veio a se tornar a doutrina dominante na Igreja Católica.

Como intelectual, produziu diversas obras, dentre as quais se destacam as Etimologias, que são um conjunto de livros que formam uma enciclopédia etimológica, que resume e organiza o conhecimento de diversos autores da antiguidade clássica. A obra segue a tradição de enciclopedistas clássicos, como Plínio, o Velho (c. 23 - 79), com o uso de ordenação alfabética de tópicos e de uma abordagem literária para o conhecimento, baseada no pensamento analógico.

As Etimologias tratam de temas diversos como gramática, retórica, matemática, direito, a Igreja, heresias, guerra, agricultura, entre outros. A sua influência foi tão grande nos séculos seguintes que algumas das obras clássicas utilizadas como base deixaram de ser lidas e copiadas e acabaram se perdendo no tempo. Era considerado o texto base para a educação sobre o período clássico durante a maior parte da idade média.

O estudo crítico das Etimologias revela suas possíveis fontes clássicas, que na maioria das vezes não são citadas por Isidoro. De acordo com Stephen A. Barney, tradutor para o inglês (Barney *et al.*, 2006) das Etimologias, é possível identificar que o material dos livros I e II, que tratam de gramática, retórica e dialética (as disciplinas do *trivium*) provavelmente foram extraídos dos Institutos, de Cassiodoro (c. 485 - c. 585), o livro III, sobre matemática (contendo as disciplinas do *quadrivium*), provavelmente foi inspirado em Boécio (c. 480 - 524), e o livro XVII, sobre agricultura, deriva de Catão, o Velho (234 - 149 a.C.) e Varrão (116 - 27 a.C.), para citar alguns exemplos.

Marco Pórcio Catão, o Velho (234 - 149 a.C.), foi um soldado, senador e historiador romano. Nascido em uma família de plebeus, que era a classe baixa de cidadãos livres em Roma, desecndia de gerações de soldados com reputação de bravura, como seu pai e seu bisavô. Ainda na infância, com a morte de seu pai, passou a cuidar das atividades da fazenda família. Como jovem soldado, especula-se que aos 20 anos tenha participado de campanhas das Guerras Púnicas no papel de tribuno militar, uma patente de oficial do exército romano. Ao retornar do campo de batalha para a sua fazenda, e com o apoio do seu vizinho e amigo Lúcio Valério Flaco, iniciou carreira política como *questor*, cargo que possuía diversas atribuições, entre elas a de cobrança de impostos e de supervisão financeira. Daí se seguiram diversos cargos políticos importantes, como *pretor* (magistrado), *consul* (o cargo mais alto da República Romana) e *censor* (magistrado de nível superior). Em paralelo a suas atividades políticas, escreveu diversas obras, cuja maioria infelizmente foi perdida. Escreveu uma história de Roma em sete livros chamada de Origens, uma obra sobre assuntos militares, e a obra sobre agricultura que utilizaremos neste trabalho, a única preservada na íntegra. Além disso, foi famoso orador e teve cerca de 150 discursos registrados.

Marco Terêncio Varrão (116 - 27 a.C.) foi um intelectual e polímata romano, descrito por Petrarca como a "terceira grande luz" de Roma, depois de Virgílio e Cícero. Nascido em família pertencente à classe equestre de Roma, abaixo somente da classe senatorial, ocupou cargos políticos ao longo da vida, como *questor*, *pretor* e tribuno do povo. Estudou com o filólogo romano Lúcio Élio Estilo e com o filósofo platonista Antíoco de Ascalão. Foi também um líder militar sem grande prestígio durante a Guerra Civil Cesariana. Foi um escritor prolífico, que produziu cerca de 74 obras sobre temas diversos, entre as quais se destacam a Cronologia Varroniana, que lista as datas de eventos importantes de Roma,

e os nove livros das Disciplinas, organizados de acordo com os temas das artes liberais da época, e que serviram de exemplo para enciclopedistas que vieram posteriormente, como Plínio, o Velho, e o próprio Isidoro de Sevilha.

Capítulo 3

Experimentos

Neste capítulo detalhamos os dados utilizados neste trabalho juntamente com o *pipeline* de processamento empregado para a sua coleta e preparação, e a configuração experimental utilizada no estudo.

