UNIVERSIDADE DE BRASÍLIA
FACULDADE DE DIREITO

ANTONIO LUCAS NERES DE OLIVEIRA BARROS

RELATÓRIO PROJETO SINAPSES – AGRUPAMENTO POR SIMILARIDADE (SAS)

Brasília – DF
2023

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE DIREITO

RELATÓRIO PROJETO SINAPSES – AGRUPAMENTO POR SIMILARIDADE (SAS)

Antonio Lucas Neres De Oliveira Barros

Professor Orientador: Dr. Henrique Araújo Costa

Monografia apresentada como requisito parcial à obtenção do grau de Bacharel, no Programa de Graduação da Faculdade de Direito da Universidade de Brasília.

Brasília – DF
2023

FOLHA DE APROVAÇÃO

Antonio Lucas Neres de Oliveira Barros

RELATÓRIO PROJETO SINAPSES – AGRUPAMENTO POR SIMILARIDADE (SAS)

Monografia apresentada como requisito parcial à obtenção do grau de Bacharel, no Programa de Graduação da Faculdade de Direito da Universidade de Brasília.

Aprovada em: 13 de junho de 2023.

BANCA EXAMINADORA

_____

Prof. Dr. Henrique Araújo Costa

(Orientador – Presidente)

_____

Me. José Francisco Pereira Notaro

_____

Esp. Daniel Vitor Oliveira Rodrigues

FICHA CATALOGRÁFICA

BB277r Barros, Antonio Lucas Neres de Oliveira

Relatório Projeto Sinapses – Agrupamento por Similaridade (SAS) / Antonio Lucas Neres de Oliveira Barros; orientador Henrique Araújo Costa. -- Brasília, 2023.

52 p.

Monografia (Graduação – Direito) -- Universidade de Brasília, 2023.

1. Direito Digital. 2. Inteligência Artificial aplicada ao Direito. 3. Processamento de linguagem natural. I. Costa, Henrique Araújo, orient. II. Título.

REFERÊNCIA BIBLIOGRÁFICA

BARROS, A. L. N. O. **Relatório Projeto Sinapses — Agrupamento por Similaridade (SAS)**. 2023. 52 f. Monografia Final (Bacharelado em Direito) — Faculdade de Direito, Universidade de Brasília, Brasília, DF, 2023.

Dedico esta monografia a minha mãe (mãinha), Jacira Neres de Oliveira, por todo amor e carinho que sempre recebi, estando perto ou longe, e também por sempre me incentivar a estudar e proporcionar, mesmo quando não tinha condições, a melhor educação, pois ela já sabia sua importância e que se reflete até hoje em minha vida. Obrigado do fundo de meu coração!

AGRADECIMENTOS

Gostaria de agradecer inicialmente a Cris Bottega, que tem sido minha companheira de todas as horas, nos bons e maus momentos, por todo esse percurso na UnB, desde antes da entrada na faculdade, que envolveu depois a troca de curso para Direito, que ocorreu bem no início da pandemia, o tempo sem atividades, o início das aulas remotas (que foram muito boas), retorno presencial (que não foi tão bom assim). A você, que sempre me incentivou e sofreu comigo quando não conseguia escrever, muito obrigado! Sem o seu apoio, eu não concluiria essa graduação.

Gostaria também de agradecer ao Tribunal Regional do Trabalho da 12ª Região (TRT12) por ter concedido a licença capacitação (para espanto de muitos colegas de outros órgãos) para o levantamento de dados de pesquisa e escrita desse TCC. Gostaria de agradecer, em especial a Carlos Mazzi e Valdir Cunha, por terem aprovado e autorizado a abertura do processo. Com certeza, há a visão aqui de que servidores com tempo para investir em capacitação têm condições de produzir mais e melhor. Foi muito importante poder me afastar um pouco do trabalho e poder desenvolver esse trabalho dedicando toda a minha energia à pesquisa. Aproximadamente um mês e meio acabou deixando o cronograma um pouco apertado (o ideal seriam os três meses que estão previstos), mas foi o possível no momento. Com esse ciclo concluído, espero poder contribuir para o TRT12 ainda mais agregando os conhecimentos adquiridos nessa pesquisa.

Não posso finalizar sem deixar de agradecer aos professores e professoras da Faculdade de Direito e aos servidores e servidoras administrativos da UnB, que conseguiram manter um padrão de excelência em tempos tão incertos como foram os últimos anos. Mesmo com tantas mudanças, com um cenário político tão distópico, a Faculdade continuou a funcionar e cumprir a sua missão da UnB de ser uma universidade inclusiva nos qualificando para o exercício profissional em nossa área de formação.

RESUMO

Esta monografia de graduação trata de um relatório sobre o projeto SINAPSES – Agrupamento por Similaridade, desenvolvido pelo Centro de Excelência em Inteligência Artificial (CEIA), com a coordenação do professor Dr. Eliomar Araújo Lima, pesquisador da Universidade Federal de Goiás (UFG), através de carta acordo entre a mesma e o Programa das Nações Unidas para o Desenvolvimento (PNUD), no escopo do Programa Justiça 4.0, realizado em cooperação com o CNJ. Desse projeto, surgiu o KAIROS (*k-means clustering similarity for legal documents*), composto por modelos de inteligência artificial com a finalidade de agrupar processos, não a partir de seus rótulos ou metadados, mas a partir de similaridade entre suas peças. KAIROS obteve uma acurácia de 77,79% contra um conjunto de dados padrão ouro, também elaborado no escopo do projeto SAS. Além dos conjuntos de dados e de inteligência artificial, também será apresentada a Aplicação SAS, com uma interface gráfica para a utilização da solução KAIROS pelos usuários. Nesse trabalho, o leitor será apresentado aos conceitos de inteligência artificial, aprendizado de máquina e processamento de linguagem natural, de modo a entender como foi estruturada a solução trazida nesse projeto. Em seguida, cada etapa do trabalho será apresentada com os algoritmos estudados pela equipe da UFG. Esse trabalho também traz a aplicação entregue, que utiliza os modelos de IA para apresentar os resultados aos usuários. Ao final, são apresentadas a conclusão, com destaque para as entregas do projeto SAS, além de possibilidades de trabalhos futuros, que possam evoluir as soluções já construídas no contexto do projeto SAS.

Palavras-chave: direito digital; inteligência artificial aplicada ao Direito; processamento de linguagem natural.

ABSTRACT

This undergraduate thesis addresses a report on the SAS project – Sinapses - Similarity Clustering, developed by the Center of Excellence in Artificial Intelligence (CEIA), under the coordination of Dr. Eliomar Araújo Lima, a researcher at the Federal University of Goiás (UFG), through a memorandum of understanding between UFG and the United Nations Development Programme (UNDP), within the scope of the Justice 4.0 Program, carried out in cooperation with the National Council of Justice (CNJ). From this project, KAIROS (k-means clustering similarity for legal documents) emerged, composed of artificial intelligence models aimed at grouping legal processes not based on their labels or metadata, but on the similarity between documents. KAIROS achieved an accuracy of 77.79% against a gold standard dataset, also developed within the SAS project. In addition to the datasets and artificial intelligence, the SAS Application will also be presented, featuring a graphical interface for users to utilize the KAIROS solution. In this work, the reader will be introduced to the concepts of artificial intelligence, machine learning, and natural language processing, in order to understand how the solution presented in this project was structured. Then, each stage of the work will be presented along with the algorithms studied by the UFG team. This work also delivers the implemented application, which utilizes AI models to present results to users. Finally, the conclusion is presented, highlighting the achievements of the SAS project, as well as possibilities for future work that could further develop the solutions already built within the SAS project context.

Keywords: digital law; artificial intelligence applied to law; natural language processing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM – Aprendizado de Máquina

BNPR – Banco Nacional de Precedentes

CEIA – Centro de Excelência em Inteligência Artificial da UFG

CNJ – Conselho Nacional de Justiça

CODEX – Repositório de dados do Judiciário

IA – Inteligência Artificial

JSON – *Javascript Object Notation*

KAIROS – k-means clustering similarity for legal documents

PJe – Processo Judicial Eletrônico

PNUD – Programa das Nações Unidas para o Desenvolvimento

SAS – Projeto Sinapses – Agrupamento por Similaridades

UFG – Universidade Federal de Goiás

UnB – Universidade de Brasília

SUMÁRIO

1 INTRODUÇÃO

A inteligência artificial (IA) está revolucionando vários setores e o judiciário não é exceção. Com o avanço das tecnologias de IA, surgem novas oportunidades para melhorar a eficiência do sistema judiciário. A aplicação da IA no judiciário envolve o uso de algoritmos, modelos de aprendizado de máquina e processamento de linguagem natural (PLN) para automatizar tarefas, analisar dados legais, realizar pesquisas jurídicas e até mesmo prever resultados de casos (ZHONG et al., 2020). Em levantamento realizado pelo CNJ em 2022, foram contabilizados 111 projetos de IA, 171% a mais do que no ano anterior, dos quais 63 já estavam em uso ou aptas para uso. No mesmo levantamento, 53 Tribunais informaram desenvolver soluções de IA (CONSELHO NACIONAL DE JUSTIÇA, 2022).

Uma das áreas em que a IA tem um impacto significativo é a automação de tarefas repetitivas e de rotina. Sistemas de processamento de linguagem natural podem ler, analisar e resumir grandes volumes de documentos legais de forma rápida e precisa, reduzindo a carga de trabalho manual e muitas vezes redundantes para os profissionais jurídicos (ZHONG et al., 2020). Além disso, *chatbots* baseados em IA podem fornecer informações básicas e orientação jurídica aos cidadãos, melhorando o acesso à justiça e aliviando a pressão sobre os tribunais (CONSELHO NACIONAL DE JUSTIÇA, 2022).

A IA também desempenha um papel importante na análise de dados legais e pesquisa jurídica. Com a capacidade de processar e compreender grandes conjuntos de dados legais, os algoritmos de IA podem identificar padrões, *insights* e jurisprudências relevantes. Isso pode auxiliar na tomada de decisões, na pesquisa de casos semelhantes e na identificação de precedentes legais, permitindo que os profissionais do direito tenham acesso a informações mais precisas e relevantes. Além disso, a IA tem o potencial de elaborar minutas de sentença, prever resultados judiciais e realizar análise de riscos legais. Por meio de modelos estatísticos e algoritmos de aprendizado de máquina, é possível analisar dados históricos e extrair *insights* sobre as possíveis decisões judiciais em casos semelhantes. Essas previsões podem ser valiosas para advogadas e partes envolvidas em um processo, permitindo que tomem decisões informadas sobre a melhor estratégia legal a adotar (BRAGANÇA, 2021).

Embora a aplicação da IA no judiciário traga promessas de eficiência e aprimoramento, é importante destacar que a tomada de decisões judiciais ainda é uma responsabilidade dos magistrados. Logo, a IA deve ser usada como uma ferramenta de apoio para auxiliar juízes, advogados e profissionais jurídicos, complementando suas habilidades e conhecimentos. Ao adotar a IA de forma ética e responsável, o judiciário pode colher os benefícios dessa tecnologia

inovadora, melhorando a justiça, a acessibilidade e a eficácia do sistema jurídico como um todo (BRAGANÇA, 2021). O CNJ regulamentou os padrões éticos, de governança, de transparência e de auditabilidade para o judiciário na resolução n. 332/2020 (BRASIL, 2023b).

Nesse sentido, no Brasil temos a Plataforma Digital do Poder Judiciário Brasileiro (PDPJ-Br). Essa plataforma é uma iniciativa do Conselho Nacional de Justiça (CNJ) com o objetivo de promover a modernização e a informatização dos serviços judiciários no Brasil. A PDPJ-Br busca integrar os diferentes órgãos do Poder Judiciário, como tribunais, cartórios e demais instituições, por meio de soluções tecnológicas e sistemas eletrônicos. A ideia é promover a automação de processos, o compartilhamento de informações e a agilidade na tramitação de casos judiciais. A plataforma abrange uma série de funcionalidades e serviços, como a disponibilização de sistemas de processo eletrônico, a emissão de certidões e documentos judiciais, a consulta de processos e andamentos, o agendamento de audiências entre outros recursos (BRASIL, 2023a).

No âmbito da IA, o CNJ, através da já citada resolução n. 332/2020, estabeleceu o SINAPSES como a plataforma nacional do Poder Judiciário brasileiro para armazenamento, treinamento supervisionado, controle de versionamento, distribuição e auditoria de modelos de Inteligência Artificial. Essa resolução também definiu os parâmetros para a implementação e funcionamento da plataforma. A gestão e responsabilidade pelos modelos e conjuntos de dados são atribuídas a cada órgão do Poder Judiciário, por meio de seus profissionais técnicos e colaboradores da plataforma. O Departamento de Tecnologia da Informação do Conselho Nacional de Justiça (CNJ) tem a responsabilidade de fornecer a manutenção da Plataforma Sinapses (BRASIL, 2023b).

