UNIVERSIDADE ESTADUAL DO MARANHÃO

CENTRO DE CIÊNCIAS TECNOLÓGICAS

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO E SISTEMAS

MESTRADO PROFISSIONAL EM ENGENHARIA DE COMPUTAÇÃO E SISTEMAS

THYAGO MACHADO RODRIGUES

SEGMENTAÇÃO DE TEXTOS JURÍDICOS: ESTUDO DE CASO EM PETIÇÕES

2026

UNIVERSIDADE ESTADUAL DO MARANHÃO

CENTRO DE CIÊNCIAS TECNOLÓGICAS

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO E SISTEMAS

MESTRADO PROFISSIONAL EM ENGENHARIA DE COMPUTAÇÃO E SISTEMAS

SEGMENTAÇÃO DE TEXTOS JURÍDICOS: ESTUDO DE CASO EM PETIÇÕES

THYAGO MACHADO RODRIGUES

Trabalho apresentado ao curso de Mestrado Profissional em Engenharia da Computação e Sistemas na Universidade Estadual do Maranhão como pré-requisito para obtenção do título de Mestre sob orientação do(a) Prof. Dr. Antonio Fernando Lavareda Jacob Jr. e Coorientação do(a) Prof. Dr. Fábio Manoel França Lobato

2026

Rodrigues, Thyago Machado
Segmentação de textos jurídicos: estudo de caso em petições. / Thyago Machado Rodrigues. – São Luis, MA, 2026.
64 f

Dissertação (Mestrado em Engenharia da Computação e Sistemas) - Universidade Estadual do Maranhão, 2026.

Orientador(a): Prof. Dr. Antonio Fernando Lavareda Jacob Junior,.

1.Segmentação Textual. 2.Justiça 4.0. 3.Redes Neurais. I.Titulo.

CDU: 347.931:004.8

Elaborado por Cássia Diniz - CRB 13/910

Thyago Machado Rodrigues

Segmentação de Textos Jurídicos: estudo de caso em Petições

Trabalho apresentado ao curso de Mestrado Profissional em Engenharia da Computação e Sistemas na Universidade Estadual do Maranhão como pré-requisito para obtenção do título de Mestre sob orientação do(a) Prof. Dr. Antonio Fernando Lavareda Jacob Jr. e Coorientação do(a) Prof. Dr. Fábio Manoel França Lobato

São Luís - MA, 28 de março de 2026:

gov.br Documento assinado digitalmente
ANTONIO FERNANDO LAVAREDA JACOB JUNIOI
Data: 31/03/2026 17:06:01-0300
Verifique em https://validar.iti.gov.br

Profº. Dr. Antonio Fernando Lavareda Jacob Junior
(Orientador - UEMA)

gov.br Documento assinado digitalmente
FABIO MANOEL FRANCA LOBATO
Data: 31/03/2026 17:20:37-0300
Verifique em https://validar.iti.gov.br

Profº. Dr. Fábio Manoel França Lobato
(Coorientador - UFOPA)

gov.br Documento assinado digitalmente
OMAR ANDRES CARMONA CORTES
Data: 01/04/2026 02:00:22-0300
Verifique em https://validar.iti.gov.br

Profº. Dr. Omar Andres Carmona Cortes
(Examinador Interno - IFMA)

gov.br Documento assinado digitalmente
RICARDO MARCONDES MARCACINI
Data: 01/04/2026 07:33:31-0300
Verifique em https://validar.iti.gov.br

Profº. Dr. Ricardo Marcondes Marcacini
(Examinador Externo - USP)

São Luís - MA

2026

RESUMO

O crescente volume de processos judiciais evidenciado no relatório de 2024 do Conselho Nacional de Justiça sublinha a urgência de estratégias eficazes para triagem e análise documental. Com 83,8 milhões de processos pendentes até o final de 2023, o Poder Judiciário enfrenta um grande desafio em termos de gestão e tomada de decisões sobre esse vasto conjunto de documentos. Esses documentos, em sua maioria textuais, podem ser vistos como uma sequência de segmentos semanticamente coerentes, concebidos para facilitar a transição entre diferentes subtópicos dentro de um único documento. Dessa forma, dividir um documento em segmentos semelhantes pode ajudar a melhorar e/ou acelerar aplicações *downstream*. Nesse sentido, a segmentação de petições iniciais em tópicos nomeados de Fato, Tese e Pedido (FTP) emerge como uma abordagem promissora. Sendo assim, este estudo propõe a construção de um segmentador de petições iniciais, utilizando redes neurais juntamente com incorporações provenientes do modelo *Bidirectional Encoder Representations from Transformers* em um banco de dados de petições a serem manualmente anotadas, acobertado pela metodologia *Cross Industry Standard Process for Data Mining*. O objetivo reside em realizar a identificação automatizada da tripla FTP, a fim de fornecer os trechos mais relevantes da petição no contexto jurídico. Além disso, a abordagem proposta é passível de adaptação para outras tarefas de processamento de linguagem natural, como sumarização e busca textual. Ao concentrar-se nos elementos essenciais para análise e resposta, espera-se reduzir o volume de informação a ser processada, otimizando assim a eficiência operacional. Além disso, a divisão da petição em seções distintas facilita a identificação rápida e precisa dos elementos-chave, acelerando o processo de compreensão, particularmente quando empregado em conjunto com sistemas automatizados e soluções oriundas do programa Justiça 4.0.

Palavras-chaves: Segmentação Textual, Justiça 4.0, Redes Neurais.

ABSTRACT

The growing volume of legal cases highlighted in the 2023 report of the Conselho Nacional de Justiça highlights the urgency of effective strategies for screening and document analysis. With 81.4 million cases pending at the end of 2022, the Judiciary faced a major challenge in terms of management and decision-making regarding this vast set of documents. These documents, mostly textual, can be seen as a sequence of semantically coherent segments, designed to facilitate the transition between different subtopics within a single document. In this way, dividing a document into similar segments can help improve and/or speed up *downstream* applications. In this sense, the segmentation of initial petitions into descriptions named fact, thesis and request (FTP) emerges as a promising approach. Therefore, this study proposes the construction of an initial petition segmenter, using neural networks together with incorporations from the *Bidirection Encoder Representations from Transformers* model in a database of petitions to be annotated manually, discovered by *Cross Industry Standard Process for Data Mining* methodology. The objective lies in performing automated identification of the FTP triplet in order to provide the most relevant parts of the petition in the legal context. Furthermore, the proposed approach can be adapted to other natural language processing tasks, such as summarization and textual search. By focusing on the essential elements for analysis and response, it is expected to reduce the volume of information to be processed, thus optimizing operational efficiency. Furthermore, the division of the petition in particular facilitates the quick and accurate identification of key elements, accelerating the understanding process, especially when sent in conjunction with automated systems and solutions from the Justice 4.0 program.

Key-words: Text Segmentation, Justice 4.0, Neural Networks.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BERT	*Bidirectional Encoder Representations from Transformers*
BiGRU	*Bidirectional Gated Recurrent Unit*
BiLSTM	*Bidirectional Long-Short Term Memory*
CNN	*Convolutional Neural Networks*
CPC	Código do Processo Civil
CRISP-DM	*Cross-Industry Standard Process for Data Mining*
CSV	*Comma Separated Values*
FRNN	*Fast Recurrent Neural Network*
FTP	Fato, Tese e Pedido
GRU	*Gated Recurrent Unit*
IA	Inteligência Artificial
KNN	*K-Nearest Neighbors*
LincProg	Laboratório de Inteligência Computacional
LSTM	*Long-Short Term Memory*
MLM	*Masked Language Modeling*
NSP	*Next Sentence Prediction*
OCR	*Optical Character Recognition*
PTR-Net	*Pointer Networks*
ReLU	*Rectified Linear Unit*
RNA	Rede Neural Artificial
RNN	*Recurrent Neural Network*
SVM	*Support Vector Machine*
TF-IDF	*Term Frequency-Inverse Document Frequency*
TJMA	Tribunal de Justiça do Estado do Maranhão
UEMA	Universidade Estadual do Maranhão

SUMÁRIO

1 INTRODUÇÃO

Este capítulo busca tratar dos temas intrinsecamente ligados a necessidade do uso de técnicas de inteligência artificial no contexto do judiciário nacional. Para tal, o mesmo está estruturado da seguinte forma: contextualização de assuntos em que o trabalho está associado, justificativa, objetivos propostos, resultados esperados e, por fim, a organização geral do estudo.

1.1 CONTEXTUALIZAÇÃO

O Poder Judiciário brasileiro encerrou o ano de 2023 com um estoque de 83,8 milhões de processos pendentes (CNJ, 2024), evidenciando uma realidade marcada pela sobrecarga de documentos jurídicos e pela complexidade crescente de sua análise. A informatização dos tribunais, impulsionada pela ampla adoção de sistemas eletrônicos de processo, como o Processo Judicial Eletrônico (PJe), tem gerado um aumento exponencial na quantidade de dados textuais produzidos diariamente, exigindo esforços intensificados na triagem, categorização e tomada de decisão dessas informações (PORTO, 2022). Nesse contexto, surge a necessidade de incorporar tecnologias capazes de auxiliar os operadores do direito na gestão desse volume significativo de conteúdo textual, integrando áreas como ciência de dados, inteligência artificial e linguística computacional ao universo jurídico (GUIMARÃES; ROCHA; MUGNAINI, 2023).

Essa quantidade excessiva de processos sem uma resolução adequada promove a morosidade e compromete a confiança no sistema judicial (MAGALHÃES; FREITAS, 2023). A deficiência na celeridade do sistema judicial gera incerteza no ambiente econômico ao inibir investimentos nacionais e estrangeiros (MELCARNE; RAMELLO; SPRUK, 2021) bem como alimenta o sentimento de insegurança diante da sensação de impunidade decorrente da demora na aplicação da lei, afetando diretamente a confiabilidade sobre o sistema legal, tanto em relação à capacidade de proteger os direitos dos cidadãos quanto à eficácia das penas impostas aos seus infratores (GRANGEIA, 2011).

Considerando os reflexos negativos na sociedade, têm-se apostado em estratégias voltadas para a desburocratização e simplificação de processos decisórios (GRANGEIA, 2011) sobretudo na automatização de processos, envolvendo as Tecnologias da Informação e Comunicação (TICs) (NETTO; CAMPAGNOLI; GARCIA, 2021). Uma das principais alternativas que visam a desburocratização dos processos em prol da eficiência do serviço público foi o desenvolvimento do sistema PJe, que permitiu o acompanhamento *online* da tramitação dos processos, propiciando a participação mais efetiva e transparente dos profissionais que dele fazem uso (TEIXEIRA; RÊGO, 2017). Dessa forma, a digitalização

do processo judicial tem sido fundamental na modernização da justiça, possibilitando maior agilidade e eficiência nas atividades judiciárias (RAMPIM; IGREJA, 2022).