3.1 Conjunto de dados

Como conjunto de dados a ser explorado neste trabalho, temos as seguintes obras:

- Etimologias (*Etymologiae*) (Barney *et al.*, 2006), por Isidoro de Sevilha (c. 560 - 636);
- Sobre Agricultura (*De Agri Cultura*) (Cato e Varro, 1934), por Marco Pórcio Catão, o Velho (234 - 149 a.C.);
- Sobre Ciência Agrícola (*De Re Rustica*) (Cato e Varro, 1934), por Marco Terêncio Varrão (116 - 27 a.C.).

No caso das Etimologias, foi utilizada a tradução (Barney *et al.*, 2006) do latim para o inglês em formato *pdf*. Para os textos de Catão e Varrão foram utilizadas edições da *Loeb Classical Library* (Cato e Varro, 1934), atualmente em domínio público e disponíveis *online* (Thayer, 2025) em formato *html*.

3.2 Preparação dos dados

(TODO: Adicionar figura do esquema geral do processo)

3.2.1 Limpeza

No caso das Etimologias, o texto foi extraído do arquivo *pdf* do livro e convertido para formato de texto puro, removendo-se cabeçalhos, rodapés, números de página, e outros artefatos dessa conversão que não são necessários para a análise. Devido a peculiaridades do formato *pdf* e dificuldades encontradas nas bibliotecas utilizadas para a extração do texto, parte do processo de limpeza teve de ser realizado manualmente. No caso das outras

obras, disponíveis em formato *html*, o processamento foi realizado de forma totalmente automática.

3.2.2 Pré-processamento

Primeiramente, pelo modo como cada livro está formatado, fizemos um processamento inicial para separar cada livro em capítulos, e juntar palavras com hífen. Em seguida, cada capítulo foi subdividido em sentenças usando o módulo *SentenceRecognizer* da biblioteca Spacy (Honnibal *et al.*, 2020), que possui diversos recursos para processamento de linguagem natural (NLP).

Uma particularidade da edição escolhida das Etimologias é que cada capítulo é subdividido em seções relativamente curtas, a maioria tendo somente uma oração. Com isso, podemos fazer a modelagem dos tópicos usando dois níveis diferentes de granularidade, a depender de como definimos um "documento", unidade básica de nossa análise:

- **Seções**, já presentes no texto;
- **Sentenças**, delimitadas automaticamente;

Em nossa análise, utilizamos as duas abordagens e comparamos os resultados.

TODO: Exemplos das seções.

3.2.3 *Stop words*, *stemming* e lematização

Tarefas rotineiras de pré-processamento de linguagem natural incluem: remoção de *stop words*, que são palavras muito comuns e irrelevantes, *stemming*, e lematização, que são formas de padronizar palavras, reduzindo-as a sua forma mais básica. Apesar de outras técnicas de modelagem de tópicos adotarem essas tarefas como parte do seu processo de preparação dos dados, não utilizamos essas técnicas para este trabalho, pois os modelos de *embedding* que usamos utilizam informações contextuais de cada palavra em uma sentença, e remover palavras ou modificá-las poderia prejudicar a performance de tais modelos (Chaerul Haviana *et al.*, 2023).

3.2.4 Geração de *embeddings*

Depois da etapa de pré-processamento dos textos, geramos *embeddings* para cada documento, utilizando modelos do tipo *sentence embedders* pré-treinados. Os modelos usados para a geração desses *embeddings* foram: *sentence-transformers/LaBSE*, *jinaai/jina-embeddings-v3*, *intfloat/multilingual-e5-large-instruct*, *nomic-ai/nomic-embed-text-v2-moe*. Estes modelos foram desenhados e treinados para que sentenças semanticamente similares em línguas diferentes, ou traduções, estejam próximas umas das outras em um espaço latente. Podemos usar estes modelos para comparar textos em diferentes idiomas e analisar suas conexões.

3.2.5 Conjuntos de dados

Após essas etapas iniciais, definimos nossos conjuntos de dados. Cada conjunto é definido por um modelo de embedding e um nível de granularidade, já explicado. TODO: exemplos, visualizações dos espaços de embeddings.

3.3 Persistência

Os embeddings gerados, suas reduções, e diversos metadados foram armazenados em um banco de dados do *ChromaDB*, juntamente com cada documento.

TODO: versão final do modelo de dados no ChromaDB.

Dessa forma, podemos fazer diversos tipos de busca, como busca por similaridade, busca textual, filtrar resultados baseados em metadados de cada documento, como autor, idioma, etc.

3.4 Experimentos

Como já foi explicado, a técnica de modelagem de tópicos usada (BERTopic) é composta de diferentes etapas, com várias possibilidades de escolha de algoritmos para cada etapa, e hiperparâmetros para estes algoritmos. Neste trabalho, foram usados o UMAP para redução de dimensionalidade, e o HDBSCAN para o clustering dos documentos e deteção dos tópicos.