Um dos problemas na eficiência do judiciário se dá pelo alto volume de processos em trâmite no país. Conforme apontado pelo relatório produzido pelo Conselho Nacional de Justiça (CNJ), Justiça em Números de 2022, havia um total de 77,3 milhões processos em tramitação no Judiciário brasileiro em 2021. Apenas nos últimos 12 meses da elaboração do relatório, foram protocolados 27,7 milhões de processos; Ainda da mesma fonte, constata-se que foram necessários R$ 103,9 bilhões no ano em questão para se manter a estrutura do Judiciário. Nesse sentido, projetos e estratégias que visem diminuir esses números têm o potencial de melhorar a eficiência do sistema judiciário nacional (BRASIL, 2022).

1.1 Projeto SINAPSES – Agrupamento por Similaridades (SAS)

O projeto SINAPSES – Agrupamento por Similaridades (SAS), coordenado pelo professor Dr. Eliomar Araújo Lima, da Universidade Federal de Goiás (UFG), foi realizado pelo Centro de Excelência em Inteligência Artificial (CEIA) através de carta acordo entre a UFG e o Programa das Nações Unidas para o Desenvolvimento (PNUD), como parte do Programa Justiça 4.0, realizado em cooperação com o CNJ.

O escopo do projeto SAS é identificar agrupamentos de processos similares por intermédio da aplicação de técnicas de processamento de linguagem natural (PLN) que realizem a análise do conteúdo de documentos processuais, com ênfase nas petições iniciais e nos seus pedidos. Para isso, foram utilizadas técnicas de aprendizado de máquina não supervisionadas, redes neurais para a construção de um algoritmo que possibilite os agrupamentos. A solução final foi batizada como KAIROS (*k-means clustering similarity for legal documents*) no Fórum Internacional Justiça e Inovação (BRASIL, 2023c). A solução obteve uma acurácia de 77,79% quando executada contra um conjunto de dados padrão ouro, criado e analisado manualmente por especialistas em Direito que participaram do projeto.

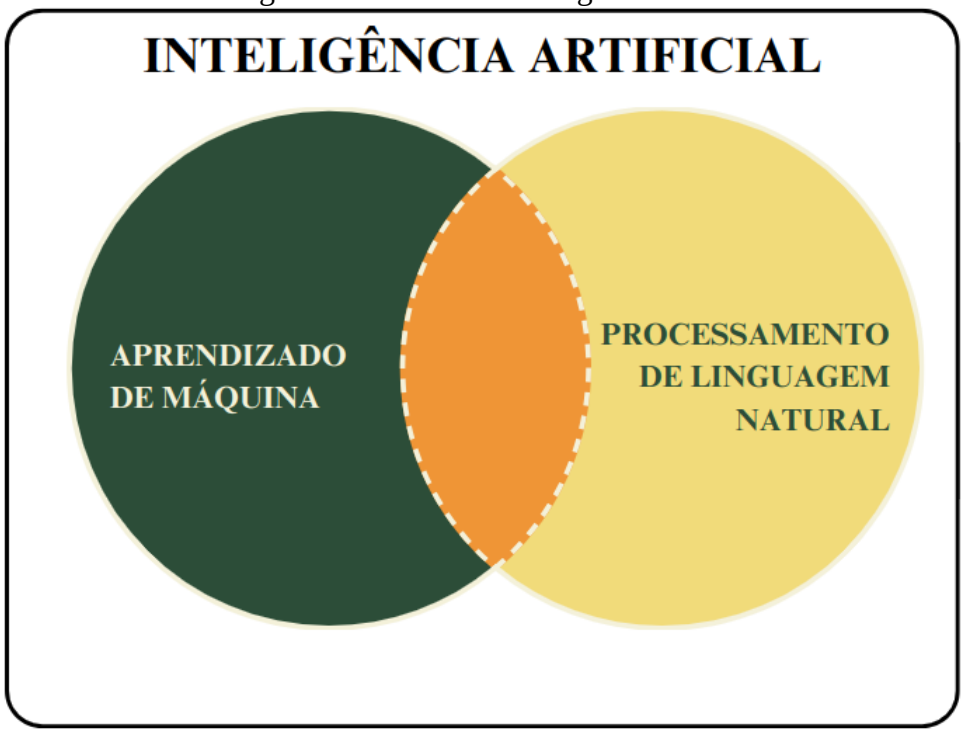
Há diversas aplicações para uma solução nesse sentido, como na lista não taxativa, trazida por (LIMA, 2023): melhorar a gestão do acervo do órgão julgador através de uma classificação por assuntos com base no conteúdo, facilitando o trabalho de pessoas especializadas em determinados temas; realizar busca por jurisprudência similar ao processo atual; Identificar processos semelhantes para tratamento em recurso especial repetitivo ou incidente de resolução de demandas repetitivas; realização de mutirão para julgar processos semelhantes. Ainda, pode-se prever processos com potencial grande de conciliação, de modo a prover uma solução consensual para essas causas; ainda, pode se identificar litispendência, de modo a diminuir o volume processual.

A presente monografia divide-se na seguinte estrutura. O capítulo 2 faz uma apresentação sobre os algoritmos de Inteligência Artificial, aprendizado de máquina e processamento de linguagem natural. A partir do capítulo 3, são apresentados os algoritmos de cada etapa da solução KAIROS que foram implementados no projeto SAS, na seguinte sequência: pré-processamento e sumarização, representação semântica do texto, algoritmos de agrupamento e a interface de usuário da aplicação SAS. Por último, vem a conclusão e os trabalhos futuros que podem ser desenvolvidos a partir do projeto SAS.

2 INTELIGÊNCIA ARTIFICIAL

A inteligência artificial (IA) é uma área da ciência da computação que busca desenvolver sistemas capazes de realizar tarefas que normalmente exigiriam inteligência humana. Através de algoritmos e técnicas avançadas, a IA possibilita que máquinas e sistemas computacionais realizem atividades como reconhecimento de padrões, processamento de linguagem natural, tomada de decisões, aprendizado automático e muito mais. A IA tem o potencial de transformar diversos setores e aspectos da sociedade, incluindo a medicina, transporte, indústria, comércio (MUELLER; MASSARON, 2018). No contexto do judiciário, pode auxiliar na análise de grandes volumes de dados, na automatização de processos, na pesquisa jurídica e até mesmo na tomada de decisões judiciais (ZHONG et al., 2020). A figura a seguir apresenta a relação entre as áreas da IA que serão abordadas no restante do capítulo. Pode-se notar que há uma interseção entre as áreas, mas nenhuma está contida na outra:

Figura 1 — Áreas da Inteligência Artificial



Fonte: elaboração própria.

2.1 Aprendizado de máquina

O aprendizado de máquina (AM) é um subcampo da inteligência artificial (IA) que envolve o desenvolvimento de algoritmos e técnicas que permitem que os sistemas

computacionais aprendam e melhorem de maneira independente, a partir dos dados, sem serem explicitamente programados (RASHKA; MIRJALILI, 2019).

O objetivo do aprendizado de máquina é capacitar os computadores a reconhecer padrões nos dados e tomar decisões ou fazer previsões com base nesses padrões. Em vez de seguir regras rígidas e pré-programadas, os algoritmos de aprendizado de máquina têm a capacidade de aprender e se adaptar aos dados disponíveis (RASHKA; MIRJALILI, 2019). Existem três tipos principais de aprendizado de máquina, explicados com brevidade, a seguir.

O aprendizado supervisionado (*supervised learning)* é aquele no qual o algoritmo é treinado em um conjunto de dados que possui exemplos rotulados. O objetivo é fazer com que o algoritmo aprenda a mapear corretamente as entradas para as saídas desejadas. Por exemplo, um modelo de aprendizado de máquina pode ser treinado para reconhecer imagens de gatos e cachorros com base em um conjunto de imagens previamente rotuladas (GÉRON, 2019).

Já no aprendizado não supervisionado (*unsupervised learning*), o algoritmo é alimentado com um conjunto de dados não rotulados e deve descobrir automaticamente padrões, estruturas ou agrupamentos neles. O objetivo é encontrar *insights* e informações ocultas nos dados. Por exemplo, um algoritmo de agrupamento pode ser usado para identificar grupos semelhantes de clientes com base em seus padrões de compra (GÉRON, 2019).

Por seu turno, o algoritmo de aprendizado por reforço (*reinforcement learning*) aprende por meio de interação com um ambiente. O agente de aprendizado toma ações em um ambiente e recebe *feedback* em forma de recompensa ou punição. O objetivo é aprender uma política de ação que maximize a recompensa ao longo do tempo. Jogos, como o xadrez e o *Go*, são exemplos comuns de aplicação do aprendizado por reforço (FENNER, 2019).

2.1.1 Algoritmos

Os algoritmos de aprendizado de máquina podem ser aplicados em uma ampla variedade de domínios, desde reconhecimento de fala, visão computacional e processamento de linguagem natural até análise de dados, recomendação de produtos, diagnóstico médico. Eles têm a capacidade de lidar com grandes quantidades de dados, encontrar padrões complexos e realizar tarefas que seriam difíceis ou impossíveis de serem programadas manualmente (MUELLER; MASSARON, 2018)

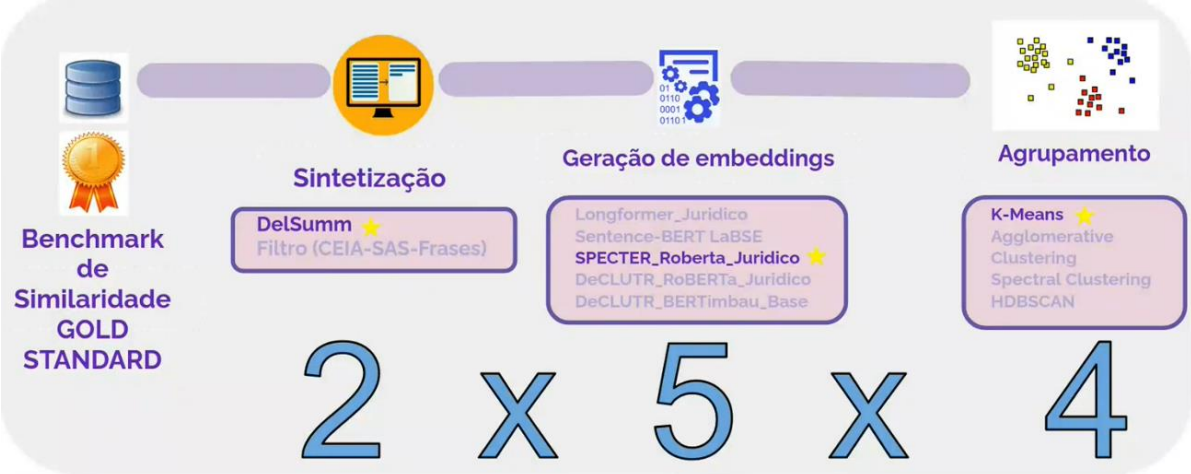
Na sequência, pode-se ver a estrutura em alto nível da solução KAIROS: Há um pré-processamento, que é uma espécie de limpeza do texto; Sintetização, onde se extrai um resumo do texto; *Representação semântica*, onde ocorre a transformação do texto para um formato

entendido por algoritmos, normalmente um vetor de números; e o Agrupamento, que gera o resultado. Na figura seguinte, abaixo de cada fase, estão os trabalhos estudados pela equipe da UFG e que serão apresentados no decorrer deste trabalho. Em cada capítulo será dado destaque para os algoritmos escolhidos para cada tarefa. A escolha se deu através do modelo fatorial, onde todas as alternativas foram testadas e a melhor combinação escolhida.

Figura 2 — Etapas de um algoritmo de PLN

Fonte: (LIMA, 2023).

Figura 3 — Algoritmos escolhidos para cada etapa da solução *KAIROS*



Fonte: (LIMA, 2023).

No processo de aprendizado de máquina, é comum dividir os dados em conjuntos de treinamento e teste. O conjunto de treinamento é usado para treinar o modelo, enquanto o conjunto de teste é usado para avaliar o desempenho do modelo em dados não vistos anteriormente. Esse procedimento é detalhado na próxima seção.

2.1.2 Treinamento

O treinamento em aprendizado de máquina é um processo em que um algoritmo ou modelo é ajustado com base em dados de treinamento, a fim de aprender a fazer previsões ou tomar decisões em relação a novos dados não vistos anteriormente. O objetivo do treinamento é capacitar o modelo a generalizar a partir dos exemplos fornecidos, para que possa fazer

previsões precisas ou tomar ações corretas em situações futuras (RASHKA; MIRJALILI, 2019).

Esse processo geralmente envolve as seguintes etapas: coleta de dados, pré-processamento dos dados, divisão dos dados em conjuntos de treinamento e validação, escolha de um algoritmo ou modelo apropriado, inicialização dos parâmetros do modelo, treinamento iterativo, avaliação do desempenho usando o conjunto de validação e ajuste de hiperparâmetros (RASHKA; MIRJALILI, 2019).