Como consequência, a digitalização do judiciário proporcionou um ambiente favorável para o desenvolvimento de soluções focadas no uso da tecnologia, em especial o programa Justiça 4.0 do Conselho Nacional de Justiça. Lançado em fevereiro de 2021, o programa visa desenvolver ações e estratégias para ampliar o leque de ferramentas de apoio jurídico com o objetivo de facilitar o acesso à justiça no país (BRAGANÇA, 2023). Além disso, como uma iniciativa de aperfeiçoamento dos serviços jurisdicionais, busca não apenas modernizar os procedimentos judiciais, mas também tornar o judiciário mais acessível e menos complexo, reduzindo a morosidade nos processos (SCHNEIDER; MOREIRA, 2023; ROCHA; RIBEIRO; JEVAUX, 2023). Essas iniciativas têm apresentado impactos significativos no fluxo processual dos tribunais na medida em que impulsionam a celeridade e permitem uma comunicação mais eficiente, reduzindo, por consequência, a morosidade judicial (ROCHA; RIBEIRO; JEVAUX, 2023).

Como uma das vertentes da Justiça 4.0, a Inteligência Artificial (IA) surge como uma tecnologia com potencial para impactar significativamente os julgamentos e as relações de trabalho no Judiciário, especialmente diante do congestionamento processual e da redução paulatina da força de trabalho (NETTO; CAMPAGNOLI; GARCIA, 2021). No contexto jurídico, a IA já está presente por meio de diversos artefatos voltados ao suporte de atividades jurídicas (MACHADO; COLOMBO, 2021), como o projeto Elis, dedicado à triagem de processos; o Victor (FILHO; JUNQUILHO, 2018) e Firmina, responsáveis pela classificação de textos jurídicos; e o Athos, que realiza a clusterização de documentos (TAUK; SALOMÃO, 2023).

Diante do cenário de modernização tecnológica do Judiciário e da crescente sobrecarga processual, o Tribunal de Justiça do Maranhão (TJMA) e a Universidade Estadual do Maranhão (UEMA) firmaram, em março de 2021, um Acordo de Cooperação Técnica para desenvolver soluções de software baseadas em Inteligência Artificial e automação de rotinas, buscando maior eficiência e celeridade processual no Judiciário estadual. Como um de seus primeiros resultados, o projeto Firmina surge como uma das frentes da cooperação, voltado à automatização da leitura e análise de petições iniciais, enquadrando-as dentre os precedentes mais recorrentes do tribunal do Maranhão.

Dentro dos artefatos de IA no âmbito jurídico, o robô Firmina utiliza como insumo as petições iniciais dos processos jurídicos. No entanto, em função das características do documento, faz-se necessário quebrá-lo em partes específicas com o objetivo de aprimorar a qualidade de sua entrada. Nesse sentido, o foco do trabalho reside na construção de uma ferramenta baseada em IA para lidar com a segmentação automática de petições iniciais, considerando diferentes formatos e *templates* de escrita. Para isso, foi adotada a metodologia *Cross-Industry Standard Process for Data Mining* (CRISP-DM) para estruturar o processo

de desenvolvimento, que orientou desde a compreensão do problema e preparação dos dados até a modelagem, avaliação e implantação da solução. A anotação dos dados foi realizada para identificar os segmentos de fatos, tese e pedido, possibilitando a criação de um conjunto de treinamento específico.

A etapa de modelagem envolveu o uso de diferentes arquiteturas de redes neurais, incluindo *Long-Short Term Memory* (LSTM) para captura de dependências temporais, Redes de Ponteiros para delimitação dos limites de segmentos, incorporações *Bidirectional Encoder Representations from Transformers* (BERT) para representação semântica dos textos e perceptrons multicamadas para classificação final dos trechos. Essa combinação de técnicas possibilitou explorar tanto aspectos sintáticos quanto semânticos dos documentos, com o objetivo de alcançar segmentações mais precisas e generalizáveis.

1.2 JUSTIFICATIVA

Um dos principais desafios no processamento de linguagem natural é a interpretação de textos de domínios específicos, como o jurídico, e sua transformação em representações computacionalmente tratáveis (POLO et al., 2021). Nesse contexto, Processamento de Linguagem Natural (PLN) surge como uma área de Ciência da Computação que estuda o desenvolvimento de programas de computador que analisam, reconhecem e/ou geram textos em linguagens humanas (VIEIRA; LOPES, 2010). Embora PLN esteja comumente relacionada a texto, também costuma tratar computacionalmente diversos aspectos da comunicação humana, como sons, palavras, sentenças e discursos, considerando formatos e referências, estruturas e significados, contextos e usos (GONZALEZ; LIMA, 2003).

Tarefas que envolvem PLN não são triviais devido à rica ambiguidade da linguagem natural. Essa ambiguidade torna o PLN diferente do processamento das linguagens de programação de computador, as quais são formalmente definidas de maneira a evitar a ambiguidade (VIEIRA; LOPES, 2010). Portanto, ao trabalhar com processamento de texto, é necessário delimitar um conjunto de palavras ou trechos potencialmente importantes (VALE, 2019). Esse tipo de tratamento pode ser feito por meio da segmentação textual, a qual é o processo de segmentar um texto em unidades menores, que podem ser cláusulas, orações, sentenças, parágrafos ou tópicos (PARDO; NUNES, 2003). O processo de segmentação é uma tarefa típica na fase de pré-processamento de ferramentas de PLN, sendo essencial para que essas ferramentas possam realizar suas funções específicas, como rotulação morfossintática, análise sintática, alinhamento textual, análise retórica, sumarização, tradução automática, entre outras (PARDO, 2006).

A segmentação textual, no entanto, pode ocorrer em diferentes níveis de granularidade, sendo quatro os principais (PAK; TEH, 2017): i) nível de palavra, como no reconhecimento de entidades nomeadas (LI et al., 2021); ii) nível de sentença, que divide

o texto em unidades discursivas ou sintáticas (LI et al., 2020); iii) nível de frase, que isola orações coordenadas ou subordinadas para análises mais finas (AUMILLER et al., 2021); e iv) nível de tópico, que agrupa blocos temáticos coerentes, útil em tarefas como sumarização e detecção de mudanças de assunto (JÚNIOR; CALIXTO; CASTRO, 2020). O presente estudo se concentrará na segmentação em tópicos textuais.

A segmentação de tópicos é o processo de dividir um texto em segmentos semanticamente coerentes, e a rotulagem de segmentos envolve atribuir um rótulo de tópico a cada um desses segmentos (XIA; WANG, 2023). A segmentação de tópicos não é uma tarefa de caráter trivial, tendo em vista a identificação e rotulação de tópicos também é complexa para humanos. Pessoas diferentes podem detectar diferentes limites de contexto, perder alguns deles ou localizar a sua posição em locais diferentes (LATTISI et al., 2022).

Dessa forma, a segmentação da petição em tópicos nomeados de Fato, Tese e Pedido (FTP) permite a redução do tamanho da petição ao produzir trechos apenas com as partes essenciais para a análise computacional de seu conteúdo. Esse processo facilita a identificação rápida e precisa dos elementos-chave do texto ao dividir a petição em seções distintas, agilizando a compreensão das informações contidas na petição. Essa abordagem também promove uma organização mais clara e eficiente do conteúdo do documento, sendo passível de uso em outros tipos de tarefas de PLN, como sumarizações ou buscas textuais.

1.3 OBJETIVOS

À luz do contexto apresentado, o presente trabalho tem como objetivo geral construir e disponibilizar um segmentador automático de texto com foco na segmentação de petições iniciais. Visando atingir este objetivo, os seguintes objetivos específicos foram delineados:

1. Realizar um levantamento de técnicas na literatura existente sobre segmentação textual;

2. Preparar um conjunto de dados específico de petições iniciais e segmentos jurídicos em tópicos FTP;

3. Implementar e avaliar modelos de redes neurais para segmentação textual;

4. Implementar e avaliar modelos de classificação de texto em FTP;

5. Comparar segmentações com grandes modelos de linguagem atuais;

6. Comparar classificações FTP com métodos de clássicos de classificação;

1.4 PRINCIPAIS CONTRIBUIÇÕES

Considerando os objetivos apresentados, foram atingidos com este projeto de mestrado os seguintes marcos:

1. Uma ferramenta para o auxílio aos profissionais jurídicos na leitura eficiente dos documentos, permitindo uma análise mais precisa e focada em segmentos relevantes;
2. Um método a ser incorporado no Firmina para o auxílio na classificação de precedentes no TJMA para o auxílio na celeridade visando à promoção da celeridade processual e aprimoramento na gestão de casos judiciais;
3. Um registro de programa de computador junto ao INPI, configurando produção técnica e salvaguardando o direito à propriedade intelectual do artefato de *software*.

1.5 ORGANIZAÇÃO DO DOCUMENTO

Este trabalho está estruturado em 6 capítulos. No Capítulo 2 são apresentados os conceitos fundamentais necessários para a compreensão da pesquisa. Inicialmente, aborda-se a estrutura e as características da petição inicial, seguida pelos principais fundamentos de mineração de texto e vetorização textual, incluindo técnicas como *Term Frequency-Inverse Document Frequency* (TF-IDF) e BERT. Em seguida, discutem-se os modelos de redes neurais relevantes para este estudo, com destaque para arquiteturas LSTM, *Bidirectional Long-Short Term Memory* (BiLSTM) e Redes de Ponteiros.

O Capítulo 3 revisa estudos e soluções existentes relacionados à segmentação e classificação de textos, especialmente no contexto jurídico, evidenciando lacunas que este trabalho busca preencher.

No Capítulo 4, descrevem-se a formalização do problema, os objetivos da segmentação e classificação de trechos, bem como a metodologia CRISP-DM utilizada. São detalhadas as etapas de entendimento do negócio, preparação e anotação dos dados, modelagem com redes neurais para identificação e classificação dos trechos, avaliação dos modelos e considerações sobre a implantação da solução desenvolvida.

O Capítulo 5 apresenta os experimentos realizados e analisa os resultados obtidos nas tarefas de identificação e classificação de trechos das petições. São comparadas diferentes arquiteturas de redes neurais, com a interpretação dos resultados à luz dos objetivos propostos.

Por fim, o Capítulo 6 consolida as conclusões do estudo, destacando as contribuições obtidas, limitações encontradas e possíveis direções para trabalhos futuros.

2 REFERENCIAL TEÓRICO

No âmbito deste capítulo, serão explorados os fundamentos essenciais para compreender o processo de segmentação textual, incluindo a caracterização da petição inicial, mineração de texto, os métodos de vetorização textual e as redes neurais aplicadas à classificação de dados.