3.4.1 Redução de dimensionalidade

A seguir, fizemos uma redução de dimensionalidade de cada um dos conjuntos de embeddings gerados. Para isso, utilizamos o UMAP, com os seguintes parâmetros:

Tabela 3.1: *Parâmetros usados para o UMAP*

Nome do parâmetro	Descrição	Valor usado	Nota
`n_neigbors`	Controla como o UMAP equilibra estrutura local versus global nos dados. Valores maiores levam a uma visão mais global da estrutura dos dados, e valores menores a uma visão mais local.	`10`	O valor padrão para o BERTopic é 15.
`n_components`	Dimensionalidade do *dataset* resultante.	`5`	Valor padrão para o BERTopic.

Continua na próxima página

Tabela 3.1 – continuação

Nome do parâmetro	Descrição	Valor usado	Nota
`min_dist`	Controla o quão dispersos os pontos estarão na projeção de baixa dimensionalidade. Valores baixos podem ser interessantes para tarefas de clustering.	`0`	Valor padrão para o BERTopic.
`low_memory`	Restringe o uso de memória em detrimento da velocidade da computação. Útil quando há pouca memória disponível.	`false`	
`metric`	Métrica usada para calcular distâncias entre pontos.	`"cosine"`	Valor padrão para o BERTopic.
`random_state`	Usado para garantir determinismo.	`42`	

TODO: normalmente o BERTopic usa `n_neighbors = 15`, então por que eu mudei esse valor? A implementação usada foi a da biblioteca umap-learn.[1]

3.4.2 Clustering

Com os embeddings gerados e suas reduções já computadas, inicializamos o BERTopic de modo a pular essas etapas iniciais e começar pela etapa de clustering. O algoritmo utilizado foi o HDBSCAN, disponível na biblioteca Scikit-learn,[2] e utilizamos os seguintes parâmetros:

Tabela 3.2: *Parâmetros usados para o HDBSCAN*

Nome do parâmetro	Descrição	Valor usado	Nota
`min_cluster_size`	Controla o tamanho mínimo e a quantidade de *clusters* gerados.	`10`	Valor padrão para o BERTopic.
`min_samples`	Controla a quantidade de pontos classificados como ruído.	`1`	Escolhemos um valor baixo para tentar diminuir a quantidade pontos classificados como outliers.

Para melhorar a representação dos tópicos gerados, fizemos alguns ajustes nas etapas finais do método:

[1] https://umap-learn.readthedocs.io/en/latest/

[2] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.HDBSCAN.html

- Remoção de stop-words, no CountVectorizer
- Remoção de palavras comuns e uso de BM-25 weighting measure.

Capítulo 4

Resultados

Capítulo 5

Conclusão

Referências

[Aggarwal *et al.* 2001] Charu C. Aggarwal, Alexander Hinneburg e Daniel A. Keim. "On the surprising behavior of distance metrics in high dimensional spaces". In: *Proceedings of the 8th International Conference on Database Theory*. ICDT '01. Berlin, Heidelberg: Springer-Verlag, 2001, pp. 420–434. isbn: 3540414568 (citado na pg. 11).

[Allaoui *et al.* 2020] Mebarka Allaoui, Mohammed Lamine Kherfi e Abdelhakim Cheriet. "Considerably improving clustering algorithms using umap dimensionality reduction technique: a comparative study". In: jul. de 2020, pp. 317–325. isbn: 978-3-030-51934-6. doi: 10.1007/978-3-030-51935-3_34 (citado na pg. 11).

[Angelov 2020] Dimo Angelov. *Top2Vec: Distributed Representations of Topics*. 2020. arXiv: 2008.09470 [cs.CL]. url: https://arxiv.org/abs/2008.09470 (citado na pg. 11).

[Bahdanau *et al.* 2016] Dzmitry Bahdanau, Kyunghyun Cho e Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL]. url: https://arxiv.org/abs/1409.0473 (citado nas pgs. 7, 8).

[Barney *et al.* 2006] Stephen A Barney, Wendy J Lewis, Jennifer A Beach e Oliver Berghof. *The etymologies of Isidore of Seville*. Cambridge University Press, 2006 (citado nas pgs. 1, 14, 17).