O primeiro passo é coletar um conjunto de dados de treinamento que seja representativo e relevante para a tarefa em questão. Esses dados podem incluir exemplos rotulados, em que as entradas e suas respectivas saídas esperadas são fornecidas, ou dados não rotulados em casos de aprendizado não supervisionado (FENNER, 2019).

Na sequência, os dados de treinamento são pré-processados para garantir a consistência e a qualidade dos dados. Isso pode envolver etapas como limpeza de dados, remoção de ruídos, normalização, tokenização, remoção de *stopwords* e outros processos de transformação dos dados em uma forma adequada para o treinamento do modelo (FENNER, 2019).

Então, os dados de treinamento são divididos em duas partes principais: um conjunto de treinamento e um conjunto de validação. O conjunto de treinamento é usado para ajustar os parâmetros do modelo, enquanto o conjunto de validação é usado para avaliar o desempenho do modelo durante o treinamento (FENNER, 2019).

A partir da tarefa e dos dados disponíveis, um algoritmo ou modelo apropriado é selecionado. Existem várias opções de algoritmos de aprendizado de máquina, como regressão linear, árvores de decisão, redes neurais, entre outros. A escolha do algoritmo depende da natureza do problema e das características dos dados. Com isso, os parâmetros iniciais do modelo são definidos. Dependendo do algoritmo escolhido, esses parâmetros podem ser inicializados aleatoriamente ou por algum método específico, que serão utilizados no treinamento iterativo, onde o modelo é treinado iterativamente usando os dados de treinamento. Durante cada iteração, o modelo faz previsões com base nos dados de treinamento e ajusta seus parâmetros para minimizar a diferença entre as previsões e as saídas corretas. Isso é feito por meio da otimização de uma função de custo ou perda, que quantifica o quão bem o modelo está se ajustando aos dados (VASILIEV, 2020).

Após o treinamento, há a avaliação do desempenho do modelo tendo como base o conjunto de validação. Métricas apropriadas são usadas para medir a qualidade das previsões ou decisões do modelo, como precisão, *revocação*, acurácia ou outras métricas específicas do problema em questão. Com esse resultado, realiza-se o ajuste dos hiperparâmetros, que

controlam o processo de treinamento, como taxa de aprendizado, número de iterações e tamanho (RASHKA; MIRJALILI, 2019).

Durante o treinamento, o modelo faz previsões com base nos dados de treinamento e ajusta seus parâmetros para minimizar a diferença entre as previsões e as saídas corretas. Isso é feito por meio da otimização de uma função de custo ou perda, que quantifica o quão bem o modelo está se ajustando aos dados (VASILIEV, 2020).

O desempenho do modelo é avaliado usando o conjunto de validação, utilizando métricas apropriadas para medir a qualidade das previsões ou decisões. Os hiperparâmetros, que controlam o processo de treinamento, também podem ser ajustados para melhorar o desempenho do modelo. É um processo iterativo em que os parâmetros e hiperparâmetros são ajustados repetidamente até que o modelo alcance um desempenho satisfatório. Uma vez concluído o treinamento, o modelo pode ser aplicado a novos dados para fazer previsões ou tomar decisões com base no que aprendeu durante o treinamento (RASHKA; MIRJALILI, 2019; VASILIEV, 2020).

2.1.3 Métricas de performance

Há diversas métricas comumente usadas para avaliar o desempenho de modelos de aprendizado de máquina. Essas métricas ajudam a medir o quão bem um modelo está executando uma determinada tarefa. A escolha da métrica depende do tipo de tarefa e dos objetivos específicos da aplicação.

A acurácia é uma métrica básica que mede a proporção de predições corretas em relação ao total de predições. É amplamente usada para problemas de classificação, onde o objetivo é atribuir rótulos corretos às instâncias de texto (RASHKA; MIRJALILI, 2019).

A precisão e a revocação são frequentemente usadas em tarefas de classificação binária, como detecção de *spam* ou identificação de sentimentos. A precisão mede a proporção de instâncias classificadas como positivas que são realmente positivas, enquanto a revocação mede a proporção de instâncias positivas corretamente identificadas em relação ao total de instâncias positivas (RASHKA; MIRJALILI, 2019).

A pontuação F1 (também conhecida como F1-*score*) é uma métrica que combina a precisão e a revocação em uma única medida, fornecendo uma visão geral do desempenho do modelo. É a média harmônica entre precisão e revocação e é útil quando há um desequilíbrio entre as classes ou quando ambas as métricas são igualmente importantes (RASHKA; MIRJALILI, 2019).

2.1.4 Conjuntos de dados (*datasets*)

Como visto na seção de treinamento, os conjuntos de dados (do inglês *datasets*) são muito importantes para a resultado final do trabalho. Para se obter modelos com resultados confiáveis é necessário trabalhar sobre os dados para chegar a um conjunto que represente bem a realidade. Nesse sentido, os conjuntos de dados *gold standard* (padrão ouro) desempenham um papel fundamental, pois eles são especialmente criados e anotados manualmente por especialistas humanos para fornecer uma referência confiável e precisa para diversas tarefas no âmbito da inteligência artificial.

Como visto, os conjuntos de dados são usados para treinar algoritmos e modelos de IA. Esses dados são considerados como referências confiáveis para ensinar os modelos a executarem tarefas específicas com precisão. Além disso, os conjuntos de dados *gold standard* permitem a avaliação de desempenho dos algoritmos e modelos e também são fundamentais para a reprodutibilidade e comparação de resultados na pesquisa. Ao fornecer um conjunto de dados bem definido e anotado, outros pesquisadores podem replicar estudos, testar abordagens diferentes e comparar seus resultados com base nas mesmas métricas. Isso promove a transparência e o avanço da pesquisa na área. Por fim, esses conjuntos são frequentemente utilizados para criar *benchmarks* ou referências de desempenho, que são utilizados para comparar diferentes algoritmos, abordagens e modelos.

A criação de um conjunto de dados *gold standard* envolve várias etapas e requer uma abordagem cuidadosa para garantir a qualidade e a confiabilidade dos dados anotados. Começa com uma definição clara do objetivo do conjunto de dados *gold standard*. Deve-se saber quais informações específicas deseja-se anotar; então, deve-se proceder com a coleta dos dados brutos, sejam textos, áudios, etc…; após o pré-processamento, deve-se anotar manualmente os dados. Isso pode incluir marcação de entidades nomeadas, categorização, classificação, conferência dos dados contra os rótulos já existentes, etiquetagem de sentimentos, estruturação gramatical ou qualquer outra anotação necessária. Após a anotação, deve-se revisar e documentar para garantir a representatividade e consistência do trabalho realizado.

No entanto, é importante notar que a criação de conjuntos de dados *gold standard* é um processo intensivo em termos de tempo e recursos humanos. A anotação manual pode ser trabalhosa e requer especialistas no domínio para garantir a qualidade e a consistência dos dados anotados. Portanto, é essencial equilibrar o tamanho do conjunto de dados com a disponibilidade de recursos, para garantir um equilíbrio entre representatividade e viabilidade na criação desses conjuntos.

Visto isso, é importante destacar como uma das entregas valiosas do projeto SAS foram os conjuntos de dados *gold standard* entregues. O primeiro contou com a análise manual de 317 processos judiciais, que foram agrupados de acordo com a similaridade de seu conteúdo, gerando 33 grupos distintos. A partir dele, foi possível calcular métricas de performance para a solução proposta. Além disso, esse dataset pode ser utilizado para comparação do KAIROS com outras soluções que venham a ser criadas futuramente. Além disso, também foram entregues conjunto de dados para entidades e frases, chamados de CEIA-SAS-Entidades e CEIA-SAS-Frases. Eles foram construídos a partir de 300 documentos que foram anotados por 4 especialistaas e revisados. Utilizou-se a medição de concordância entre os pares. Com isso, foram obtidas 50.943 frases anotadas, de 10 classes distintas e 58.049 entidades em 21 classes distintas. Abaixo estão as tabelas de classes de frases e entidades utilizadas, cada uma com o seu respectivo F1-score.

Tabela 1 — Classes utilizadas na criação do conjunto de dados CEIA-SAS-Frases

Classe da Frase	**Pontuação F1**
Argumentação	89,78%
Decisão	83,40%
Definição	85,93%
Exposição	75,37%
Fato	54,62%
Pedido	86,52%
Referência	89,61%
Sem Classe	91,10%
Verbetação[1]	85,26%

Fonte: (LIMA, 2023).

1 Verbetação é o cabeçalho da ementa, ou "… a sequência de palavras-chave e/ou expressões por meio das quais se identificam os assuntos abordados no acórdão… Por exemplo: APELAÇÃO CÍVEL – AÇÃO DE REINTEGRAÇÃO DE POSSE – PRELIMINAR – AUSÊNCIA DE FUNDAMENTAÇÃO DA DECISÃO AGRAVADA – ESPÓLIO – EXERCÍCIO DA POSSE PELOS HERDEIROS – POSSIBILIDADE." (MINAS GERAIS, 2013).

Tabela 2 — Classes utilizadas na criação do conjunto de dados CEIA-SAS-Entidades

Classe da Entidade	Pontuação F1
Atribuição	72,00%
Data do fato	52,73%
Decisão	79,24%
Endereço	89,07%
Função	92,59%
Fundamento	60,10%
Fundamento jurisprudencial	84,19%
Fundamento normativo	93,14%
Local	75,27%
Organização	86,17%
Pedido material	81,05%
Pedido processual	79,35%
Pessoa	96,79%
Princípio	82,03%
Referência doutrinária	74,58%
Reflexo	84,09%
Tipo ação	91,01%
Tribunal	93,64%
Valor causa	91,30%
Valor pedido	89,50%
Vara	69,70%

Fonte: (LIMA, 2023).

2.2 Processamento de Linguagem Natural

Antes de falar do Processamento, é necessário discorrer sobre a linguagem natural. A linguagem natural refere-se à forma de comunicação usada pelos seres humanos para se expressarem verbalmente ou por escrito. É a maneira como as pessoas se comunicam entre si em suas interações diárias, seja em conversas, textos, discursos ou qualquer outra forma de expressão linguística. Ela se caracteriza por sua complexidade e diversidade. Ela possui estruturas gramaticais, vocabulário, regras sintáticas e semânticas específicas que permitem a comunicação efetiva entre as pessoas. Além disso, a linguagem natural é flexível e aberta a interpretações contextuais, sendo influenciada por fatores culturais, sociais e individuais (VAJJALA et al., 2020).

Essa forma de comunicação é composta por elementos como palavras, frases, sentenças e discursos, que podem transmitir significado, intenção, emoção e informação. A linguagem natural também possui características como ambiguidade, polissemia (múltiplos significados), figuras de linguagem e variações regionais e culturais. O estudo da linguagem natural é uma área de pesquisa multidisciplinar que envolve a linguística, a ciência da computação, a psicologia, entre outras disciplinas. Compreender e processar a linguagem natural é um desafio para a inteligência artificial, que busca desenvolver sistemas capazes de interpretar, entender e gerar textos e conversas de forma similar aos seres humanos (VAJJALA et al., 2020).

O Processamento de Linguagem Natural (PLN) refere-se ao campo da inteligência artificial que lida com a interação entre computadores e a linguagem humana. É um ramo da ciência da computação que se concentra em desenvolver algoritmos e técnicas para que as máquinas possam compreender, interpretar, analisar e gerar linguagem natural de maneira semelhante à dos seres humanos. O PLN abrange diversas tarefas, incluindo reconhecimento de fala, compreensão de texto, tradução automática, sumarização de documentos, geração de texto e muito mais. O objetivo final é capacitar as máquinas a processar, entender e utilizar a linguagem humana de forma eficaz e significativa em várias aplicações, como *chatbots* (como o ChatGPT), assistentes virtuais (por exemplo, Alexa, Cortana, Google *Assistant,* Siri, etc...), análise de sentimentos em redes sociais, recuperação de informações (VAJJALA et al., 2020).

O PLN pode ser utilizado para o agrupamento de informações semelhantes. Uma das técnicas comumente utilizadas nesse contexto é a chamada análise de similaridade de texto, que envolve a comparação e o agrupamento de documentos ou trechos de texto com base em sua semelhança. Existem várias abordagens para realizar esse agrupamento, como a comparação de vetores de representação de texto, a análise de tópicos ou até mesmo a aplicação de técnicas de aprendizado de máquina, como algoritmos de *clustering*. Esses métodos permitem identificar textos que possuem temas semelhantes, informações relacionadas ou que abordam os mesmos tópicos. No agrupamento de informações semelhantes, o PLN desempenha um papel importante ao extrair e entender o significado e o contexto dos textos, permitindo assim a identificação e a categorização eficiente de informações relevantes com base em sua similaridade (VAJJALA et al., 2020; VASILIEV, 2020).