2.1 A PETIÇÃO INICIAL NO PROCESSO JUDICIAL

A petição inicial constitui a peça por meio da qual o autor provoca a função jurisdicional do Estado, dando início à demanda judicial (MAGALHÃES; LUCAS, 2025). A partir dela, o conflito se transfere da esfera privada das partes e passa a ser formalmente submetido à análise institucional do Estado.

É por meio da petição inicial que o indivíduo lesado leva seu problema ao conhecimento do juiz, narrando os fatos, a lesão sofrida e os direitos que amparam sua pretensão de compensação e finaliza com o pedido para que o julgador aplique corretamente o direito ao caso concreto (OLIVEIRA, 2023). Um exemplo de estrutura comum de petição é descrito na Figura 1.

Em seu âmbito estrutural, os requisitos da petição inicial são definidos no artigo 282 do Código do Processo Civil (CPC), o qual estabelece elementos essenciais como a identificação das partes, a exposição dos fatos e fundamentos jurídicos do pedido, o valor da causa, as provas que serão apresentadas e o requerimento para citação do réu (OLIVEIRA; FERNANDES; KERSCHER, 2014).

Para além de seus requisitos formais, a petição inicial apresenta uma organização discursiva relativamente padronizada, pois além de apresentar elementos obrigatórios delimitados pelo CPC, também oferece espaço para personalização dos advogados para que expressarem seus raciocínios (LOURENÇO, 2008). Em linhas gerais, o texto é estruturado em regiões que cumprem funções específicas, como a descrição dos fatos, a fundamentação jurídica e a formulação do pedido. Embora a redação possa variar entre autores, essa divisão funcional tende a se repetir, o que permite observar padrões estruturais no documento (JÚNIOR; CALIXTO; CASTRO, 2020).

A identificação desses diretrizes estruturais é relevante não apenas sob o ponto de vista jurídico, mas também sob a perspectiva da análise textual. Uma vez que a petição inicial apresenta padrões recorrentes de organização, torna-se possível tratá-la como um documento estruturado, o que possibilita a aplicação de abordagens automatizadas para identificação e classificação de suas partes (QUEIROZ; BUENO; LISBINO, 2024).

Figura 1 – Exemplo Sintético de Petição

Exemplo de Petição Inicial Sintética

Autos nº: [Número do Processo] **Requerente:** QUALIFICAÇÃO DO REQUERENTE.

1. DOS FATOS

O Requerente, cliente do Banco [Nome do Banco] há [Número] anos, notou em seus extratos bancários mensais a cobrança recorrente e indevida de uma taxa denominada ...

Apesar de ter entrado em contato com o banco diversas vezes para questionar a cobrança e solicitar o estorno...

2. DO DIREITO

A cobrança duplicada da tarifa de manutenção de conta se configura como **prática abusiva** e **cobrança indevida**, violando o Código de Defesa do Consumidor (CDC), especificamente o artigo 39, inciso V, que proíbe o fornecedor de exigir do consumidor vantagem manifestamente excessiva.

A Lei nº ...

A jurisprudência majoritária dos tribunais brasileiros...

3. DO PEDIDO

Diante do exposto, o Requerente solicita a Vossa Excelência:

a) A **antecipação de tutela**, para que ...;

b) A **citação do Requerido** ...;

c) A **condenação do Requerido** a restituir, em dobro, o valor total ...;

d) O **pagamento das custas processuais** e dos honorários advocatícios.

Local e Data: [Cidade e Data]

Fonte: Autores (2026).

Nesse contexto, a compreensão da estrutura interna da petição inicial constitui etapa fundamental para a aplicação de técnicas de análise automatizada (MARINATO et al., 2022). A segmentação do texto em unidades funcionais, como fato, tese e pedido (FTP), demanda métodos capazes de reconhecer padrões linguísticos e estruturais em grandes volumes de dados. Assim, torna-se necessário recorrer a abordagens próprias da área de Mineração de Texto, que serão apresentadas na seção seguinte.

2.2 MINERAÇÃO DE TEXTO

A mineração de textos é um processo de descoberta de conhecimento que aplica técnicas de análise e extração de dados a partir de textos, frases ou até mesmo palavras individuais (REZENDE, 2003). Esse processo envolve o uso de algoritmos computacionais capazes de processar grandes volumes de texto e identificar informações úteis, muitas vezes

implícitas, que não poderiam ser obtidas por métodos tradicionais (MORAIS; AMBRÓSIO, 2007). Inspirada no conceito de *data mining* (mineração de dados), a mineração de textos tem como objetivo extrair conhecimentos relevantes de dados não estruturados ou semi-estruturados (BARION; LAGO, 2008).

A mineração de textos permite a transformação de grandes volumes de dados textuais não estruturados em conhecimento útil, muitas vezes inovador para as organizações (REZENDE; MARCACINI; MOURA, 2011). Isto posto, a automação desse processo se torna essencial haja vista o vasto volume de informações disponíveis, sendo viabilizada principalmente pelo uso de *softwares* e computadores (MORAIS; AMBRÓSIO, 2007). Nesse sentido, a organização inteligente dessas coleções textuais é fundamental para agilizar processos de busca e recuperação da informação em diversas instituições (REZENDE; MARCACINI; MOURA, 2011).

Essa organização envolve a aplicação de métodos computacionais para analisar e extrair informações úteis e implícitas de textos, frases ou palavras, que não seriam facilmente acessíveis por métodos tradicionais de consulta (GAUTAM et al., 2023). Em resumo, a mineração de textos é um processo de descoberta de conhecimento que examina grandes conjuntos de documentos para identificar novas informações ou responder questões específicas de pesquisa, convertendo essas informações em um formato estruturado para análise posterior (ABDUSALOMOVNA et al., 2023).

2.2.1 VETORIZAÇÃO TEXTUAL

A vetorização textual compreende o conjunto de métodos e ferramentas que são capazes de transformar dados textuais em dados numéricos. Esta seção abordará dois caminhos para a representação numérica de textos: a baseada puramente no *corpus* por meio de métodos estatísticos, o *Term Frequency–Inverse Document Frequency* (TF-IDF), e outra baseada e modelos de linguagem, o *Bidirectional Encoder Representations from Transformers* (BERT).

2.2.1.1 *TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY*

A técnica estatística TF-IDF é amplamente utilizada no processo de mineração de texto e tem como principal utilidade descobrir palavras de importância em um texto não estruturado ou semi estruturado (ABUBAKAR; UMAR; BAKALE, 2022). Atribui-se um valor a cada termo, que considera sua frequência no texto e em todos dos documentos da base, indicando sua importância (KIDO; JUNIOR; MORIGUCHI, 2014).

$$\mathrm{TF}(t,d) = \frac{count(t,d)}{|d|} \tag{2.1}$$

$$\text{IDF}(t, D) = \log\left(\frac{|D|}{count(t, D) + 1}\right) \tag{2.2}$$

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{2.3}$$

Onde:

t**:** termo (ou palavra) considerado.

d**:** documento específico do corpus.

D**:** conjunto de documentos.

$count(t, d)$**:** contabiliza o número de vezes que o termo t aparece em d.

$count(t, D)$**:** número de documentos em que o termo t ocorre.

Na Equação 2.3: a componente *term frequency* (TF), representada por $TF(t, d)$ descrita em 2.1, é o número de vezes que o termo t aparece no documento d. Já a componente da frequência inversa do documento (IDF), representada como $IDF(t, D)$ e descrita em 2.2, é o logaritmo do total de documentos D dividido pelo número de documentos que contém o termo t.

Dessa maneira, cada documento é representado por um vetor de tamanho m, definido durante a modelagem, onde cada dimensão corresponde a um termo único no conjunto de documentos e seu valor é o indicado pela equação 2.3.

2.2.1.2 *BIDIRECTIONAL ENCODER REPRESENTATIONS FROM TRANSFORMERS*

Bidirectional Encoder Representations from Transformers é um modelo de representação de linguagem o qual se destaca por sua aplicabilidade e desempenho em tarefas de processamento de linguagem natural (DEVLIN et al., 2018; ALAMMARY, 2022). O BERT baseia-se na arquitetura Transformer (VASWANI et al., 2017), que substituiu redes recorrentes pelo mecanismo de autoatenção, capaz de modelar relações de dependência entre palavras sem necessidade de processamento sequencial.

Dentre características, uma de suas principais abordagens para o uso dos modelos BERT é a extração de recursos (DEVLIN et al., 2018). Nessa extração de recursos ou características, a arquitetura do modelo BERT é preservada, ou seja, os parâmetros do modelo são "congelados", preservando o estado do modelo. Os recursos do texto são extraídos através do modelo BERT pré-treinado e, em seguida, incorporados a camadas de redes intermediárias, como redes recorrentes, ou acopladas aos modelos de classificação para resolver uma determinada tarefa (ALAMMARY, 2022).

O pré-treinamento do BERT é feito em duas tarefas centrais: *Masked Language Modeling* (MLM), em que palavras são ocultadas e o modelo deve prevê-las com base no contexto, e *Next Sentence Prediction* (NSP), que avalia a relação entre pares de sentenças (DEVLIN et al., 2018). Essas etapas permitem que o BERT aprenda representações semânticas antes de ser aplicado a tarefas específicas.

A criação de incorporações, ou vetorizações textuais, ocorre pela aplicação de modelos BERT já treinados e disponíveis publicamente, como no *Hugging Face* [1]. A versão base do modelo possui 110 milhões de parâmetros treináveis e é capaz de realizar representações textuais de maneira bidirecional, i.e, considera o contexto à direita e à esquerda de um *token* (DEVLIN et al., 2018). As representações geradas por esse modelo são utilizadas como representações vetoriais de alta qualidade para alimentar tarefas específicas, como classificação de texto, sumarização e tradução automática.

2.3 REDES DE CLASSIFICAÇÃO

As Redes Neurais Artificiais (RNAs) têm a capacidade de calcular, armazenar e distribuir as informações voltadas a uma determinada tarefa (HAYKIN, 2009). São sistemas compostos por unidades de processamento simples (neurônios artificiais) que calculam determinadas funções matemática normalmente não lineares (FALCÃO et al., 2013). A seguir serão descritas as principais redes neurais utilizadas para a segmentação textual.

2.3.1 REDES RECORRENTES

Uma Rede Neural Recorrente, do inglês *Recurrent Neural Networks* (RNN), opera de forma semelhante a outras redes neurais, com uma camada de entrada, camadas ocultas e uma camada de saída. Porém, sua principal característica são as conexões recorrentes dentro das camadas ocultas, que a tornam capaz de retornar a certas camadas dentro da rede (ASHBAUGH; ZHANG, 2024).