[Bengio *et al.* 2003] Yoshua Bengio, Réjean Ducharme, Pascal Vincent e Christian Janvin. "A neural probabilistic language model". In: *Journal of machine learning research*. 2003. url: https://api.semanticscholar.org/CorpusID:221275765 (citado nas pgs. 5, 8).

[David M Blei *et al.* 2003] David M Blei, Andrew Y Ng e Michael I Jordan. "Latent dirichlet allocation". *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022 (citado nas pgs. 1, 10).

[David M. Blei 2012] David M. Blei. "Probabilistic topic models". *Commun. ACM* 55.4 (abr. de 2012), pp. 77–84. issn: 0001-0782. doi: 10.1145/2133806.2133826. url: https://doi.org/10.1145/2133806.2133826 (citado na pg. 10).

[Campello *et al.* 2013] Ricardo J. G. B. Campello, Davoud Moulavi e Joerg Sander. "Density-based clustering based on hierarchical density estimates". In: *Advances in Knowledge Discovery and Data Mining*. Ed. por Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda e Guandong Xu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 160–172. isbn: 978-3-642-37456-2 (citado na pg. 12).

[Cato e Varro 1934] Marcus Porcius Cato e Marcus Terentius Varro. *On Agriculture; On Agriculture*. Ed. por William Davis Hooper e Harrison Boyd Ash. Vol. 283. Loeb Classical Library. Latin text with facing English translation; revised edition. Cambridge, MA: Harvard University Press, 1934. isbn: 978-0674993136. doi: 10.4159/DLCL.cato-agriculture.1934 (citado na pg. 17).

[Chaerul Haviana *et al.* 2023] Sam Farisa Chaerul Haviana, Sri Mulyono e Badie'Ah. "The effects of stopwords, stemming, and lemmatization on pre-trained language models for text classification: a technical study". In: *2023 10th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*. 2023, pp. 521–527. doi: 10.1109/EECSI59885.2023.10295797 (citado na pg. 18).

[Cho *et al.* 2014] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau e Yoshua Bengio. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. 2014. arXiv: 1409.1259 [cs.CL]. url: https://arxiv.org/abs/1409.1259 (citado na pg. 7).

[Deerwester *et al.* 1990] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer e Richard Harshman. "Indexing by latent semantic analysis". *Journal of the American society for information science* 41.6 (1990), pp. 391–407 (citado nas pgs. 1, 5).

[Devlin *et al.* 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee e Kristina Toutanova. "Bert: pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics*. 2019, pp. 4171–4186 (citado nas pgs. 1, 10).

[Firth 1957] J. R. Firth. "A synopsis of linguistic theory 1930-55." 1952-59 (1957), pp. 1–32 (citado na pg. 5).

[Gemini Team 2025] Gemini Team. *Gemini: A Family of Highly Capable Multimodal Models*. 2025. arXiv: 2312.11805 [cs.CL]. url: https://arxiv.org/abs/2312.11805 (citado na pg. 10).

[Grootendorst 2022] Grootendorst. "Bertopic: neural topic modeling with a class-based tf-idf procedure". *arXiv preprint arXiMaartenv:2203.05794* (2022) (citado na pg. 11).

[Harris 1954] Zellig S. Harris. "Distributional structure". *WORD* 10.2-3 (1954), pp. 146–162. DOI: 10.1080/00437956.1954.11659520. eprint: https://doi.org/10.1080/00437956.1954.11659520. URL: https://doi.org/10.1080/00437956.1954.11659520 (citado na pg. 5).

[Herrmann *et al.* 2022] Moritz Herrmann, Daniyal Kazempour, Fabian Scheipl e Peer Kröger. *Enhancing cluster analysis via topological manifold learning*. 2022. arXiv: 2207.00510 [cs.LG]. URL: https://arxiv.org/abs/2207.00510 (citado na pg. 11).

[Honnibal *et al.* 2020] Matthew Honnibal, Ines Montani, Sofie Van Landeghem e Adam Boyd. "spaCy: industrial-strength natural language processing in python". *arXiv preprint arXiv:2002.06201* (2020). URL: https://arxiv.org/abs/2002.06201 (citado na pg. 18).

[Jelodar *et al.* 2019] Hamed Jelodar *et al.* "Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey". *Multimedia tools and applications* 78 (2019), pp. 15169–15211 (citado na pg. 1).