Em PLN, um *corpus* refere-se a um conjunto de textos ou dados linguísticos coletados e organizados para fins de análise e estudo. Um *corpus* é uma amostra representativa da linguagem natural que serve como base de dados para o desenvolvimento e treinamento de modelos de PLN. Ele pode ser composto por uma ampla variedade de textos, como livros, artigos de jornais, transcrições de conversas, documentos legais, postagens em redes sociais, entre outros. Esses textos são coletados e estruturados de forma a permitir análises e investigações linguísticas. A criação de um *corpus* requer uma seleção cuidadosa de textos relevantes para o objetivo do estudo. O tamanho e a diversidade do *corpus* podem variar dependendo do projeto em questão. Além disso, é importante que um *corpus* seja anotado ou marcado com informações adicionais, como categorias gramaticais, etiquetas de sentido, entidades nomeadas, entre outros, para permitir uma análise mais aprofundada. Assim, fica claro que o *corpus* desempenha um papel muito importante no desenvolvimento e avanço do

PLN de várias maneiras (VAJJALA et al., 2020). Aqui estão algumas das principais razões pelas quais o *corpus* é importante no PLN:

- Treinamento de modelos: Um *corpus* é usado para treinar modelos de PLN, como modelos de linguagem, modelos de tradução automática, classificadores de sentimentos, entre outros. Os textos do *corpus* são usados para expor o modelo a uma ampla gama de estruturas linguísticas, padrões e contextos, permitindo que ele aprenda a entender e gerar texto de maneira adequada.
- Análise linguística: O *corpus* fornece um conjunto de dados reais para análise linguística. Os pesquisadores podem estudar os textos do *corpus* para identificar padrões, características gramaticais, variações linguísticas, uso de vocabulário, tendências semânticas e outras propriedades linguísticas. Isso ajuda a entender melhor como a linguagem natural é usada e processada.
- Avaliação e benchmarking: O *corpus* é usado para avaliar a qualidade e o desempenho dos modelos de PLN. Ao testar os modelos em um conjunto de dados do *corpus*, é possível medir sua precisão, compreensão e capacidade de gerar respostas corretas. Além disso, os *corpora* também são usados para criar benchmarks, permitindo que diferentes modelos sejam comparados e avaliados com base em um conjunto comum de dados.
- Desenvolvimento de recursos linguísticos: O *corpus* é uma fonte valiosa para o desenvolvimento de recursos linguísticos, como dicionários, ontologias, etiquetadores de partes do discurso, reconhecedores de entidades nomeadas e outras ferramentas de processamento de texto. Os textos do corpus são usados para identificar palavras-chave, relações semânticas, contextos de uso e outras informações relevantes para a construção desses recursos.

3 PRÉ-PROCESSAMENTO E SUMARIZAÇÃO

Nesse capítulo serão apresentadas as duas primeiras fases de um algoritmo de PLN, o pré-processamento e sumarização. A depender do algoritmo, podem ser executadas em conjunto ou sequencialmente.

3.1 Pré-processamento

A fase de pré-processamento no PLN envolve uma série de etapas para preparar os dados linguísticos antes de serem analisados e processados (GANEGEDARA, 2022). Isso inclui diversas tarefas, explicadas sucintamente na sequência:

- Limpeza de dados: Essa etapa envolve a remoção de caracteres indesejados, como pontuação, símbolos especiais ou caracteres não alfabéticos. Também pode envolver a remoção de espaços extras, quebras de linha ou outros caracteres de formatação.
- Conversão de maiúsculas e minúsculas: Em muitos casos, é útil converter todas as letras do texto em minúsculas para evitar duplicações desnecessárias de palavras com maiúsculas e minúsculas diferentes. Isso ajuda a garantir uma correspondência mais consistente nas etapas posteriores.
- Remoção de *stopwords*: *Stopwords* são palavras comuns que geralmente não contêm informações relevantes para a análise, como artigos, preposições e pronomes. A remoção de *stopwords* pode reduzir o tamanho do texto e melhorar a eficiência do processamento.
- Normalização de palavras: Isso envolve a redução de palavras para sua forma base ou canônica. Por exemplo, aplicar *stemming* ou lematização para reduzir as palavras às suas raízes ou formas lematizadas, o que ajuda a agrupar palavras relacionadas.
- Remoção de ruído ou caracteres especiais: Às vezes, é necessário remover caracteres especiais, URLs, números ou outros elementos que possam interferir na análise ou processamento posterior.
- Tokenização: A tokenização é a divisão do texto em unidades menores chamadas de *tokens*. Geralmente, isso é feito com base em espaços em branco, mas também pode envolver regras mais complexas para lidar com contrações, números, URLs, emojis, entre outros casos. Após a tokenização, cada token isolado pode ser processado individualmente nas fases subsequentes da PLN (GANEGEDARA, 2022).

Essas são as tarefas realizadas durante a fase de pré-processamento. A escolha e a ordem dessas etapas podem variar dependendo da aplicação específica, do idioma e dos requisitos do projeto em questão. O objetivo geral é preparar os dados linguísticos de forma consistente e relevante para as etapas de análise e processamento subsequentes do PLN.

3.2 Sumarização

As redes neurais podem apresentar limitações em relação ao tamanho do texto de entrada. Essas limitações estão relacionadas principalmente a dois fatores: recursos computacionais e capacidade de modelagem. Elas exigem recursos computacionais significativos para processar grandes volumes de texto. Quanto maior o texto de entrada, mais memória e poder computacional são necessários para realizar o processamento. Em alguns casos, o tamanho do texto pode exceder os limites de capacidade da memória disponível, tornando difícil ou impossível a utilização de redes neurais para tarefas de processamento de linguagem natural em textos muito extensos (GHIASSI et al., 2012).

As redes neurais têm uma capacidade limitada de modelar e compreender sequências longas de texto. À medida que o tamanho do texto de entrada aumenta, a complexidade da modelagem também aumenta, o que pode resultar em dificuldades de aprendizado e em uma queda no desempenho da rede neural. Além disso, as redes neurais têm uma janela de contexto limitada, ou seja, podem perder informações relevantes em sequências muito longas (GHIASSI et al., 2012).

Para contornar essas limitações, algumas abordagens podem ser adotadas. Uma delas é a divisão do texto em segmentos menores para processamento em lotes, permitindo que a rede neural trabalhe com tamanhos gerenciáveis. Outra estratégia é a utilização de técnicas de sumarização ou extração de informações relevantes para reduzir o tamanho do texto antes de fornecê-lo à rede neural.

A etapa de sumarização ocorre geralmente após o pré-processamento e a tokenização dos dados textuais no *pipeline* de PLN. Após o texto ser dividido em unidades menores, como palavras ou frases, e passar por outras etapas de processamento, como a remoção de *stopwords* e a vetorização, a sumarização é aplicada para resumir um texto longo ou um conjunto de textos de maneira concisa, preservando as informações-chave e a essência do conteúdo original. A sumarização é uma tarefa desafiadora, pois requer a compreensão do texto e a habilidade de selecionar as informações mais relevantes. Existem dois tipos principais de sumarização, sumarização extrativa e abstrativa (BOORUGU; RAMESH, 2020).

Na sumarização extrativa, o objetivo é identificar e extrair frases ou trechos do texto original que sejam considerados mais importantes para compor o resumo. Essas frases ou trechos são selecionados com base em critérios como relevância, importância e representatividade do conteúdo. A sumarização extrativa geralmente envolve a análise estatística das palavras, frases ou outras unidades textuais para determinar sua importância em relação ao texto como um todo. Algoritmos de aprendizado de máquina, como o uso de modelos de linguagem pré-treinados, podem ser empregados para auxiliar nesse processo (BOORUGU; RAMESH, 2020).

Já a sumarização abstrativa, por sua vez, envolve a geração de um resumo mais abstrato e original, onde as frases do resumo não são necessariamente extraídas diretamente do texto original. Nesse caso, a máquina precisa entender e compreender o texto de maneira mais profunda, capturando o significado e a intenção por trás das palavras. A sumarização abstrativa é mais desafiadora, pois requer a capacidade de gerar linguagem natural coesa e fluente, utilizando palavras e estruturas não necessariamente presentes no texto original (BOORUGU; RAMESH, 2020).

Ambos os tipos de sumarização têm suas vantagens e desafios. A sumarização extrativa é geralmente mais simples de implementar e mantém a fidelidade ao texto original, enquanto a sumarização abstrativa tem a capacidade de gerar resumos mais humanos e expressivos, mas requer uma compreensão mais profunda da linguagem (BOORUGU; RAMESH, 2020).

A sumarização tem aplicações amplas, desde a criação de resumos de notícias e artigos científicos até a geração automática de resumos para informações de negócios e revisões de produtos. A tecnologia de sumarização tem avançado com o uso de abordagens baseadas em aprendizado de máquina e modelos de linguagem avançados, permitindo a automação dessa tarefa e facilitando a extração de informações importantes de grandes volumes de texto (BOORUGU; RAMESH, 2020).

3.3 DELSumm

O DELSumm é um algoritmo proposto para a sumarização automática de documentos jurídicos. Ele foi desenvolvido para lidar especificamente com documentos oriundos de casos judiciais e incorporar conhecimento especializado nesse domínio. O algoritmo é do tipo extrativo, ou seja, ele seleciona frases importantes do documento original e as inclui no resumo. O objetivo do DELSumm é produzir um resumo que siga diretrizes específicas definidas por

especialistas legais. Essas diretrizes indicam quais partes do documento devem ser incluídas no resumo e quanto peso cada segmento deve ter no resumo (BHATTACHARYA et al., 2021).

O DELSumm utiliza uma abordagem de programação linear para otimizar a seleção das frases mais informativas e garantir uma representação balanceada dos diferentes segmentos retóricos presentes no documento legal, como fatos do caso, questões legais discutidas e decisão final. Uma das suas vantagens é por se tratar de um algoritmo não supervisionado, o que significa que não requer um conjunto de treinamento prévio com resumos anotados. Mesmo assim, ele demonstrou desempenho comparável ou superior a modelos supervisionados que foram treinados com muitos pares de documentos e resumos (BHATTACHARYA et al., 2021).

A implementação do DELSumm está disponível publicamente no GitHub, o que possibilita seu uso em outros projetos. Isso, aliado a sua flexibilidade, podendo ser adaptado para diferentes jurisdições, o que permitiu seu uso no projeto SAS, o que já era um dos trabalhos futuros propostos pelos autores, que utilizaram inicialmente documentos da Suprema Corte da Índia. É possível ajustar as funções objetivas e restrições do algoritmo para atender às necessidades específicas de cada contexto legal (BHATTACHARYA et al., 2021).

O projeto SAS utilizou o algoritmo DELSumm em conjunto com um filtro que classifica o texto em entidades e frases, criados a partir dos conjuntos de dados CEIA-SAS-Entidades e CEIA-SAS-Frases apresentados no capítulo anterior. A partir desse classificador, o DELSumm foi configurado para utilizar frases do tipo argumentação, exposição e fato como mais relevantes. A ideia inicial era utilizar apenas os fatos, porém a acurácia do algoritmo ficou em 54,62% (havia dissenso até entre os especialistas do direito que anotaram as frases para o conjunto de dados analisado). Com isso, foram adicionadas os outros dois tipos de frase. Para cada uma das frases dos tipos listados, foram destacadas as que contivessem entidades do tipo fundamento jurisprudencial, fundamento normativo, pedido material ou processual e referência doutrinária.

4 REPRESENTAÇÃO SEMÂNTICA DO TEXTO

Após o pré-processamento e sumarização do texto, os tokens resultantes são convertidos em representações semânticas (normalmente vetores, como já visto) que podem ser processadas por algoritmos de IA. Essa conversão é necessária para permitir que os algoritmos processem e compreendam a linguagem humana. Essa etapa é chamada de representação semântica ou vetorização de texto (VASILIEV, 2020).

4.1 Modelos sem treinamento prévio

Em PLN, os algoritmos de aprendizado de máquina geralmente operam em dados numéricos. Na Representação *Bag of Words* (BoW), cada documento ou texto é representado como um vetor em que cada elemento corresponde a uma palavra ou token único. O valor do elemento pode ser a frequência dessa palavra no documento (abordagem *CountVectorizer*) ou uma medida de importância, como a frequência inversa do documento – abordagem TF-IDF – *Term Frequency-Inverse Document Frequency* (SALTON; BUCKLEY, 1988). Suponha que temos os seguintes documentos:

- Documento 1: “O sol está brilhando”.
- Documento 2: “O dia está bonito.”
- Documento 3: “A noite está estrelada.”

Usando a abordagem BoW, podemos criar uma matriz onde cada documento é representado por um vetor contendo a contagem das palavras:

Tabela 3 — Exemplo *Bag of Words*

	O	**sol**	**está**	**brilhando**	**dia**	**bonito**	**a**	**noite**	**estrelada**
Doc 1	1	1	1	1	0	0	0	0	0
Doc 2	1	0	1	0	1	1	0	0	0
Doc 3	0	0	1	0	0	0	1	1	1

Fonte: elaboração própria.

No exemplo acima, cada coluna representa uma palavra única e cada valor na matriz representa a frequência da palavra no respectivo documento.