2.3.1.1 *LONG-SHORT TERM MEMORY*

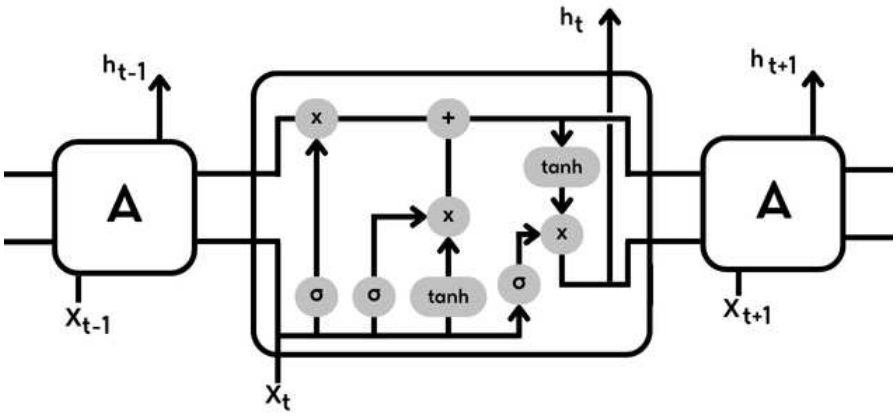
A LSTM é um tipo de rede neural recursiva no tempo e um tipo especial de RNN que tem a capacidade de reter e ponderar a relevância de informações por um longo tempo. Além disso, é adequado para processar e prever eventos importantes com intervalos relativamente longos e atrasos em séries temporais (YU et al., 2019a).

Redes LSTM diferem-se das redes densas ao possuírem a capacidade de usar informações contextuais ao realizarem um mapeamento computacional entre a sequência temporal de entrada e a saída (SHAHID; ZAMEER; MUNEEB, 2020). Sua arquitetura

[1] https://huggingface.co/docs/transformers/model_doc/bert

consiste em um conjunto de sub-redes conectadas recorrentemente, chamadas de blocos de memória. Cada bloco possui uma ou mais células de memória autoconectadas, além das conexões com os portões de entrada, saída e de esquecimento (GRAVES; GRAVES, 2012). A arquitetura de uma célula LSTM é descrita na Figura 2.

Figura 2 – Célula LSTM



Fonte: Adaptado de Yu et al. (2019a).

Onde:

t**:** passo temporal.

X_t**:** entrada no tempo t.

h_t**:** estado oculto no tempo t.

x**:** multiplicação elemento a elemento.

$+$**:** soma elemento a elemento.

σ**:** função sigmoid.

$tanh$**:** função tangente hiperbólica.

As portas da LSTM regulam o fluxo de informação na célula por meio de operações de multiplicação elemento a elemento. A porta de esquecimento, Fg_t, decide a fração do estado de memória anterior, C_{t-1}, que deve ser mantida. Quando o valor da porta do esquecimento, Fg_t, é 1, as informações são mantidas enquanto um valor 0 implica que as informações serão descartadas (YU et al., 2019b).

$$Fg_t = \text{sigmoid}\left(W_{fg}X_t + W_{hfg}h_{t-1} + b_{fg}\right) \tag{2.4}$$

A Equação 2.4 descreve o funcionamento do *forget gate* (Fg_t) em uma célula LSTM. Nesse contexto, X_t representa o vetor de entrada no instante de tempo t, h_{t-1}

corresponde ao estado oculto do instante anterior, W_{fg} e W_{hfg} são matrizes de pesos associadas, respectivamente, à entrada atual e ao estado oculto anterior, e b_{fg} é o vetor de viés. A função sigmoid$(\cdot)$ é utilizada para comprimir os valores no intervalo $(0, 1)$, permitindo que o portão controle o quanto da informação anterior deve ser mantida ou descartada. Para os demais portões, a mesma formulação é adotada, alterando-se apenas o subscrito que identifica cada um deles.

Em seguida, a porta de entrada Ig_t descrita pela equação 2.5 controla a quantidade de novas informações candidatas que serão incorporadas à célula enquanto a porta de saída Og_t explicitada em 2.6 decide quais informações podem ser geradas com base no estado da célula (YU et al., 2019b).

$$Ig_t = \text{sigmoid}\left(W_{ig}X_t + W_{hig}h_{t-1} + b_{ig}\right) \tag{2.5}$$

$$Og_t = \text{sigmoid}\left(W_{og}X_t + W_{hog}h_{t-1} + b_{og}\right) \tag{2.6}$$

Em outras palavras, a função dos portões de entrada e saída são de controle de fluxo da célula referente às entradas e saídas para o restante da rede, enquanto o portão de esquecimento limita ou permite a passagem da informação anterior para o próximo bloco, sendo presa ao bloco atual em caso de uma ativação e perpassada para as próximas células no caso contrário (SHAHID; ZAMEER; MUNEEB, 2020).

A Equação 2.7 demonstra como a célula LSTM integra as informações provenientes do estado de memória anterior, C_{t-1}, e da entrada atual, X_t, moduladas pelos portões de esquecimento e entrada. O portão de esquecimento, Fg_t, controla a fração do estado anterior que deve ser preservada, enquanto o portão de entrada, Ig_t, pondera a contribuição das novas informações candidatas geradas pela função de ativação tanh. Dessa forma, o estado da célula C_t funciona como uma memória de longo prazo, capaz de reter informações relevantes ao longo de várias iterações. Por fim, a saída da célula, dada pela Equação 2.6, é regulada pelo portão de saída Og_t, que decide quais partes da memória atual serão expostas à próxima camada ou ao próximo passo temporal, representando o estado oculto h_t. Essa separação entre o estado da célula e o estado oculto permite à LSTM equilibrar a retenção de informações históricas com a produção de saídas dinâmicas para cada instante de tempo.

$$(C)_t = (C)_{t-1} \otimes (Fg)_t + (Ig)_t \otimes \left(\tanh\left(W_C X_t + W_{hC} h_{t-1} + b_C\right)\right) \tag{2.7}$$

$$h_t = Og_t \otimes \tanh\left((C)_t\right) \tag{2.8}$$

Das equações 2.5 a 2.7, W_{fg}, W_{hfg}, W_{ig}, W_{hig}, W_{og}, W_{hog} e b_{fg}, b_{ig}, b_{og}, b_C representam os pesos e vieses (bias) dos 3 portões e da célula de memória, respectivamente. A entrada da rede é representada pela variável X_t e h_t retrata a saída da célula no instante t. Nessa formulação, $C(t-1)$ representa a memória de longo prazo, capaz de reter informações relevantes, enquanto h_t fornece a saída da célula para o próximo instante.

2.3.1.2 *BIDIRECTIONAL LONG-SHORT TERM MEMORY*

As redes LSTMs bidirecionais são uma extensão dos modelos LSTM, as quais duas LSTMs são aplicadas aos dados de entrada. Na primeira rodada, um LSTM é aplicado na sequência de entrada (ou seja, camada direta). Na segunda rodada, a forma reversa da sequência de entrada é alimentada no modelo LSTM (ou seja, camada reversa) (SIAMI-NAMINI; TAVAKOLI; NAMIN, 2019).

LSTMs convencionais capturam características das palavras, representadas pelos seus estados $H = (h_1, h_2, ..., h_n)$, enquanto as BiLSTMS concatenam a sequência H em ambas as direções. Portanto, a primeira LSTM processa uma sequência de palavras p_1 a p_n e captura seus estados h, denotada por $\overrightarrow{h}$, enquanto a segunda LSTM processa a sequência p_n a p_1, produzindo uma sequência de estados h de maneira reversa, denotada por $\overleftarrow{h}$. Portanto, o estado h das BiLSTMs é calculado da seguinte forma:

$$h_i = \overrightarrow{h_i} \oplus \overleftarrow{h_i}, h_i \in R^{2L} \tag{2.9}$$

Onde o operador $\oplus$ indica a função de concatenação e L o comprimento da sequência de estados produzida por uma LSTM convencional.

2.3.2 REDES DE PONTEIROS

Modelos de redes neurais do tipo sequência-para-sequência (*seq2seq*), compostos por uma RNN como codificador e outra como decodificador, normalmente assumem um vocabulário de saída fixo de tamanho n, definido durante a modelagem da rede. Esse cenário limita a atuação do modelo quando o tamanho da saída depende da própria entrada, como em problemas em que o número de elementos a prever varia com o comprimento da sequência de entrada (VINYALS; FORTUNATO; JAITLY, 2015).

Essa limitação decorre do fato de que o decodificador produz saídas a partir de um vocabulário pré-definido. Em tarefas como o Problema do Caixeiro Viajante, em que a saída é uma sequência de índices da própria entrada, o tamanho desse "vocabulário" depende diretamente do número de elementos de entrada, impossibilitando uma solução única para diferentes comprimentos de sequência (ZHONG et al., 2023).

Esse problema é tratado com uma RNN adicional que usa um mecanismo de atenção sobre toda a sequência de estados da RNN codificadora. Em modelos *seq2seq* em

que as RNN são, por exemplo, LSTMs, o mecanismo de atenção atua nos estados ocultos do codificador e do decodificador, logo após a multiplicação aplicada por meio do portão de saída. O vetor de atenção em cada instante i é calculado da seguinte forma:

$$u_j^i = v^T \tanh\left(W_1 e_j + W_2 d_i\right) \quad j \in (1, \ldots, n) \tag{2.10}$$

$$a_j^i = \text{softmax}\left(u_j^i\right) \quad j \in (1, \ldots, n) \tag{2.11}$$

$$d_i' = \sum_{j=1}^{n} a_j^i e_j \tag{2.12}$$

Onde $(e_1, ..., e_n)$ e $(d_1, ..., d_m(P))$, são os estados ocultos do codificador e decodificador, respectivamente; O vetor u_j^i representa a máscara de atenção sobre as entradas, o qual será normalizada pela função softmax e v, W_1 e W_2 são parâmetros treináveis; $d\prime_i$ e d_i são concatenados e usados como estados ocultos, utilizados para previsões ou alimentação das etapas ou redes subsequentes.

O vetor $d\prime_i$, portanto, resulta da atenção aplicada sobre a entrada e carrega a informação contextual relevante naquele passo. A concatenação com o estado do decodificador d_i permite que a rede preserve tanto a memória do histórico da saída quanto a atenção sobre a entrada, enriquecendo a representação usada para a próxima predição.

Redes com o mecanismo de atenção, por si só, não são capazes de lidar com o problema do vocabulário de saída variável. As redes de ponteiros (*Pointer Nets*) partem da redução do mecanismo de atenção, removendo a combinação do estado do codificador e_j que propagava informações extras para o decodificador.

$$\begin{aligned} u_j^i &= v^T \tanh\left(W_1 e_j + W_2 d_i\right) \quad j \in (1, \ldots, n) \\ p\left(C_i \mid C_1, \ldots, C_{i-1}, \mathcal{P}\right) &= \text{softmax}\left(u^i\right) \end{aligned} \tag{2.13}$$

A principal modificação é o uso do u_j^i como um vetor de ponteiros que pondera a relevância de cada elemento da entrada, produzindo uma sequência de probabilidades, denotada por $p\left(C_i \mid C_1, \ldots, C_{i-1}, \mathcal{P}\right)$, que representa a função a ser aproximada no treinamento da rede (VINYALS; FORTUNATO; JAITLY, 2015). Aqui, $\mathcal{P}$ corresponde ao conjunto de elementos da entrada. A distribuição resultante não está sobre palavras de um vocabulário fixo, mas sobre os índices dos próprios elementos de entrada, o que permite que a saída acompanhe naturalmente variações no tamanho da sequência de entrada.