[Jolliffe e Cadima 2016] Ian T. Jolliffe e Jorge Cadima. "Principal component analysis: a review and recent developments". *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (abr. de 2016), p. 20150202. ISSN: 1364-503X. DOI: 10.1098/rsta.2015.0202. eprint: https://royalsocietypublishing.org/rsta/article-pdf/doi/10.1098/rsta.2015.0202/1381479/rsta.2015.0202.pdf. URL: https://doi.org/10.1098/rsta.2015.0202 (citado na pg. 12).

[Krizhevsky *et al.* 2012] Alex Krizhevsky, Ilya Sutskever e Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. Lake Tahoe, Nevada: Curran Associates Inc., 2012, pp. 1097–1105 (citado na pg. 4).

[Maaten *et al.* 2008] Laurens van der Maaten, Geoffrey Hinton e Yoesoep Rachmad. "Viualizing data using t-sne". *Journal of Machine Learning Research* 9 (nov. de 2008), pp. 2579–2605 (citado na pg. 12).

[McCulloch e Pitts 1943] Warren S. McCulloch e Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". *Bulletin of Mathematical Biophysics* 5 (1943), pp. 115–133 (citado na pg. 3).

[McInnes *et al.* 2018] Leland McInnes, John Healy e James Melville. "Umap: uniform manifold approximation and projection for dimension reduction". *arXiv preprint arXiv:1802.03426* (2018) (citado na pg. 12).

[Mikolov, Chen *et al.* 2013] Tomas Mikolov, Kai Chen, Greg Corrado e Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL]. URL: https://arxiv.org/abs/1301.3781 (citado na pg. 5).

[Mikolov, Sutskever *et al.* 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado e Jeffrey Dean. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. arXiv: 1310.4546 [cs.CL]. url: https://arxiv.org/abs/1310.4546 (citado na pg. 6).

[Radford e Narasimhan 2018] Alec Radford e Karthik Narasimhan. "Improving language understanding by generative pre-training". In: 2018. url: https://api.semanticscholar.org/CorpusID:49313245 (citado na pg. 10).

[Radovanović *et al.* 2010] Miloš Radovanović, Alexandros Nanopoulos e Mirjana Ivanović. "Hubs in space: popular nearest neighbors in high-dimensional data". *J. Mach. Learn. Res.* 11 (dez. de 2010), pp. 2487–2531. issn: 1532-4435 (citado na pg. 11).

[Reimers e Gurevych 2019] Nils Reimers e Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL]. url: https://arxiv.org/abs/1908.10084 (citado na pg. 10).

[Rosenblatt 1958] F. Rosenblatt. *The Perceptron: A Theory of Statistical Separability in Cognitive Systems (Project Para)*. Cornell Aeronautical Laboratory, Inc. Cornell Aeronautical Laboratory, 1958. url: https://books.google.com.br/books?id=7Q4-AQAAIAAJ (citado na pg. 4).

[Searle 1980] John R. Searle. "Minds, brains, and programs". *Behavioral and Brain Sciences* 3.3 (1980), pp. 417–457 (citado na pg. 4).

[Sia *et al.* 2020] Suzanna Sia, Ayush Dalmia e Sabrina J. Mielke. *Tired of Topic Models? Clusters of Pretrained Word Embeddings Make for Fast and Good Topics too!* 2020. arXiv: 2004.14914 [cs.CL]. url: https://arxiv.org/abs/2004.14914 (citado na pg. 11).

[Steinbach *et al.* 2003] Michael Steinbach, Levent Ertöz e Vipin Kumar. "The challenges of clustering high dimensional data". *Univ. Minnesota Supercomp. Inst. Res. Rep.* 213 (jan. de 2003). doi: 10.1007/978-3-662-08968-2_16 (citado na pg. 11).

[Sutskever *et al.* 2014] Ilya Sutskever, Oriol Vinyals e Quoc V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. arXiv: 1409.3215 [cs.CL]. url: https://arxiv.org/abs/1409.3215 (citado nas pgs. 7, 8).

[Thayer 2025] William P. Thayer. *LacusCurtius: A Gateway to Ancient Rome*. penelope.uchicago.edu/Thayer/E/Roman/home.html. Bill Thayer's Web Site, hosted by the University of Chicago. Last updated: 20 October 2025 (according to the main site). 2025. (Acesso em 08/11/2025) (citado na pg. 17).

[Turing 1950] Alan Mathison Turing. "Computing machinery and intelligence". *Mind* 49 (1950), pp. 433–460 (citado na pg. 3).

[Vaswani *et al.* 2017] Ashish Vaswani *et al.* “Attention is all you need”. *Advances in neural information processing systems* 30 (2017) (citado nas pgs. 1, 4, 8).