Outra abordagem para a vetorização são os modelos de incorporação de palavras (*Word Embeddings*): Nessa abordagem, palavras individuais são mapeadas para vetores de números reais de tamanho fixo, também conhecidos como "embeddings". Esses vetores são criados de forma que palavras semanticamente semelhantes tenham representações vetoriais próximas

umas das outras. Exemplos populares de modelos de incorporação de palavras incluem *Word2Vec*, *GloVe* e *FastText*. Segue abaixo um exemplo utilizando o *Word2Vec, onde são atribuídos vetores de números reais a cada palavra* (CHALKIDIS; KAMPAS, 2019). Aqui está um exemplo simplificado usando palavras e vetores de baixa dimensionalidade:

Tabela 4 — Exemplo Word2Vec

Palavra	**Vetor**
O	[0.2, 0.1, -0.3]
sol	[0.4, 0.6, -0.2]
está	[-0.1, 0.8, 0.2]
brilhando	[0.7, 0.3, -0.5]
dia	[0.9, -0.2, 0.1]
bonito	[0.6, 0.4, 0.2]
a	[0.1, -0.6, 0.4]
noite	[-0.3, -0.5, 0.8]
estrelada	[-0.4, -0.3, 0.6]

Fonte: elaboração própria.

Cada palavra é representada por um vetor numérico de dimensão fixa. Ao vetorizar um documento, as palavras são substituídas por seus vetores correspondentes. Por exemplo, o documento 1 “O sol está brilhando” seria vetorizado como a soma dos vetores de suas palavras componentes no respectivo documento: [0.2, 0.1, -0.3] + [0.4, 0.6, -0.2] + [-0.1, 0.8, 0.2] + [0.7, 0.3, -0.5] = [1.2, 1.8, -0.8].

4.2 Modelos pré-treinados

Pode-se trabalhar também com modelos de linguagem pré-treinados: Esses modelos são treinados em grandes quantidades de texto e capturam informações contextuais e semânticas das palavras. Eles podem ser usados para gerar representações vetoriais de sentenças ou documentos inteiros, capturando o significado e a estrutura textual em um espaço vetorial. Esses modelos geram representações vetoriais contextualizadas que capturam informações de contexto e semântica.

4.2.1 Modelos baseados no BERT

Um exemplo desses modelos é o BERT (*Bidirectional Encoder Representations from Transformers*), que é um modelo de linguagem pré-treinado desenvolvido pelo Google que revolucionou o campo do Processamento de Linguagem Natural. Aqui, a vetorização ocorre em

níveis mais granulares, como níveis de sentenças ou trechos de texto, desempenhando um papel fundamental ao gerar *embeddings* contextuais de alta qualidade. Ao contrário de métodos anteriores (*Word2Vec* e *GloVe* por exemplo), que geravam *embeddings* estáticos para palavras individuais, o BERT utiliza um modelo de linguagem baseado em *transformers* para aprender representações contextuais de palavras. Ele é treinado em uma tarefa de "preenchimento de lacunas" em que o modelo recebe um texto com uma palavra ocultada e precisa prever qual é a palavra correta com base no contexto ao redor dela (DEVLIN; CHANG; LEE; TOUTANOVA, 2019).

O processo de pré-treinamento permite que o BERT capture as relações entre as palavras em uma frase ou documento e gere *embeddings* que levam em consideração o contexto em que as palavras aparecem. Isso é particularmente útil para tarefas de PLN, pois palavras podem ter significados diferentes dependendo do contexto em que são usadas. Depois de pré-treinado, o BERT pode ser finamente ajustado em tarefas específicas, como classificação de texto, sumarização ou tradução. Durante o ajuste fino, o BERT é combinado com camadas adicionais específicas da tarefa e é treinado em dados rotulados para aprender a realizar a tarefa específica de maneira mais precisa (DEVLIN; CHANG; LEE; TOUTANOVA, 2019).

Dessa forma, o BERT se destaca na vetorização de texto, gerando *embeddings* ricos em informações contextuais. Isso possibilita uma compreensão mais profunda da semântica do texto e melhora o desempenho em várias tarefas de PLN. Por exemplo, ao alimentar um trecho de texto como "A temperatura está alta hoje", esse modelo gera uma representação vetorial contextualizada para esse trecho, levando em consideração as palavras anteriores e posteriores. O vetor resultante contém informações sobre o significado e a estrutura do trecho de texto.

O *Language-agnostic* BERT *Sentence Embedding* (LaBSE) é uma abordagem que utiliza o modelo BERT para gerar representações vetoriais de sentenças em diferentes idiomas. Ao contrário das versões anteriores do BERT, que foram treinadas principalmente em inglês, o LaBSE é treinado em um conjunto de dados multilíngue, o que o torna capaz de lidar com diferentes idiomas de forma eficaz (FENG et al., 2022).

O objetivo do LaBSE é capturar informações semânticas e sintáticas de sentenças de maneira contextualizada. Ele realiza isso atribuindo um vetor de alta dimensão para cada sentença, onde cada elemento do vetor representa uma característica específica da sentença. Esses vetores de sentença são gerados pela alimentação das sentenças no modelo BERT e capturando a saída da camada especial [CLS] do modelo, que é projetada para fornecer uma representação agregada da sentença (FENG et al., 2022).

A vantagem do LaBSE é que ele produz *embeddings* de sentença que são independentes do idioma, o que significa que as representações vetoriais geradas podem ser usadas para tarefas de processamento de linguagem natural em diferentes idiomas, sem a necessidade de treinar modelos separados para cada idioma. Eles permitem que as informações semânticas e contextuais das sentenças sejam capturadas de forma mais precisa e facilitam a comparação e o agrupamento de sentenças em diferentes idiomas (FENG et al., 2022).

Dessa forma, o LaBSE é uma abordagem que aproveita o poder do modelo BERT para gerar representações vetoriais de sentenças em diferentes idiomas, permitindo uma compreensão mais profunda e contextualizada das sentenças em tarefas de processamento de linguagem natural multilíngue (FENG et al., 2022).

O SPECTER RoBERTa é um modelo de linguagem que combina o SPECTER (COHAN et al. , 2020) e o RoBERTa (*Robustly Optimized* BERT *Pretraining Approach*) para gerar *embeddings* de sentenças e parágrafos de alta qualidade. O SPECTER é um modelo desenvolvido especificamente para tarefas de classificação e aprendizado por transferência. Ele utiliza a arquitetura do BERT para capturar informações contextuais de sentenças e parágrafos. No entanto, em vez de usar uma abordagem de token-level, o SPECTER gera *embeddings* de nível mais alto, representando sentenças ou parágrafos inteiros. O RoBERTa, por sua vez, é uma variação otimizada do BERT que utiliza um processo aprimorado de pré-treinamento. Ele emprega técnicas como treinamento em lote grande, remoção de pré-processamento desnecessário e ajuste de hiperparâmetros para obter resultados melhores em uma ampla variedade de tarefas de processamento de linguagem natural (LIU et al., 2020).

Ao combinar o SPECTER e o RoBERTa, o SPECTER RoBERTa é capaz de gerar *embeddings* de alta qualidade para sentenças e parágrafos, tornando-o especialmente útil para tarefas de classificação, agrupamento, recuperação de informações e outras tarefas de aprendizado de máquina que requerem representações semânticas de texto. Uma das vantagens do SPECTER RoBERTa é que ele pode ser facilmente adaptado para diferentes tarefas por meio de ajustes finos (*fine-tuning*) específicos. Isso permite que o modelo seja treinado em conjuntos de dados de tarefas específicas, refinando ainda mais suas representações e melhorando seu desempenho em tarefas específicas.

O SPECTER RoBERTa Jurídico é uma adaptação do modelo SPECTER RoBERTa especificamente para o contexto jurídico. Ele foi desenvolvido para lidar com documentos legais, como decisões judiciais, contratos e pareceres jurídicos. Seu objetivo é capturar a semântica e o contexto específicos do domínio jurídico, a fim de gerar *embeddings* de sentenças e parágrafos que sejam altamente informativos e relevantes para tarefas relacionadas ao direito.

A adaptação do SPECTER RoBERTa para o contexto jurídico envolve treinar o modelo em conjuntos de dados específicos do domínio jurídico, como documentos legais anotados e corpus jurídicos. Isso permite que o modelo aprenda a representar corretamente as nuances e características únicas da linguagem e do conteúdo jurídico. Ao utilizar o SPECTER RoBERTa Jurídico, os operadores do direito e pesquisadores podem obter representações de texto de alta qualidade e relevantes para o contexto jurídico.

Outra variação testada no projeto SAS foi a união do DeCLUTR (GIORGI et al., 2021) com o RoBERTa Jurídico, que é um modelo desenvolvido especificamente para o processamento de linguagem natural no contexto jurídico. Seu objetivo é extrair informações relevantes e representar de forma eficaz o conteúdo desses documentos legais. Ele foi treinado em grandes conjuntos de dados jurídicos para aprender padrões linguísticos e semânticos específicos do domínio jurídico.

A principal vantagem do DeCLUTR RoBERTa Jurídico é a capacidade de compreender a terminologia e a estrutura dos documentos legais, capturando o contexto jurídico de maneira mais precisa. Ele pode lidar com desafios específicos do domínio, como o uso de linguagem técnica, ambiguidades legais e complexidade da terminologia jurídica. Ao empregar o DeCLUTR RoBERTa Jurídico, os profissionais do direito podem automatizar e acelerar tarefas que anteriormente exigiam uma análise manual extensiva. Isso pode melhorar a eficiência no processamento de documentos legais, pesquisa de jurisprudência, revisão de contratos, análise de riscos legais e várias outras atividades relacionadas ao campo jurídico.

O DECLUTR BERTimbau base é uma variante do modelo BERTimbau, que por sua vez é uma adaptação do BERT para o processamento de linguagem natural em português (SOUZA; NOGUEIRA; LOTUFO, 2020). O DECLUTR BERTimbau base foi especificamente desenvolvido para lidar com tarefas relacionadas ao campo jurídico em língua portuguesa. Assim como outras variantes do BERT, o DECLUTR BERTimbau base é um modelo de linguagem pré-treinado que aprende representações contextualizadas das palavras e frases em um corpus extenso de texto em português. Ele é capaz de capturar o contexto e a semântica dos textos, permitindo uma compreensão mais aprofundada da linguagem.

O treinamento do DECLUTR BERTimbau base envolveu o uso de conjuntos de dados jurídicos em português, como decisões judiciais, leis, pareceres jurídicos e outros documentos legais. Isso permite que o modelo adquira conhecimento específico do domínio jurídico em língua portuguesa, compreendendo a terminologia, as estruturas e as nuances linguísticas presentes nos textos legais. O DECLUTR BERTimbau tem a capacidade de lidar com a especificidade e complexidade da linguagem jurídica em língua portuguesa. Ele pode ajudar a

automatizar e otimizar várias tarefas jurídicas, proporcionando uma compreensão mais precisa e eficiente dos textos legais em português (SOUZA; NOGUEIRA; LOTUFO, 2020).

4.2.2 *Longformer*

O *Longformer* é um modelo de linguagem desenvolvido para lidar com textos longos de maneira eficiente. Ao contrário de modelos tradicionais, como o BERT, que têm restrições de tamanho de entrada devido à sua arquitetura, o *Longformer* foi projetado especificamente para processar sequências mais longas, como documentos ou artigos extensos. Esse algoritmo utiliza uma abordagem chamada "*attention span"* (alcance de atenção) para lidar com textos longos. Ele usa uma espécie de atenção localizada, em vez de uma atenção global, o que significa que ele se concentra em partes específicas do texto que são relevantes para cada palavra em vez de analisar o texto inteiro. Isso permite que o modelo mantenha o desempenho e a eficiência mesmo com entradas mais longas (BELTAGY; PETERS; COHAN, 2020).

A geração de *embeddings* no *Longformer* segue um processo semelhante a outros modelos de PLN. Primeiro, o texto é tokenizado, dividindo-o em unidades menores, como palavras ou subpalavras. Em seguida, esses *tokens* são convertidos em vetores de palavras usando uma tabela de incorporação pré-treinada. Os vetores de palavras podem ser ajustados durante o treinamento ou podem ser fixos, dependendo da abordagem escolhida. Em seguida, o *Longformer* aplica camadas de transformação neural, como camadas convolucionais ou camadas de transformador, para capturar informações contextuais dos *tokens*. Essas camadas ajudam a criar representações ricas e densas dos *tokens*, capturando relacionamentos semânticos e estruturais entre as palavras (BELTAGY; PETERS; COHAN, 2020).

Os *embeddings* gerados pelo *Longformer* podem ser usados em várias tarefas de PLN, como classificação de texto, extração de informações ou sumarização. Eles capturam informações contextuais importantes do texto, permitindo que o modelo entenda e processe melhor a linguagem natural em contextos mais longos. No geral, o *Longformer* é uma abordagem interessante para a geração de *embeddings* em textos longos, permitindo que modelos de PLN lidem de forma eficiente e precisa com documentos extensos (BELTAGY; PETERS; COHAN, 2020).