Em resumo, enquanto o *seq2seq* clássico gera símbolos de um vocabulário fixo e o mecanismo de atenção enriquece o contexto para essa geração, as Pointer Nets utilizam diretamente os escores de atenção para apontar para elementos da entrada,

tornando-se adequadas para problemas com saídas de tamanho variável. Dessa forma, as Pointer Nets representam uma extensão natural dos modelos *seq2seq* com atenção ao reinterpretar os escores de atenção como uma distribuição diretamente sobre os elementos da entrada. Essa reformulação elimina a dependência de um vocabulário fixo e torna o modelo estruturalmente adequado a tarefas em que a saída consiste em selecionar ou ordenar elementos do próprio conjunto de entrada.

3 TRABALHOS CORRELATOS

O trabalho realizado por Li et al. (2020) destaca um modelo de atenção hierárquica para análise de sentimento por meio da segmentação textual a nível de frase. Os autores apresentam uma arquitetura de rede capaz de lidar com a problemática da dimensão de saída fixa dos modelos de inteligência artificial, nomeada de SegBot, permitindo a variabilidade do vocabulário de entrada. Os autores propõem um sistema fundamentado em representação distribuída, capaz de aprender recursos informativos de maneira autônoma para cada tarefa e domínio, eliminando a necessidade de intervenção humana. Para tal, os autores adotam a estratégia de previsão de sequência, empregando modelos codificador-decodificador, utilizando uma rede neural recorrente bidirecional para a codificação da sequência de entrada e outra RNN como modelo de linguagem para gerar a sequência de saída. A limitação do vocabulário fixo é superada pela incorporação de um mecanismo de ponteiros que permite inferir os limites dos segmentos. Os resultados experimentais, obtidos a partir de *benchmarks* como *Movie Review* e *Stanford Sentiment Treebank*, demonstram que o SegBot performa bem nas tarefas de segmentação em tópicos e em unidades de discurso. Embora eficaz na segmentação de tópicos curtos (mudanças entre assuntos vizinhos), o SegBot não demonstra capacidade de rastrear tópicos distribuídos em longos intervalos textuais, uma limitação relevante para petições jurídicas, onde fatos e argumentos frequentemente se estendem por múltiplas páginas com intercalação de subtópicos.

Lukasik et al. (2020) propõe três arquiteturas baseadas em *transformers*: BERT de segmento cruzado, BERT+BiLSTM e BERT-Hierárquico para as tarefas de segmentação textual. Todas utilizando atenção entre contextos locais para identificar mudanças temáticas. O modelo BERT de segmento cruzado destaca-se por alimentar o modelo com representações de quebras candidatas, considerando seus contextos locais esquerdo e direito. Já o Bert+BiLSTM codifica as frases independentemente com BERTLARGE e, posteriormente, usa um BiLSTM para capturar a representação da sequência de sentenças. O Bert-Hierárquico, por sua vez, substitui o BiLSTM por um codificador baseado em *transformers*. Os experimentos de segmentação foram realizados nos conjuntos Wiki-727K, Choi e RST-DT, utilizando métricas como Precision, Recall e F1-score. Entretanto, uma limitação relevante é a avaliação restrita a textos curtos e não pertencentes ao domínio jurídico, o que deixa em aberto sua eficácia em documentos longos e semanticamente densos, como as petições iniciais.

O estudo conduzido por Júnior, Calixto e Castro (2020) explora a segmentação de documentos para melhorar a classificação de petições iniciais, focando na divisão do texto em três partes FTP. A segmentação é realizada por meio de duas abordagens: o uso de um *parser* e a aplicação de redes neurais. O *parser*, um *software* especializado, segmenta o

texto identificando palavras de referência, enquanto a rede neural, um *perceptron* com duas camadas treinadas de forma supervisionada, utiliza a função de ativação arco tangente hiperbólico. Essa abordagem classifica os textos em quatro intervalos, correspondendo ao FTP ou a um rótulo de impossibilidade de detecção do FTP. Embora a segmentação não seja o foco central do estudo, os autores destacam sua importância na identificação de documentos semelhantes em diferentes comarcas, sugerindo que a segmentação textual pode melhorar significativamente a classificação e agrupamento de petições. Entretanto, três limitações são evidentes: (1) o parser depende fortemente de palavras-chave pré-definidas, tornando-o frágil perante a variabilidade linguística; (2) a rede neural possui capacidade de capturar nuances contextuais limitada; e (3) a segmentação serve apenas como etapa preliminar para classificação, sem avaliação independente de sua acurácia. O presente trabalho supera essas restrições ao propor uma arquitetura profunda especializada em segmentação semântica, utilizando *transformers* para modelar relações contextuais de longo alcance e validando diretamente a qualidade da divisão FTP em petições anotadas.

No estudo de Sheik, Ganta e Nirmala (2024) 5 modelos foram propostos, combinando redes convolucionais, LSTMs, BiLSTMS, *Gated Recurrent Unit* (GRU), *Bidicrectional Gated Recurrent Unit* (BiGRU), *Conditional Random Fields* (CRF), mecanismos de atenção e modelos baseados em transformadores como CaseLawBERT e LegalBert. A abordagem proposta pelos autores utiliza o contexto em torno do delimitador de início e fim de segmento para detectar limites de sentenças, tratando-se de um problema de classificação de sequência. Os modelos foram treinados no conjunto de dados de Savelka et al. (2017) e avaliados no conjunto de dados de Malik et al. (2021), ambos de decisões judiciais. Os modelos baseados em CNNs obtiveram melhores desempenhos com base no tamanho do modelo, pontuação F1 e tempo de inferência. No entanto, o trabalho limita-se à tarefa de detecção de limites de segmentos em nível de sentenças, sem explorar a segmentação em unidades semânticas maiores ou a estrutura hierárquica de documentos jurídicos completos.

Em Vinotheni e Lakshmanapandian (2021), os pesquisadores fazem uso de *Fast Recurrent Neural Network*s (FRNN) com BiLSTM para a segmentação textual em frases. A arquitetura FRNN proposta pelos autores faz uso de codificações *One-Hot* nos dados textuais para atuar como nível de incorporação para a rede composta por CNN, remodelagem matricial, dois blocos BiLSTM e uma camada densa. O treinamento da rede utilizou os conjuntos de dados WIKI-727, *Choi* e *Wikipedia*. Para medir o desempenho do modelo, foram empregadas métricas fundamentais, incluindo *F1-Score*, revocação e acurácia, que fornecem uma avaliação abrangente das capacidades de segmentação textual do protótipo proposto. Embora o modelo demonstre eficácia em conjuntos genéricos, a abordagem não é validada em documentos jurídicos e se restringe a segmentações a nível de frase. Ademais, os autores usam codificações *One-Hot* para a representação textual, as quais podem não capturar as relações textuais em sua totalidade.

Tabela 1 – Trabalhos correlatos sobre segmentação de texto.

Referência	**Método Aplicado**	**Aprendizado**	**Base de Dados**
Lukasik et al. (2020)	BERT + BiLSTM	Supervisionado	Wiki-727K; Choi; RST-DT
Júnior, Calixto e Castro (2020)	Parser + MLP	Supervisionado	Petições iniciais do TJGO
Sheik, Ganta e Nirmala (2024)	CNNs, LSTMs, mecanismos de atenção, CRFs e BERT	Supervisionado	Decisões judiciais
Li et al. (2020)	BiGRU + BiLSTM + PTR-Net	Semi-supervisionado	Choi; RST-DT
Vinotheni e Lakshmanapandian (2021)	CNN + BiLSTM + MLP	Supervisionado	Wiki-727; Choi; Wikipedia

Fonte: Autores (2026).

A Tabela 1 resume trabalhos correlatos relacionados à segmentação de texto, destacando diferentes abordagens de métodos aplicados, tipos de aprendizado utilizados na segmentação e bases de dados empregadas. A partir da análise apresentada na Tabela 1, observa-se que, embora existam avanços significativos no emprego de arquiteturas profundas para segmentação textual, a maior parte dos estudos concentra-se em textos curtos, domínios genéricos ou em tarefas restritas à identificação de limites em nível de sentença. Ademais, poucos trabalhos avaliam de forma independente a qualidade da segmentação em documentos jurídicos extensos. Nesse contexto, evidencia-se uma lacuna quanto à unidades semânticas mais amplas e à consideração da organização hierárquica típica do discurso jurídico. O presente trabalho posiciona-se justamente nesse espaço, ao propor uma abordagem voltada à segmentação semântica de petições iniciais, explorando modelos baseados em *transformers* capazes de capturar dependências de longo alcance e avaliando a qualidade dos segmentos propostos.

4 METODOLOGIA

Este Capítulo compreende o desenvolvimento do modelo para a segmentação de petições, guiado pelas pesquisas realizadas e pela avaliação dos correlatos da seção anterior.

4.1 DEFINIÇÃO DO PROBLEMA

Nesta Seção é apresentada a formalização dos processos envolvidos na segmentação e seleção de trechos relevantes em documentos textuais. Para tal, é introduzida as características da segmentação textual, com a variabilidade no número de palavras por documento e a identificação de trechos relevantes, estabelecendo uma metodologia para lidar com a entrada e saída de dados.

A formalização é dividida em três partes principais: segmentação, seleção de segmentos e classificação. A primeira define como dividir os documentos em blocos de tamanho fixo, seguida da identificação e representação trechos relevantes, finalizando com a classificação desses trechos.

4.1.1 SEGMENTAÇÃO

O processo de segmentação textual possui algumas peculiaridades, como a variabilidade no tamanho da entrada e saída. Modelos de segmentação textual recebem como entrada textos que possuem tamanhos variados, ou seja, um documento D_1 pode conter n_1 palavras e outro D_2 pode conter n_2 palavras. Essa característica se repete ao obter uma lista de segmentos relevantes por documento e no tamanho desses segmentos cada documento contém sua própria estrutura de coesão e transição topical específica que resulta em um conjunto de segmentos relevantes distintos de documento a documento.