Em suma, a vetorização de texto permite que algoritmos de aprendizado de máquina processem e comparem documentos de texto, apliquem técnicas de clusterização, classificação ou análise de similaridade. Também é útil para alimentar dados de texto em modelos de aprendizado profundo, como redes neurais. É importante destacar que a representação vetorial

de texto é uma simplificação da riqueza e complexidade da linguagem natural. No entanto, essa transformação em vetores numéricos permite que a informação textual seja utilizada e explorada por algoritmos de aprendizado de máquina, abrindo caminho para uma ampla gama de aplicações em PLN. Após os testes realizados pela equipe da UFG, eles optaram por adotar o SPECTER RoBERTa Jurídico por ter apresentado o melhor desempenho nas combinações testadas no projeto SAS.

5 ALGORITMOS DE AGRUPAMENTO

Algoritmos de agrupamento são amplamente utilizados no Processamento de Linguagem Natural (PLN) para organizar documentos ou textos em grupos com base em suas características semelhantes. Eles ajudam a identificar padrões, tópicos ou temas subjacentes nos dados textuais. O projeto SAS se enquadra, uma vez que objetivou o agrupamento de processos a partir da similaridade dos documentos analisados. Então, faz-se relevante a apresentação dos algoritmos que foram testados no projeto, com destaque para as suas vantagens e desvantagens.

5.1 HDBSCAN

O HDBSCAN (*Hierarchical Density-Based Spatial Clustering of Applications with Noise*) é um algoritmo de agrupamento baseado em densidade que é utilizado no campo do PLN. Ele é uma extensão do DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*), outro algoritmo popular de agrupamento baseado em densidade. Segundo McInnes e Healy (2017), suas principais características são:

- Hierarquia de clusters: O algoritmo HDBSCAN é capaz de construir uma hierarquia de *clusters*, o que significa que ele pode identificar *clusters* de diferentes tamanhos e densidades. Isso permite uma análise mais detalhada e flexível dos dados, já que os clusters podem ter diferentes níveis de coesão.
- Densidade variável: Ao contrário do DBSCAN, onde a densidade é considerada constante, o HDBSCAN leva em conta a variação da densidade dos dados. Ele usa uma abordagem baseada em grafos para capturar essa variação, permitindo que diferentes densidades sejam identificadas e agrupadas adequadamente.
- Detecção de ruído: O HDBSCAN é capaz de identificar pontos de dados como ruído ou outliers, que não pertencem a nenhum cluster. Essa capacidade de detectar e tratar ruído é útil em cenários de PLN, onde é comum encontrar dados não estruturados ou atípicos.
- Parâmetro de estabilidade: O HDBSCAN introduz o conceito de parâmetro de estabilidade, que permite controlar a sensibilidade do algoritmo à variação da densidade. Isso ajuda a obter uma segmentação mais estável dos dados, reduzindo o efeito de flutuações e variações pequenas.

O HDBSCAN é aplicado em PLN em várias tarefas, como agrupamento de documentos, clusterização de tópicos, detecção de comunidades em redes sociais e segmentação de clientes com base em comportamento textual. Sua capacidade de lidar com *clusters* de diferentes

tamanhos e densidades, além de detectar ruído, torna-o uma opção valiosa para análise exploratória e descoberta de padrões em dados textuais.

5.2 Agrupamento Aglomerativo (*Agglomerative Clustering)*

O algoritmo de agrupamento aglomerativo, também conhecido como *agglomerative clustering*, é um método hierárquico amplamente utilizado no campo do PLN. Ele constrói uma hierarquia de *clusters* ao mesclar iterativamente os exemplos em grupos maiores.

O processo de agrupamento aglomerativo começa com cada exemplo como um *cluster* separado. Em cada iteração, os clusters mais próximos são mesclados com base em uma medida de distância, como a distância euclidiana ou a distância do cosseno. A escolha da medida de distância depende da natureza dos dados e da representação utilizada.

A mesclagem de *clusters* é feita seguindo uma estratégia específica, que pode ser baseada na distância mínima, máxima, média ou na ligação de Ward, entre outras. Por exemplo, na estratégia de ligação de distância mínima (*single-linkage*), a distância entre dois clusters é definida como a menor distância entre dois exemplos pertencentes a clusters diferentes. Já na estratégia de ligação de distância máxima (*complete-linkage*), a distância é definida como a maior distância entre os exemplos dos clusters.

Esse processo de mesclagem continua até que todos os exemplos estejam em um único *cluster* ou até que um critério de parada seja alcançado, como um número específico de *clusters* desejado. O resultado é uma árvore hierárquica de *clusters*, também conhecida como dendrograma, que pode ser visualizada e analisada para identificar diferentes níveis de agrupamento.

Uma das principais vantagens do agrupamento aglomerativo é sua flexibilidade em lidar com diferentes formas e tamanhos de clusters, bem como sua capacidade de capturar estruturas hierárquicas nos dados. Além disso, o agrupamento aglomerativo não requer a especificação prévia do número de *clusters*, o que pode ser útil em muitos cenários do PLN. No entanto, o agrupamento aglomerativo pode ser computacionalmente custoso, especialmente para grandes conjuntos de dados, pois requer o cálculo repetitivo de distâncias entre todos os exemplos e clusters. Além disso, a escolha da medida de distância e da estratégia de mesclagem pode influenciar os resultados finais e a interpretabilidade dos clusters obtidos.

5.3 Agrupamento Espectral (*Spectral Clustering)*

O Agrupamento Espectral (*Spectral Clustering*) é um algoritmo que se baseia nas propriedades espectrais de uma matriz de similaridade ou afinidade entre os exemplos do conjunto de dados (NG; JORDAN; WEISS, 2001). É uma abordagem popular em PLN para agrupamento de texto e análise de comunidades em redes sociais. O processo de agrupamento espectral pode ser dividido nas seguintes etapas:

1. Construção da matriz de afinidade: Inicialmente, é construída uma matriz de afinidade que captura as relações de similaridade entre os exemplos. Essa matriz pode ser construída usando medidas de similaridade, como a similaridade do cosseno ou a distância euclidiana, calculadas com base nas características dos exemplos (por exemplo, vetores de palavras).
2. Decomposição espectral: Em seguida, a matriz de afinidade é transformada em uma matriz laplaciana, que é uma representação especializada da estrutura do conjunto de dados. A decomposição espectral é aplicada à matriz laplaciana, resultando em autovetores e autovalores.
3. Redução de dimensionalidade: Os autovetores correspondentes aos menores autovalores são selecionados e utilizados para reduzir a dimensionalidade dos dados. Essa redução é importante para lidar com conjuntos de dados de alta dimensionalidade e melhorar a eficiência computacional.
4. Agrupamento: Após a redução de dimensionalidade, algoritmos de agrupamento, como o K-*means*, são aplicados aos autovetores reduzidos para agrupar os exemplos em *clusters*. A quantidade de *clusters* a serem gerados deve ser especificada previamente ou determinada por técnicas de validação.

O agrupamento espectral é uma abordagem poderosa porque leva em consideração a estrutura global dos dados e pode identificar agrupamentos não lineares. Ele é capaz de capturar relações complexas entre os exemplos e pode superar limitações do K-*means*, como a sensibilidade à inicialização e a forma dos *clusters*. No entanto, o agrupamento espectral também apresenta desafios, como a necessidade de escolher adequadamente a matriz de afinidade, a determinação do número de *clusters* apropriado e o custo computacional associado à decomposição espectral para grandes conjuntos de dados.

Em resumo, o agrupamento espectral é um método poderoso de agrupamento que utiliza a decomposição espectral e a matriz de afinidade para identificar estruturas de agrupamento em conjuntos de dados. É especialmente útil em PLN para tarefas de agrupamento de texto e análise de comunidades.

5.4 K-*means*

O algoritmo K-*means* (K-médias) é um dos algoritmos de agrupamento mais comumente usados no campo do PLN. Ele é um algoritmo de particionamento que agrupa exemplos em K *clusters*, onde K é um número pré-definido. Conforme (AHMED; SERAJ; ISLAM, 2020), o processo de agrupamento pelo K-*means* envolve as seguintes etapas:

1. Inicialização: O algoritmo começa selecionando aleatoriamente K pontos como centroides iniciais, que serão os representantes dos clusters.
2. Atribuição de exemplos: Cada exemplo é atribuído ao cluster cujo centroide[2] é o mais próximo em termos de distância. A distância mais comumente usada é a distância euclidiana, mas outras medidas de distância também podem ser utilizadas, dependendo da natureza dos dados.
3. Atualização dos centroides: Uma vez que todos os exemplos tenham sido atribuídos a um cluster, os centroides dos *clusters* são recalculados. Isso é feito se calculando a média dos exemplos em cada *cluster*, o que atualiza a posição do centroide.
4. Repetição: Os passos 2 e 3 são repetidos iterativamente até que haja pouca ou nenhuma mudança nas atribuições dos exemplos aos *clusters* ou até que seja atingido um critério de parada predefinido, como um número máximo de iterações.

Ao final do processo, os exemplos são agrupados em K *clusters*, onde cada exemplo pertence ao *cluster* cujo centroide é o mais próximo. O K-means tem algumas características importantes a serem consideradas:

- Sensibilidade à inicialização: A escolha dos centroides iniciais pode afetar o resultado final. Inicializações diferentes podem levar a diferentes soluções. Por isso, em algumas situações, pode ser útil executar o algoritmo várias vezes com diferentes inicializações e escolher a melhor solução.
- Dependência do número de *clusters*: O número de *clusters* K precisa ser especificado antecipadamente. Escolher o valor adequado para K é um desafio e pode exigir conhecimento prévio do problema ou técnicas de validação e avaliação do agrupamento.

2 Um centroide é um ponto representativo ou central de um cluster em um algoritmo de agrupamento. Em outras palavras, é uma espécie de "ponto médio" que resume as características dos exemplos que pertencem ao cluster. O centroide é calculado como a média das coordenadas de todos os exemplos pertencentes ao cluster.

- Limitações com formatos de *cluster* complexos: O K-means assume *clusters* convexos e isotrópicos. Portanto, pode ter dificuldade em lidar com *clusters* de formas irregulares ou *clusters* com diferentes tamanhos ou densidades.
- Eficiência computacional: O K-*means* é um algoritmo computacionalmente eficiente e escalável. No entanto, para grandes conjuntos de dados, o custo computacional pode ser alto, especialmente se o número de *clusters* K for grande.

Apesar de suas limitações, o K-*means* é amplamente utilizado em PLN devido à sua simplicidade, eficiência e facilidade de interpretação dos resultados. É aplicado em diversas tarefas, como agrupamento de documentos, classificação de textos não rotulados, análise de sentimentos e recomendação de conteúdo. Ele acabou sendo o algoritmo que apresentou os melhores resultados e utilizado na solução entregue ao CNJ pela UFG.

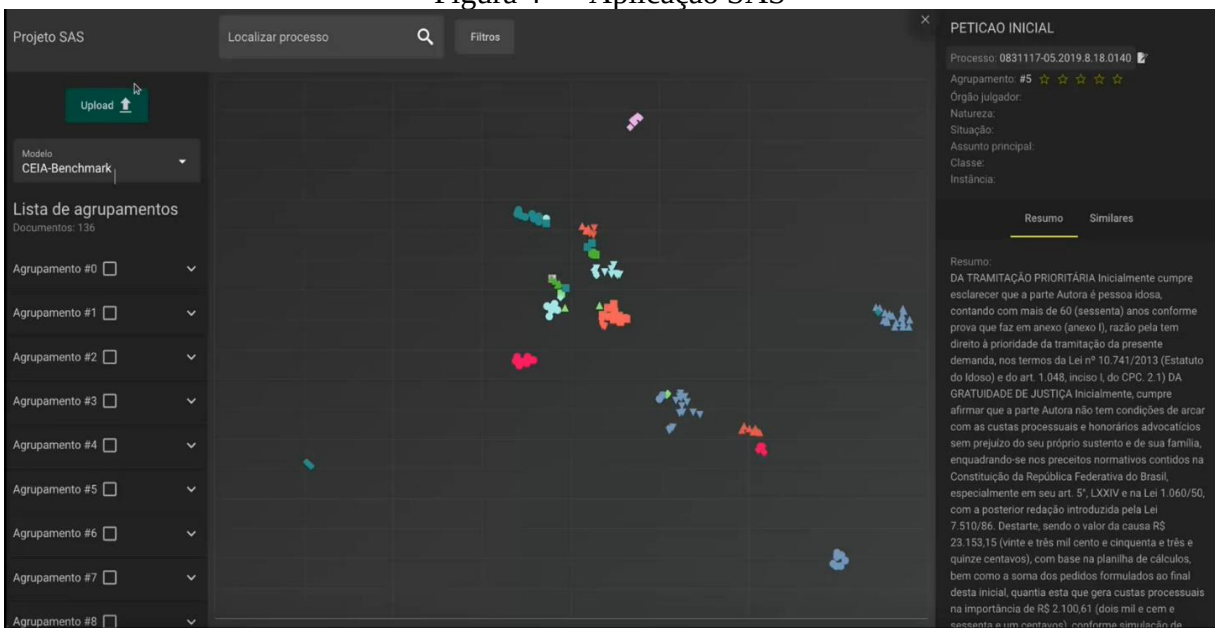
6 Aplicação SAS

Nesse trabalho, até então, foram apresentados os modelos de Inteligência Artificial que foram desenvolvidos no projeto SAS. Porém, para que os usuários possam ter acesso aos resultados e tirem proveito dessa solução, é necessário que haja uma aplicação que apresente os resultados produzidos pela solução KAIROS. Nesse sentido, foi construída a aplicação SAS, apresentada a partir de agora.