Considerando essas características, define-se documento D_n como uma sequência de palavras $D_n = (p_1, p_2, ..., p_n)$ e a identificação de um trecho relevante como uma função binária $f(p_i)$ que indica o pertencimento (1) ou não pertencimento (0) de p_i a um trecho relevante. Dessa maneira os trechos FTP anotados são transformados em uma única sequência de zeros e uns $q = (q_1, q_2, ..., q_n)$ mapeando os elementos p e, portanto, possuindo o mesmo tamanho de p. Ao eliminar a classificação FTP dos segmentos elimina-se também o caráter variável do número de segmentos das quantidades de variável de cada classe para apenas a identificação de trechos com potencial de ser FTP.

De maneira similar, para tratar o tamanho variável dos documentos, representa-se um documento D_n, definido previamente por uma sequência de palavras p, como um conjunto de sequências de tamanhos b de tal maneira que o número de blocos em um

documento seja dado por:

$$f(D_n, b) = D_n/b. \tag{4.1}$$

Dessa forma, cada documento agora é representado por uma sequência de blocos B tamanho fixo, tornando a modelagem dos documentos em uma função de b. Essa abordagem implica que os blocos finais podem conter um número de tokens menor que b, para essas situações um processo de padding é aplicado para assegurar a uniformidade dos blocos. Por fim, as transformações de divisões de blocos aplicados em p também devem ser aplicadas em q para garantir que a informação das posições p_i e q_i sejam mantidas.

Convém destacar que a divisão do texto em blocos pode ser realizada antes ou depois da tokenização dos documentos. Aplicar a divisão antes da tokenização permite o uso de artifícios como divisão do texto em parágrafos/períodos para comprimentos menores que uma certa quantidade de palavras, dando ênfase a um contexto a nível de parágrafo. Já a divisão após o processo de tokenização implica em uma modelagem que consiga capturar o contexto entre um bloco B_i e B_{i-1}.

4.1.2 SELEÇÃO DE SEGMENTOS

Uma vez definido o tamanho dos blocos, o processo de seleção de trechos relevantes, i.e, trechos que possivelmente se encaixam como FTP, o procedimento identificação desses trechos torna-se uma tarefa de mapear cada representação vetorial de uma petição de formato $(f(D_n, b), b)$ em um novo vetor binário de igual formato. O vetor binário resultante tem a finalidade de indicar quais são os tokens que fazem parte de uma sequência marcada como potencial FTP. Portanto, a seleção desses trechos pode ser entendida como uma transformação $T(v_{(f(D_n,b),b)})$.

$$T(v_{(f(D_n,b),b)}) \rightarrow v^{bin}_{(f(D_n,b),b)} \mid v^{bin}_{(f(D_n,b),b)} \in \{0, 1\} \tag{4.2}$$

Dessa forma, a sequência de blocos fixos gerada a partir do documento original é submetida a uma função de mapeamento que, por meio de vetores binários, determina uma sequência de palavras como um potencial trecho FTP. O modelo de classificação voltado para a seleção de segmentos, portanto, será uma aproximação de $T(v_{(f(D_n,b),b)})$.

4.1.3 CLASSIFICAÇÃO

Identificado os trechos potenciais FTP, a atribuição de seu respectivo rótulo é obtida por meio da aplicação de um modelo de classificação supervisionada para a aproximação de uma função ideal de classificação textual $f(bloco)$, de maneira que:

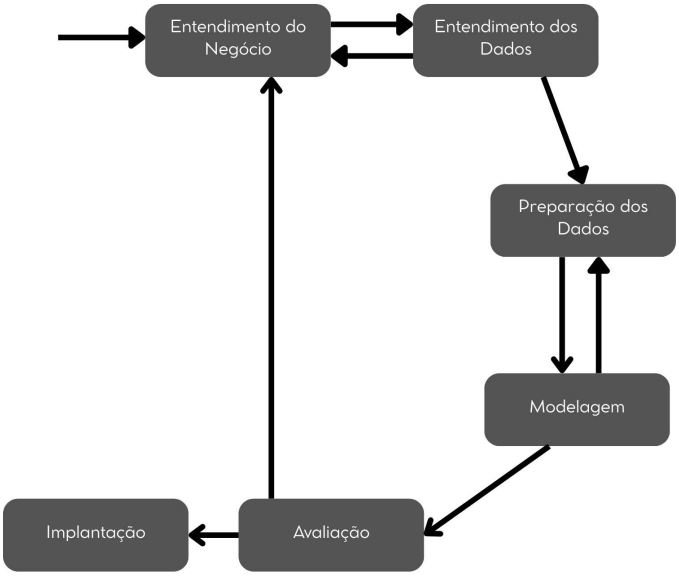
$$f(\text{bloco}) \rightarrow \{\text{fato}, \text{tese}, \text{pedido}\} \tag{4.3}$$

A classificação atuará sobre as sequências obtidas de $v^{bin}_{(f(D_n,b),b)}$ e irá mapear cada sequência binária para um rótulo específico, conforme a equação 4.3.

4.2 *CROSS-INDUSTRY STANDARD PROCESS FOR DATA MINING*

Este estudo segue o método de pesquisa conhecido como (CRISP-DM) (WIRTH; HIPP, 2000). A aplicação desse modelo compreende seis etapas: Entendimento de Negócios, Entendimento dos Dados, Preparação dos Dados, Modelagem, Avaliação e Implementação. Assim como consta a Figura 3.

Figura 3 – Fases do método CRISP-DM.



Fonte: Adaptado de Shearer (2000).

A metodologia possui um fluxo flexível na passagem de uma fase a outra. Na construção de projetos é bastante comum o avanço e retorno entre as etapas quando necessário, dando um caráter cíclico e iterativo no desenvolvimento (RAMOS et al., 2020). As duas primeiras etapas referem-se à contextualização do projeto e conhecimento dos dados obtidos, caracterizando estas etapas como um processo analítico e de exploração. As duas etapas seguintes se relacionam na medida em que os dados são tratados para se adequarem à solução proposta. Durante a fase de modelagem do CRISP-DM, algoritmos de aprendizado de máquina são frequentemente usados para extrair conhecimento valioso dos dados (RIBEIRO et al., 2020). As duas fases subsequentes tratam da avaliação da solução proposta e de sua disponibilização para uso. Em caso de avaliações inconsistentes, o modelo permite a reconsideração de novos fatores e deslocamento para a etapa inicial da metodologia.

4.2.1 ENTENDIMENTO DO NEGÓCIO

No contexto do sistema jurídico brasileiro, as petições iniciais ocupam uma posição central, uma vez que representam o ponto de partida formal das ações judiciais. Com o avanço da digitalização do Judiciário, tem-se observado um crescimento expressivo no volume de documentos processuais digitais, o que impõe desafios crescentes aos sistemas de automação e análise jurídica (GAYEN et al., 2024). Dentre esses desafios, destaca-se a dificuldade de lidar com a estrutura interna das petições iniciais, que nem sempre seguem um padrão formal bem definido e, frequentemente, apresentam uma combinação de linguagem técnica-jurídica e narrativa livre (POLO et al., 2021).

Atualmente, a maioria dos sistemas jurídicos automatizados atua com base em documentos em sua totalidade, sem segmentação explícita em partes que reflitam a lógica argumentativa jurídica (AUMILLER et al., 2021). Isso limita a capacidade desses sistemas de realizar tarefas mais especializadas, como a classificação do tipo de ação ou a extração de informações relevantes. Nesse cenário, torna-se evidente a necessidade de métodos capazes de estruturar automaticamente esses documentos, com o objetivo de melhorar a qualidade e a interpretabilidade dos dados utilizados por sistemas jurídicos inteligentes.

O presente trabalho propõe a construção de um modelo baseado em redes neurais para a segmentação de petições iniciais segundo as três categorias FTP. A escolha dessas categorias fundamenta-se na observação de que elas correspondem a núcleos argumentativos centrais em petições iniciais e que, se identificadas e separadas de forma automática, podem servir como insumo valioso para modelos de classificação ou triagem processual.

Embora existam iniciativas voltadas à análise de textos jurídicos, são ainda escassas as abordagens robustas que tratam diretamente da segmentação semântica de petições com foco nessas categorias, especialmente para o cenário brasileiro. Estudos anteriores indicam que a segmentação estrutural pode contribuir como etapa de pré-processamento para sistemas de NLP jurídico (JÚNIOR; CALIXTO; CASTRO, 2020). Nesse contexto, a hipótese central deste trabalho é que a identificação automática de segmentos FTP pode atuar como uma etapa de pré-processamento promissora, melhorando o desempenho de outros sistemas de automação ao conferir uma estrutura semântica mais clara aos documentos analisados.

A implementação dessa solução tem o potencial de melhorar a estruturação das petições, facilitar sua compreensão e otimizar o trabalho de advogados, juízes e demais profissionais do direito. Além disso, a segmentação automática das petições pode ser utilizada como um recurso auxiliar em outras aplicações de PLN, como sumarização automática e sistemas de busca textual, ampliando o impacto da pesquisa. Espera-se que a automatização desse processo contribua significativamente para a eficiência do sistema jurídico, reduzindo o tempo de análise e aumentando a acessibilidade da informação

processual.

4.2.2 ENTENDIMENTO DOS DADOS

Os dados utilizados foram fornecidos pelo TJMA e consistem em um conjunto de 9.497 arquivos binários referentes a 1030 processos judiciais sob o contexto de avaliar métodos para a identificação automática de petições presentes nesses arquivos. Inicialmente, sabia-se que esses documentos eram anexos extraídos do sistema Processo Judicial Eletrônico (PJe), incluindo as petições iniciais. No entanto, uma análise preliminar revelou que os arquivos binários continham uma variedade de formatos, abrangendo documentos de texto, vídeos e imagens.

A primeira etapa da exploração dos dados envolveu a identificação dos formatos de arquivos. Para isso, foram utilizadas técnicas de análise de metadados e inspeção do conteúdo binário, selecionando apenas arquivos cuja assinatura dos primeiros *bytes* fosse equivalente à definição de assinaturas pdf pela ISO 8859-1, a saber, os *bytes* são 25 50 44 46 2D ou 25 50 44 46. Essa filtragem reduziu o número de arquivos para 7.442.

Uma vez que os arquivos foram convertidos, uma filtragem manual foi conduzida para a identificação das petições iniciais. Essa filtragem foi realizada por meio de uma ferramenta web desenvolvida pelo autor para este fim. A ferramenta auxiliava na marcação manual dos arquivos ao exibir seu conteúdo e permitir sua marcação como petição, exportando um arquivo *Comma Separated Values* com o resultado das marcações. O critério de seleção definido foi a presença da tripla FTP nos documentos. Os resultados dessa filtragem estão sumarizados na Tabela 2.

Tabela 2 – Distribuição dos Tipos de Documentos

Tipo	**Quantidade**
Petição	675
Petição não encontrada	118
Reclamação	72
Relações de consumo	62
Termo de reclamação	48
Apenas procuração	31
Formulário de reclamação	10
Atermação “on-line”	9
Termo judiciário	1
Apenas boletim de ocorrência	1
Boletim de ocorrência e procuração	1
Termo de notificação	1
Apenas procedimento do Juizado Especial Cível	1
Total de Documentos	1.030

Fonte: Autores (2026).

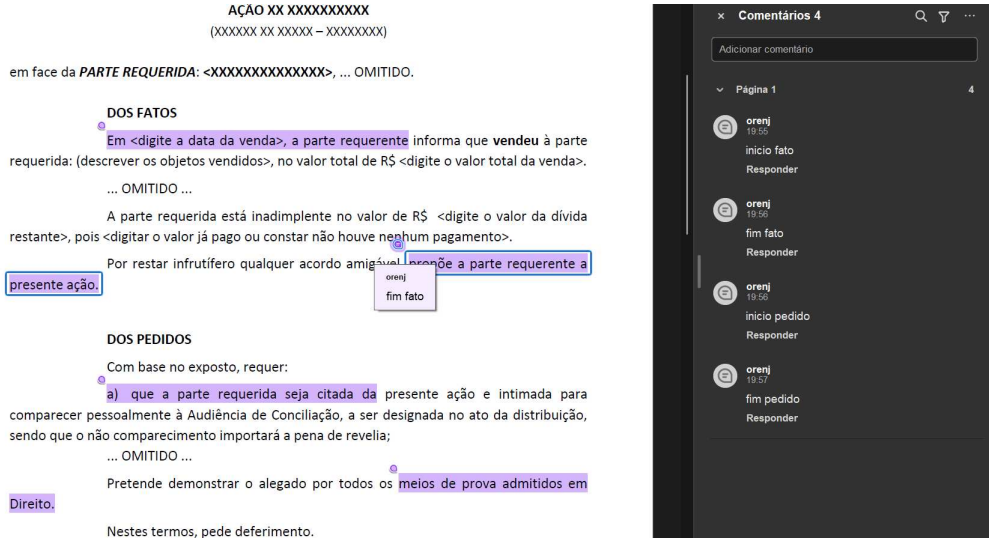
Conforme é possível observar na Tabela 2, 675 petições foram encontradas. Objetivando otimizar as etapas subsequentes, as petições digitalizadas foram desconsideradas, resultando em 619 petições.

4.2.3 PREPARAÇÃO DOS DADOS

Com as petições já identificadas, inicia-se um processo de anotação de dados com o objetivo de identificar a tripla FTP nos documentos. Para tal, utilizou-se a ferramenta *Acrobat Reader*[1] da *Adobe* para realizar a marcação dos trechos por meio de comentários no documento.

Foram definidos 6 classes de marcação, correspondendo ao início e fim de cada elemento da tripla FTP. Dessa maneira, a anotação não impunha o esforço de marcar um trecho por completo, bastando apenas marcar as 5 primeiras palavras do início e fim do trecho. A Figura 4 ilustra uma anotação de exemplo que está em conformidade com as diretivas da anotação.

Figura 4 – Exemplo de Anotação



Fonte: Autores (2026).

O processo de anotação dos dados foi conduzido com o envolvimento de pesquisadores do Laboratório de Inteligência Computacional (LincProg), que contribuíram com sua experiência na área de processamento de linguagem natural, e contou com a participação de um profissional especializado em jurimetria e anotação de dados. Para garantir um conjunto de dados representativo e viável para a modelagem, foi realizado inicialmente um cálculo de tamanho amostral considerando uma população de 675 documentos, nível de confiança de 85% e margem de erro de 10%, conforme descrito por Bussab e Morettin (2010). O cálculo resultou em uma amostra mínima próxima de 85 documentos. Com o

[1] https://get.adobe.com/br/reader/

objetivo de ampliar ligeiramente a representatividade e manter uma margem de segurança em relação ao valor estimado, optou-se pela anotação de 100 documentos sorteados aleatoriamente, número escolhido para equilibrar a diversidade textual com a necessidade de um processo de anotação criterioso e consistente.

Pós anotação, é necessário recuperar o conteúdo textual entre as *tags* de início e fim de cada tópico. Esse procedimento é realizado com a biblioteca *PyMuPDF* (ARTIFEX, 2025), a qual permite a identificação dos conteúdos dos comentários e o conteúdo efetivamente marcado através da posição espacial dos vértices dos retângulos de marcação observados na Figura 4. A identificação do trecho em sua forma completa ocorre ao localizar o conteúdo marcado e recuperar a posição no texto em que ele se encontra. Ao realizar a localização de cada par início-fim, o conteúdo em sua totalidade pode ser obtido ao selecionar o intervalo do índice da primeira palavra do conteúdo da *tag* de início e última palavra do conteúdo da *tag* de fim.

A qualidade da anotação foi avaliada com base na concordância média expressa pelo alfa de Krippendorff entre três anotadores sobre um conjunto de dados, apresentando um valor médio de $0,809$ no conjunto, valor este dentro do nível de boa concordância definido pela literatura. Para a construção do dataset, considerou-se a interseção entre as marcações feitas pelos anotadores. A métrica de Krippendorff é descrita em mais detalhes na seção de avaliação 4.2.5.

A etapa subsequente compreende a validação dos trechos recuperados. Em determinadas situações, modificações no tamanho da fonte, ocasionadas pela reconstrução do arquivo binário ou por alterações na formatação, poderiam modificar a ordem das palavras dentro de uma anotação, inviabilizando sua recuperação posterior. Nesses casos, a solução adotada foi a revisão manual da marcação, ajustando a quantidade de palavras selecionadas conforme necessário.

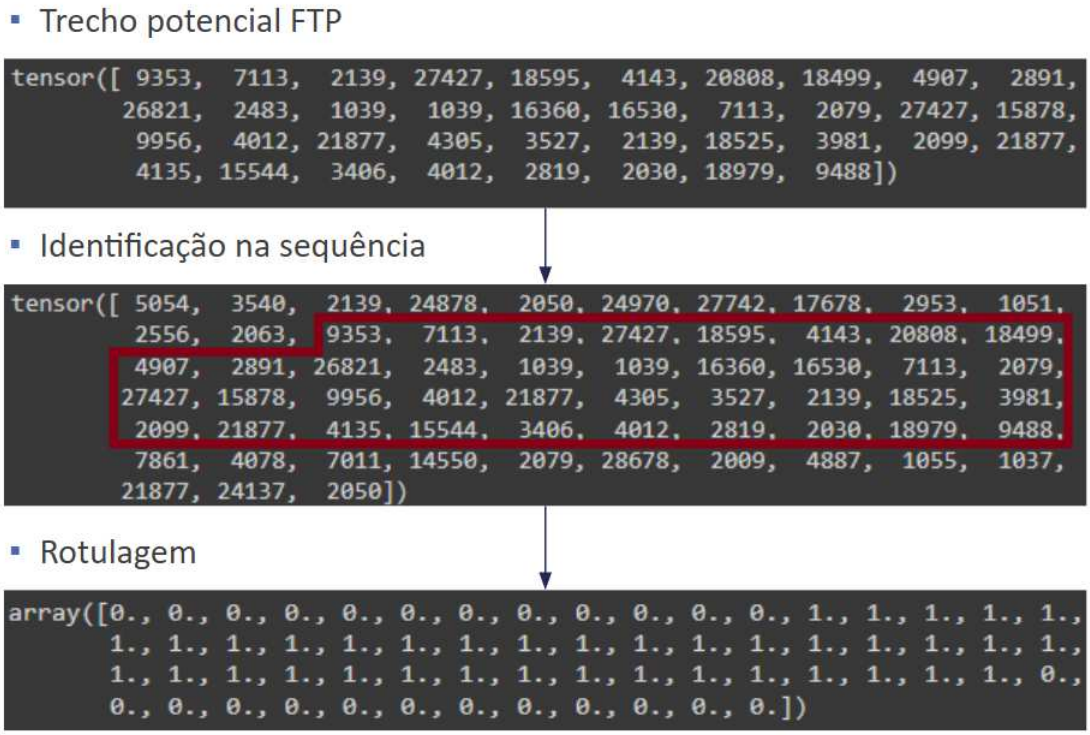
4.2.3.1 IDENTIFICAÇÃO DE TRECHOS RELEVANTES

Com os trechos validados, o conjunto de dados foi estruturado de forma a associar cada petição à sua respectiva lista de marcações. Para cada petição e anotação recuperada, é aplicado um *pipeline* de pré-processamento textual. O objetivo foi garantir que o texto fosse tratado de forma consistente e de fácil manipulação em etapas posteriores, minimizando as divergências em relação ao conteúdo original do PDF. O pré-processamento inclui a remoção de quebras de linha desnecessárias, conversão de espaços em branco e normalização de caracteres, além de garantir a uniformização do texto em minúsculas.

Em seguida, foi utilizado o modelo BERT (DEVLIN et al., 2018) em sua versão *bert-base-uncased* para realizar a tokenização dos textos e das respectivas anotações. Com base nessa tokenização, foi gerado um vetor binário para mapear cada petição tokenizada a um vetor de mesmo tamanho. Esse vetor apresenta o valor 1 para os tokens pertencentes

a uma anotação e 0 nos demais casos, sendo denominado *máscara de anotação*. A Figura 5 ilustra esse procedimento para um trecho de exemplo.

Figura 5 – Mapeamento dos trechos FTP para vetores binários



Fonte: Autores (2026).

A Figura 5 exemplifica um trecho de uma petição já tokenizada, no qual um possível segmento FTP se inicia no *token* de id 9.353 (posição 13) e se estende até o *token* de id 9.488 (posição 50). Dessa forma, o vetor binário resultante apresenta valor 1 no intervalo entre as posições 13 e 50, e valor 0 nas demais posições.

Por fim, as petições tokenizadas e suas respectivas máscaras de anotação foram segmentadas em blocos de 512 tokens. Dessa forma, cada petição foi transformada em um conjunto de segmentos de 512 tokens, onde os primeiros 256 tokens são utilizados para carregar o contexto do bloco anterior (com exceção do primeiro bloco) e 256 subsequentes para o conteúdo do bloco atual. Para os blocos finais com menos de 256 novos tokens, tokens *padding* (PAD) do BERT foram aplicados para o preenchimento dos dados, garantindo a compatibilidade com o modelo de aprendizado de máquina que será empregado nas etapas subsequentes.

4.2.3.2 CLASSIFICAÇÃO DOS TRECHOS

Após a segmentação das petições e extração dos trechos anotados, monta-se uma base de dados contendo apenas o texto das anotações e seu respectivo rótulo FTP. A

preparação dos dados, portanto, compreende a normalização textual, vetorização dos dados e codificação das classes.