6.1 Características/Requisitos

A aplicação produzida disponibilizou um meio para que os usuários possam enviar processos, e, após o processamento, ter uma visão gráfica aos agrupamentos que foram construídos através dos modelos de IA. Com isso, o usuário pode visualizar facilmente quais processos são semelhantes. O envio pode ser realizado através do botão *Upload* (envio) e a visualização em lista fica no lado esquerdo inferior e a gráfica na parte central do sistema, conforme pode ser visto na figura a seguir:

Figura 4 — Aplicação SAS



Fonte: *(LIMA, 2023)*

Além disso, através da ferramenta, no lado direito da Figura acima, é possível acessar os metadados do processo (como o número, órgão julgador dentre outros), além dos resumos que foram produzidos para cada documento, o que possibilita validar se os resumos construídos

realmente contêm informações relevantes sobre os documentos. Ainda, é possível classificar se o agrupamento daquele processo é adequado (até 5 estrelas), assim como visualizar processos similares, com o grau de similaridade. Também pode-se pesquisar por um processo específico e exportar os resultados, de modo que o usuário possa realizar mais análises. Abaixo segue um exemplo de dados exportados pela aplicação que podem ser analisados em maiores detalhes pelos usuários da ferramenta:

Figura 5 — Dados exportados pela aplicação SAS

	chave_documento	id_cluster	class_cluster	classe	competencia
1	7000680-39.2021.8.22.0001_1...	0	[0]	Procedimento do Juizado E	Juizado Especial
2	7000696-61.2019.8.22.0001_1...	0	[0]	Recurso Inominado Cível	Turma Recursal
3	7000701-83.2019.8.22.0001_5...	0	[0]	Recurso Inominado Cível	Turma Recursal
4	7000753-16.2018.8.22.0001_7...	0	[0]	Cumprimento de sentença	Juizado Especial
5	7000786-06.2018.8.22.0001_7...	0	[0]	Procedimento do Juizado E	Juizado Especial
6	7000832-24.2020.8.22.0001_...	0	[0]	Procedimento do Juizado E	Juizado Especial
7	7001438-18.2021.8.22.0001_1...	0	[0]	Procedimento do Juizado E	Juizado Especial
8	7002126-14.2020.8.22.0001_4...	0	[0]	Procedimento do Juizado E	Juizado Especial
9	7002587-83.2020.8.22.0001_7...	0	[0]	Procedimento do Juizado E	Juizado Especial
10	7002795-38.2018.8.22.0001_8...	0	[0]	Recurso Inominado Cível	Turma Recursal
11	7002983-26.2021.8.22.0001_1...	0	[0]	Procedimento do Juizado E	Juizado Especial
12	7003236-14.2021.8.22.0001_1...	0	[0]	Recurso Inominado Cível	Turma Recursal
13	7003304-61.2021.8.22.0001_1...	0	[0]	Recurso Inominado Cível	Turma Recursal
14	7004184-87.2020.8.22.0001_6...	0	[0]	Procedimento do Juizado E	Juizado Especial
15	7004282-09.2019.8.22.0001_5...	0	[0]	Procedimento do Juizado E	Juizado Especial
16	7004345-97.2020.8.22.0001_4...	0	[0]	Procedimento do Juizado E	Juizado Especial
17	7004884-63.2020.8.22.0001_...	0	[0]	Procedimento do Juizado E	Juizado Especial
18	7004959-39.2019.8.22.0001_1...	0	[0]	Procedimento do Juizado E	Juizado Especial
19	7005228-78.2019.8.22.0001_8...	0	[0]	Procedimento do Juizado E	Juizado Especial
20	7005544-23.2021.8.22.0001_1...	0	[0]	Procedimento do Juizado E	Juizado Especial

Fonte: (LIMA, 2023).

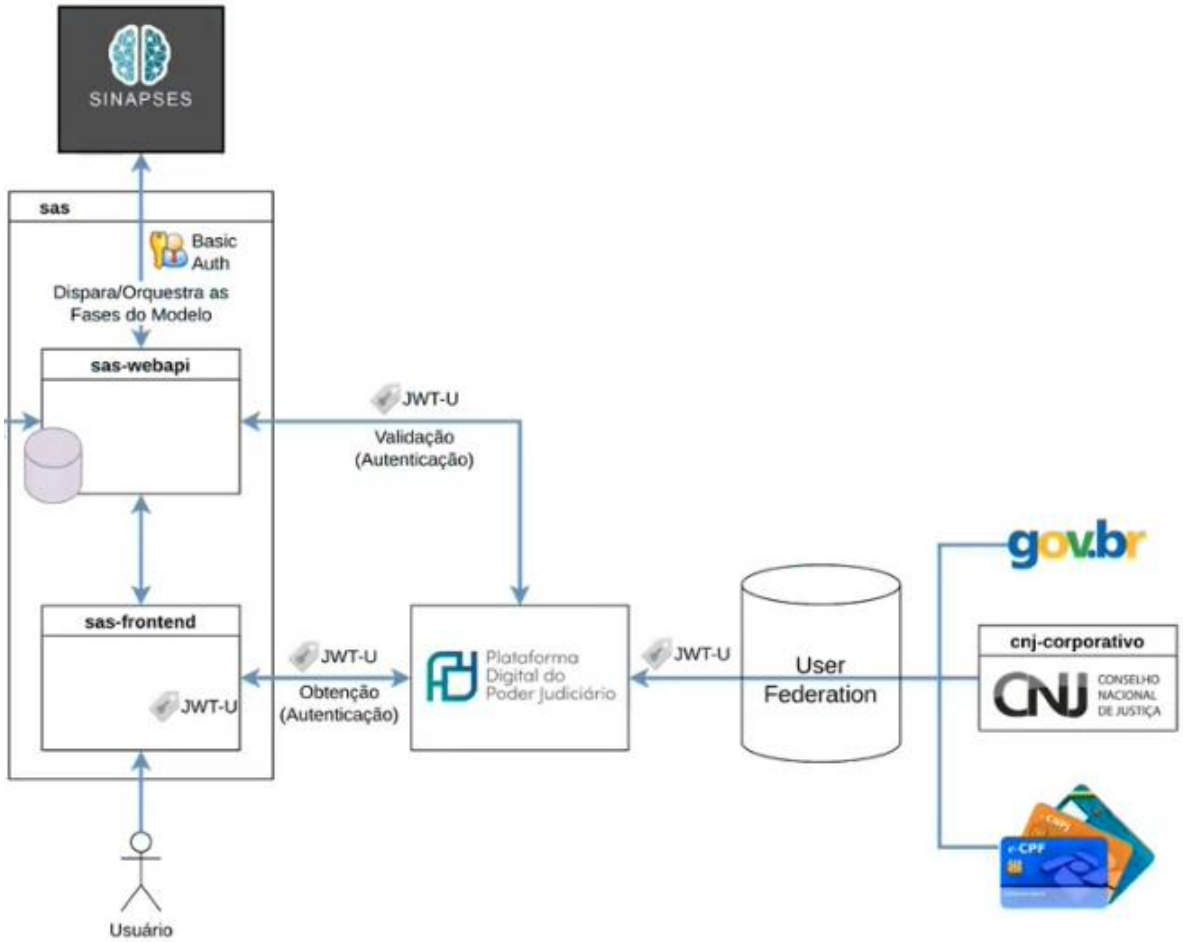
6.1 Integrações

Na atualidade, é difícil que uma aplicação exista sem depender ou consumir informações de outros sistemas. Isso facilita a reutilização de soluções existentes assim como diminui a complexidade da construção de novas aplicações. A aplicação SAS, por exemplo, utiliza a integração com a autenticação única (SSO – *Single Sign On*) da Plataforma Digital do Poder Judiciário Brasileiro – PDPJ-Br. A PDPJ-Br tem como escopo incentivar o desenvolvimento de soluções colaborativas entre os tribunais. Com isso, além de não precisar

gerenciar um cadastro próprio, lidando com permissões e autorizações, a aplicação SAS tem o potencial de possibilitar o acesso a todas as pessoas que já estão cadastradas na PDPJ-Br, aumentando o alcance da solução (BRASIL, 2023a).

Além da integração na autenticação, a aplicação SAS não contém os modelos de IA desenvolvidos nesse projeto. O acionamento é feito através da plataforma SINAPSES, também mantida pelo CNJ. Essa é a plataforma oficial para o armazenamento de soluções de Inteligência Artificial do Poder Judiciário brasileiro. Dessa forma, a aplicação SAS dispara os pedidos e a plataforma SINAPSES realiza o processamento e devolve os resultados. Uma vantagem clara dessa abordagem é que assim como a aplicação SAS, outras aplicações podem utilizar dos modelos construídos nesse projeto e desenvolver suas próprias soluções de Inteligência Artificial de acordo com suas necessidades. A imagem a seguir exibe as integrações realizadas na aplicação SAS:

Figura 6 — Arquitetura da aplicação SAS



Fonte: (LIMA, 2023).

7 CONCLUSÃO

A elaboração de um relatório que aproxime a linguagem técnica da Tecnologia da Informação da linguagem do Direito traz benefícios como um melhor entendimento do que foi realizado assim como tem o potencial de melhorar a colaboração entre as áreas na realização de novos trabalhos nessa área.

O projeto SINAPSES – Agrupamento por Similaridade, desenvolvido pela UFG, com a coordenação do professor Dr. Eliomar Araújo Lima, através de carta acordo entre a universidade e o PNUD, como parte do Programa Justiça 4.0, realizado em cooperação com o CNJ, trouxe diversos resultados, descritos a seguir.

Como grande entrega, destaca-se a solução KAIROS (*k-means clustering similarity for legal documents*), que alcançou o objetivo inicial do projeto, agrupar processos a partir da similaridade textual dos documentos. KAIROS obteve uma acurácia de 77,79% contra um conjunto de dados padrão ouro, que também foi uma das entregas do projeto, descrita ainda nessa seção. É importante destacar que a equipe do CEIA, em vez de realizar uma entrega única do KAIROS, o que dificultaria sua evolução e o reúso do trabalho realizado, disponibilizou cada parte do algoritmo final de maneira individualizada, já integrados à plataforma SINAPSES, de modo que podem ser reutilizados em outros projetos futuros de maneira facilitada.

Também foi entregue uma aplicação com interface gráfica, com integração com o login único da PDPJ-Br, que permite aos usuários submeter documentos de processos e visualizar, analisar e exportar os agrupamentos. Nessa seara, também foram entregues um manual de usuário e a documentação técnica para a implantação da aplicação.

Ainda, foram entregues conjuntos de dados do tipo *gold standard* (padrão ouro). Conforme explicitado nessa monografia, esse tipo de conjunto de dados é difícil de se encontrar por ser muito custosa sua elaboração, de modo que valoriza ainda mais essa entrega. Foram entregues conjuntos de dados de agrupamentos de processos, com 317 processos classificados em 33 grupos, além de agrupamentos de frases e entidades criados a partir de 300 peças processuais. Foram 50.943 frases anotadas, divididas em 10 classes distintas e 58.049 entidades em 21 classes distintas.

Na sequência, são apresentados os trabalhos futuros que podem ser construídos a partir das entregas realizadas no projeto SAS. Salienta-se que se trata de uma lista não taxativa, que pode ser expandida com o tempo e o paulatino ganho de experiência das equipes que venham a atuar em projetos derivados do que foi apresentado nessa monografia.

7.1 Trabalhos Futuros

Ao final do trabalho, é cristalina a importância do projeto SAS no contexto da IA no judiciário brasileiro. As entregas foram diversas e de grande valor. Contudo, como sempre, o trabalho pode evoluir. Com isso, nessa seção, são apresentadas as possibilidades de trabalhos futuros.

7.1.1 KAIROS em diferentes esferas judiciárias

O projeto SAS foi realizado e avaliado com base em um conjunto de processos disponibilizados pelo CNJ, com base no Banco Nacional de Precedentes (BNPR), sem um direcionamento específico para alguma esfera judicial. Dessa forma, novos trabalhos podem avaliar o desempenho da solução KAIROS para cada esfera da justiça. Por conta das especificidades de cada esfera, os algoritmos terão de ser calibrados com novos parâmetros e possivelmente novos conjuntos de dados podem surgir para a realização de treinamentos dos modelos.

7.1.2 Integração com o CODEX

O projeto SAS com sua interface já permite aos usuários realizar agrupamentos a partir de arquivos com formatação técnica específica. Desta forma, a ferramenta pode dificultar o acesso à informação para pessoas sem afinidade com a área de tecnologia. Assim, se faz necessária uma melhoria no sistema, de modo a permitir inserir dados a partir de formulários, sem ter de lidar com padrões de arquivos que não são comuns (como JSON) ou então a integração com alguma ferramenta que já consulte as informações de maneira simples. Nesse último caso, temos o CODEX, que já contém dados dos processos no Judiciário, então o usuário poderia apenas digitar o número dos processos ou do órgão julgador que deseja fazer suas análises. Esse era um dos objetivos no projeto, porém são necessários acesso ao CODEX para viabilizar essa integração, que claramente traz benefícios de usabilidade e aumenta o público que estará apto a utilizar a ferramenta.

7.1.3 Integração com o Processo Judicial Eletrônico (PJe)

Outra integração de grande valia para o judiciário é a integração com o PJe, ferramenta já conhecida por servidores e magistrados, o que incentiva a sua adoção. Contudo, deve-se pensar quais integrações fazem sentido nesse sistema. Lista-se aqui potenciais soluções que podem melhorar a experiência dos usuários do Processo Judicial Eletrônico:

- Pesquisa de processos similares em um mesmo órgão julgador: Esse tipo de integração auxiliará de sobremaneira a gestão das unidades judicantes, uma vez que se torna possível a resolução de determinados temas de maneira especializada ou, em alguns casos, com a reunião de processos que tratem da mesma situação, aumentando a eficiência na prestação jurisdicional.
- Identificação de litispendência: Ao se protocolar um processo, pode-se verificar se, na base do Tribunal, há outro processo com similaridade muito elevada, de modo que se configure a litispendência.
- Pesquisa de processos similares no foro: A pesquisa no foro permitiria identificar litispendência/tentativas de violar o juiz natural que não foram identificadas na origem. Também pode se fazer mutirões de conciliação com base na similaridade dos processos, o que permite uma melhor eficiência dos mediadores.
- Pesquisa de processos similares no Tribunal: A pesquisa total de processos pode ajudar a identificar processos similares em incidentes de resolução de demandas repetitivas (IRDR) de maneira prática, sem onerar magistradas(os) e servidoras(os) do egrégio em questão.
- Previsão de tendência de solução com base na situação fática: Com uma ferramenta de Inteligência Artificial, pode-se identificar o desfecho de processos similares que foram arquivados, indicando assim uma possível tendência para cada processo ainda ativo. Uma aplicação prática para essa funcionalidade seria identificar processos com grande potencial de conciliatório (já que os similares assim se encerraram) para serem levados para a conciliação.
- Resumo de documentos processuais: Essa ferramenta pode utilizar o filtro de frases e o sumarizador já disponíveis na plataforma SINAPSES para resumir peças, como por exemplo, petições iniciais e contestações, para familiarizar a servidora ou magistrada com o processo em questão, o que vai auxiliar na classificação e direcionamento dos processos.

Um ponto que deve ser analisado para as integrações acima é o desempenho da solução, afinal, por exemplo, o protocolo de um processo não pode ficar pendente por conta de

algoritmos que podem ser executados posteriormente. O caso mais crítico, dentre os listados, é o da identificação da litispendência durante o protocolo. Uma estratégia pode ser realizar essa verificação logo após o término do protocolo, deixando claro ao usuário que tal busca será realizada e que o processo pode ser deslocado para outro órgão julgador caso se constate haver litispendência.

REFERÊNCIAS

AHMED, Mohiuddin; SERAJ, Raihan; ISLAM, Syed Mohammed Shamsul. The k-means algorithm: A comprehensive survey and performance evaluation. **Electronics**, [s.l], v. 9, n. 8, p. 1.295, 2020.

BELTAGY, I.; PETERS, M.E.; COHAN, A. Longformer: The Long-Document Transformer. **ArXiv**, [*s.l*], dec. 2020. Disponível em: https://arxiv.org/abs/2004.05150. Acesso em: 31 maio 2023.

BHATTACHARYA, Paheli; PODDAR, Soham; RUDRA, Koustav; GHOSH, Kripabandhu; GHOSH, Saptarshi. Incorporating Domain Knowledge for Extractive Summarization of Legal Case Documents. **18Th International Conference On Artificial Intelligence And Law**. São Paulo: [*s.n*], 2021.

BOORUGU, Ravali; RAMESH, G. A survey on NLP based text summarization for summarizing product reviews. *In*: **2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)**. Coimbatore, India: IEEE, 2020. p. 352-356.

BRAGANÇA, Fernanda. **Justiça Digital**: Implicações sobre a proteção de dados pessoais, solução on-line de conflitos e desjudicialização. Londrina: Editora THOTH, 2021.

BRASIL. Conselho Nacional de Justiça. **Justiça em Números 2022**. Brasília: CNJ, 2022. Disponível em: https://www.cnj.jus.br/pesquisas-judiciarias/justica-em-numeros/. Acesso em: 2 maio 2023.

BRASIL. Conselho Nacional de Justiça. **Plataforma Digital do Poder Judiciário Brasileiro**. Brasília: CNJ, 2023a. Disponível em: https://www.cnj.jus.br/tecnologia-da-informacao-e-comunicacao/plataforma-digital-do-poder-judiciario-brasileiro-pdpj-br/. Acesso em: 21 maio 2023.

BRASIL. Conselho Nacional de Justiça. **Plataforma Sinapses/Inteligência Artificial**. Brasília: CNJ, 2023b. Disponível em: https://www.cnj.jus.br/sistemas/plataforma-sinapses/. Acesso em: 21 maio 2023.

BRASIL. Supremo Tribunal Federal. **Fórum Internacional Justiça e Inovação - FIJI**. Brasília: STF, 2023c. Disponível em: https://portal.stf.jus.br/hotsites/fiji/. Acesso em: 29 jun. 2023.

CHALKIDIS, Ilias; KAMPAS, Dimitrios. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. **Artificial Intelligence and Law**, [s.l], v. 27, n. 2, p. 171-198, june 2019.

COHAN, Arman; FELDMAN, Sergey; BELTAGY, Iz; DOWNEY, Doug; WELD, Daniel. SPECTER: Document-level Representation Learning using Citation-informed Transformers. **Association for Computational Linguistics**: ACL 2020. Cambridge: ACL, v. 1, p. 1430-1445, 2020. Disponível em: https://doi.org/10.48550/arXiv.2004.07180. Acesso em: 10 maio 2023.

CONSELHO NACIONAL DE JUSTIÇA. **Justiça 4.0**: Inteligência Artificial está presente na maioria dos tribunais brasileiros. Texto: Vanessa Maeji. Edição: Márcio Leal. Brasília: Agência

CNJ de Notícias, 2022. Disponível em: https://www.cnj.jus.br/justica-4-0-inteligencia-artificial-esta-presente-na-maioria-dos-tribunais-brasileiros/. Acesso em: 03 jun. 2023.

DEVLIN, Jacob; CHANG, Ming-Wei; LEE, Kenton; TOUTANOVA, Kristina. **BERT**: Pre-training of Deep Bidirectional Transformers for Language Understanding. **ArXiv**, [*s.l*], may 2019. Disponível em: https://doi.org/10.48550/arXiv.1810.04805. Acesso em: 12 jun. 2023.

FENG, Fangxiaoyu; YANG, Yinfei; CER, Daniel; ARIVAZHAGAN, Naveen; WANG, Wei. Language-agnostic BERT Sentence Embedding. **60Th Annual Meeting Of The Association For Computational Linguistics**. Dublin: Association For Computational Linguistics, 2022, p. 878-891.

FENNER, M. E. **Machine Learning With Python For Everyone**. [*s.l.*]: Addison-Wesley, 2019.

GANEGEDARA, Thushan. **Natural Language Processing with TensorFlow**: The definitive NLP book to implement the most sought-after machine learning models and tasks.2 ed.. Birmingham: Packt Publishing Ltd., 2022.

GÉRON, Aurélien. **Mãos à Obra**: Aprendizado de Máquina com Scikit-Learn & TensorFlow. Conceitos, Ferramentas e Técnicas para a Construção de Sistemas Inteligentes. Rio de Janeiro: Starlin Alta Editora, 2019.

DELSUMM: Incorporating Domain Knowledge for Extractive Summarization of Legal Case Documents. **GitHub**. [*s.l*], 2023. Disponível em: https://github.com/Law-AI/DELSumm. Acesso em: 10 jun. 2023.

GHIASSI, M.; OLSCHIMKE, M.; MOON, B.; ARNAUDO, P. **Automated text classification using a dynamic artificial neural network model. Expert Systems with Applications**, v. 39, n. 12, p. 10967-10976, 2012. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2012.03.027. Disponível em: https://www.sciencedirect.com/science/article/pii/S0957417412004976. Acesso em: 03 jun. 2023.

GIORGI, John; NITSKI, Osvald; WANG, Bo; BADER, Gary. DeCLUTR: Deep Contrastive Learning for Unsupervised Textual Representations. **Association for Computational Linguistics**: ACL 2021. Cambridge: ACL, 2021. Disponível em: https://doi.org/10.48550/arXiv.2006.03659. Acesso em: 20 maio 2023.

LIMA, Eliomar Araujo. Projeto CNJ/UFG - Aprendizado não supervisionado na clusterização de petições iniciais. *In*: **Fórum Internacional Justiça e Inovação**. Brasília: STF, 2023.

LIU, Y.; OTT, M.; GOYAL, N.; DU, J.; JOSHI, M.; CHEN, D.; LEVY, O.; LEWIS, M.; ZETTLEMOYER, L.; STOYANOV, V. RoBERTa: A Robustly Optimized BERT Pretraining Approach. **Association for Computational Linguistics**: ACL 2020. Cambridge: ACL, 2020. Disponível em: https://doi.org/10.48550/arXiv.1907.11692. Acesso em: 11 maio 2023.

MCINNES, Leland.; HEALY, John. Accelerated Hierarchical Density Based Clustering. *In*: 2017 **IEEE International Conference on Data Mining Workshops (ICDMW)**. New Orleans, LA: IEEE, 2017. p. 33-42.

MINAS GERAIS. Tribunal de Justiça de Minas Gerais. **Padronização de Ementa no TJMG**. Minas Gerais: TJMG, n.9, 2013. Disponível em: https://www.tjmg.jus.br/lumis/portal/file/fileDownload.jsp?fileId=8A80E40A5C8DD878015C9CF173F75DFF. Acesso em: 15 jun. 2023.

MUELLER, John Paul; MASSARON, Luca. **Artificial Intelligence For Dummies**. Hoboken, New Jersey: John Wiley & Sons, 2018.

NG, Andrew; JORDAN, Michael; WEISS, Yair. On Spectral Clustering: Analysis and an algorithm. *In*: DIETTERICH, Thomas; BECKER, Suzanna; GHAHRAMANI, Zoubin. (ed.). **Advances in Neural Information Processing Systems.** Cambridge: MIT Press, v. 14, 2001. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2001/file/801272ee79cfde7fa5960571fee36b9b-Paper.pdf.

RASHKA, Sebastian; MIRJALILI, Vahid. **Python Machine Learning:** Machine Learning and Deep Learning With Python, Schikit-Learn and TensorFlow 2. 3 ed. Birmingham - Mumbai: Packt Publishing, 2019.

SALTON, Gerard; BUCKLEY, Chris. Term-weighting approaches in automatic text retrieval. **Information Processing & Management**, v. 24, n. 5, p. 513-523, 1988.

SOUZA, Fábio, NOGUEIRA, Rodrigo, LOTUFO, Roberto. BERTimbau: Pretrained BERT Models for Brazilian Portuguese. *In*: CERRI, R., PRATI, R.C. (ed.). **Intelligent Systems**. Rio Grande: Springer, 2020. (Lecture Notes in Computer Science, v. 12319). Disponível em: https://doi.org/10.1007/978-3-030-61377-8_28. Acesso em: 1 junho 2023.

VAJJALA, Sowmya *et al*. **Pratical Natural Language Processing**: Comprehensive Guide to Building Real-World NLP Systems. [*s.l*]: O'Reilly Media, 2020.

VASILIEV, Yuli. **Natural Language Processing With Python and SpaCy**: A Pratical Introduction. San Francisco: [*s.n*], 2020.

ZHONG, Haoxi; XIAO, Chaojun; TU, Cunchao; ZHANG, Tianyang; LIU, Zhiyuan; SUN, Maosong. How Does NLP Benefit Legal System: A Summary of Legal Artificial Intelligence. **The 58th Annual Meeting of the Association for Computational Linguistics**. [s.l]: Association for Computational Linguistics, 2020. Disponível em: https://doi.org/10.48550/arXiv.2004.12158. Acesso em: 13 jun. 2023.