A normalização textual inicia-se com a conversão de todos os caracteres para minúsculas, seguido da remoção de caracteres numéricos e sinais de pontuação. Adicionalmente, realiza-se a eliminação de espaços excedentes e a remoção de acentos gráficos, garantindo a padronização ortográfica do texto. Posteriormente, para mitigar a interferência de termos com baixa carga informacional na vetorização TF-IDF, são eliminadas as palavras de parada (*stopwords*) da língua portuguesa. Além disso, aplica-se a lematização, permitindo a redução das palavras à sua forma canônica, de modo a minimizar variações morfológicas e a garantir maior generalização do modelo.

Finalizado o pré-processamento textual, os trechos são representados numericamente por meio da técnica TF-IDF. Esse método atribui pesos às palavras com base na sua frequência relativa no corpus (KIDO; JUNIOR; MORIGUCHI, 2014). Para controle da dimensionalidade e preservação dos aspectos mais relevantes do texto, fixou-se um limite de 5.000 características (*features*). Como resultado, obtém-se um vetor numérico cujas posições correspondem a um termo ponderado conforme sua importância relativa.

Além da vetorização textual, as classes da tripla FTP foram codificadas para viabilizar sua utilização em modelos de aprendizado de máquina. Para tal, utilizou-se a ferramenta *Label Encoder* do módulo Scikit Learn (PEDREGOSA et al., 2011) para converter as classes categóricas em valores numéricos discretos. Para compatibilizar a representação das classes com modelos baseados em redes neurais, realizou-se a conversão dos rótulos para um formato *one-hot encoded*, resultando em uma matriz binária onde cada instância do conjunto de dados possui um vetor de três dimensões, indicando sua respectiva classe.

4.2.4 MODELAGEM

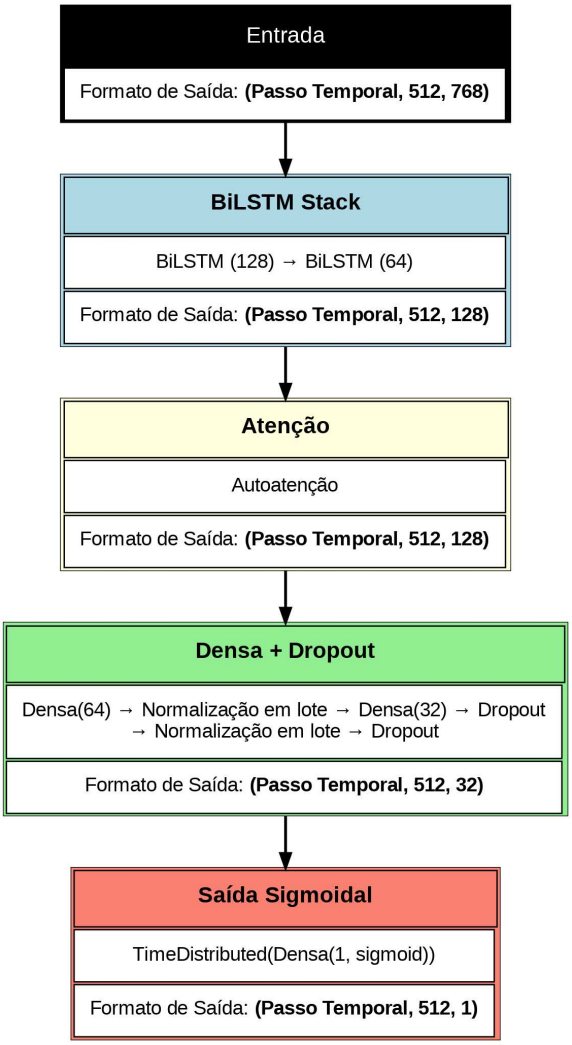
Conforme abordado na formalização, o segmentador precisa tratar da identificação de trechos relevantes para posteriormente classificá-los em FTP, tratando-se, portanto, de uma abordagem hierárquica. O modelo para a identificação dos trechos é composto por BiLSTM (SIAMI-NAMINI; TAVAKOLI; NAMIN, 2019), Redes de Ponteiros (VINYALS; FORTUNATO; JAITLY, 2015) e Perceptron Multicamadas (FALCÃO et al., 2013). Dessa forma, o modelo será treinado no conjunto de dados anotado, obtido na etapa de preparação dos dados.

4.2.4.1 REDE PARA IDENTIFICAÇÃO DOS TRECHOS

Conforme as especificações definidas, a rede para a identificação dos trechos relevantes do texto foi projetada com a intenção de capturar a sequência temporal e as dependências contextuais das palavras, utilizando uma combinação de camadas BiLSTM e

um mecanismo de atenção customizado. O modelo proposto segue a estrutura hierárquica, onde a primeira parte da rede é responsável por identificar a relevância de cada token dentro do contexto do texto. Um resumo da arquitetura é descrito na Figura 6.

Figura 6 – Arquitetura da Rede de Segmentação



Fonte: Autores (2026).

Conforme a Figura 6, entrada da rede é composta por sequências de tokens gerados pelo tokenizador, os quais são processados por uma camada de incorporações que mapeiam cada token para um espaço vetorial contínuo. O modelo usa camadas bidirecionais de LSTM para modelar a dependência temporal dos tokens, permitindo que a rede considere simultaneamente o contexto anterior e posterior de cada elemento da sequência de entrada. A primeira camada BiLSTM possui 128 unidades e outra com 64 unidades, responsáveis por extrair padrões contextuais iniciais do texto. Em seguida, aplica-se um mecanismo de autoatenção, cuja função é atribuir pesos distintos aos tokens da sequência, permitindo que o modelo enfatize as partes mais relevantes do texto para a tarefa de classificação subsequente.

Na sequência, aplica-se uma camada densa com 64 unidades para as representa-

ções contextuais extraídas nas etapas anteriores. Em seguida, aplica-se uma camada de normalização em lote, cujo objetivo é estabilizar a distribuição das ativações durante o treinamento. Posteriormente, uma segunda camada densa com 32 unidades e função de ativação *Rectified Linear Unit* (ReLU) realiza uma transformação adicional das representações aprendidas. Por fim, são aplicadas camadas de *dropout* e normalização em lote, utilizadas respectivamente para desativar aleatoriamente parte dos neurônios durante o treinamento e estabilizar novamente as ativações da rede, contribuindo para a regularização do modelo e redução do risco de sobreajuste.

Finalmente, a saída do modelo é gerada por meio de uma camada *TimeDistributed*, que aplica uma camada densa de forma independente a cada passo temporal da sequência. Essa camada possui uma única unidade com função de ativação sigmoide, responsável por produzir uma probabilidade para cada token. Essa camada permite a classificação de cada token individualmente, indicando se ele pertence ou não a um trecho relevante, de acordo com a etiqueta binária.

O modelo é compilado com o otimizador Adam a uma taxa de aprendizado inicial de $0,001$, função de perda de entropia cruzada binária e F1-score como métrica principal.

Tabela 3 – Arquitetura e hiperparâmetros da rede de segmentação

Camada	**Hiperparâmetros**
Entrada	formato=(512, 768)
BiLSTM 1	unidades=128, dropout=0,2, ativação=tanh
BiLSTM 2	unidades=64, dropout=0,2, ativação=tanh
Autoatenção	consulta=chave=valor=saída BiLSTM pontuação=produto escalar
Densa 1	unidades=64, ativação=ReLU
Normalização em lote	—
Densa 2	unidades=32, ativação=ReLU
Dropout 1	taxa=0,2
Normalização em lote	—
Dropout 2	taxa=0,2
TimeDistributed (Densa)	unidades=1, ativação=sigmoide, L2=0,01
Treinamento	Adam, taxa de aprendizado=0,001, épocas=100 perda=entropia cruzada binária, tamanho do lote=64. métrica=F1-score customizado parada antecipada (paciência=5)

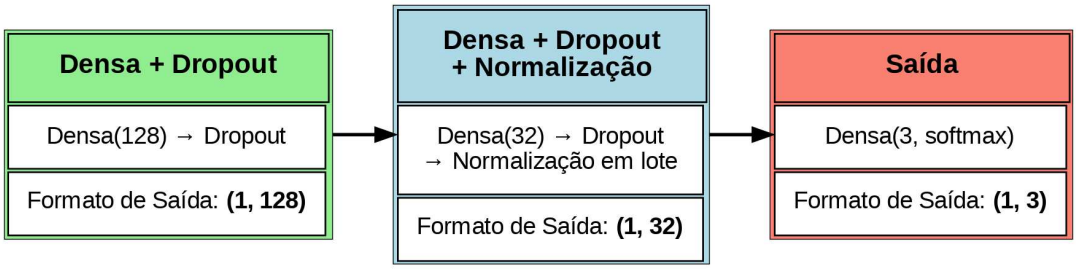
Fonte: Autores (2026).

Por fim, foram empregados lotes (batches) de 64 amostras durante o treinamento, com *dropout* de $0,2$ em diferentes camadas da rede, além de regularização L2 com coeficiente $0,01$ na camada de saída para mitigar o sobreajuste, conforme apresentado na Tabela 3.

4.2.4.2 REDE PARA CLASSIFICAÇÃO DOS TRECHOS

A rede para a classificação dos trechos foi projetada com a intenção de categorizar segmentos identificados previamente pela rede de identificação de trechos relevantes. O modelo proposto utiliza uma arquitetura composta por camadas densas, buscando um equilíbrio entre capacidade preditiva e eficiência computacional. Um resumo da arquitetura proposta é apresentado na Figura 7.

Figura 7 – Arquitetura da Rede de Classificação



Fonte: Autores (2026).

Conforme a Figura 7, a entrada da rede é composta por vetores de características produzidos pelo TF-IDF, os quais são processados por uma camada densa com 128 unidades e ativação ReLU. Essa camada é acompanhada de um regularizador L2 ($\lambda = 0,01$) e por uma camada de dropout com taxa de 50% para reduzir o risco de sobreajuste.

Em seguida, uma segunda camada densa com 32 unidades e ativação ReLU refina a representação dos trechos, sendo também seguida por uma camada de *dropout* com a mesma taxa de 50%. Para estabilizar a aprendizagem e mitigar a covariância interna do deslocamento dos dados, uma camada de normalização em lote (*BatchNormalization*) é aplicada antes da última etapa do modelo.

A saída da rede é gerada por meio de uma camada densa com 3 unidades e ativação softmax, de maneira a produzir probabilidades sobre as 3 classes e permitir a classificação dos trechos em FTP. O seus hiperparâmetros definidos são apresentado na Tabela 4.

Para o treinamento, a função de perda adotada foi a entropia cruzada categórica, apropriada para tarefas de classificação multiclasse. O modelo foi otimizado utilizando o algoritmo Adam, com taxa de aprendizado inicial de $0,001$, e a métrica utilizada para avaliação foi a acurácia.

4.2.4.3 *PIPELINE* COMPLETO

O processo completo pode ser descrito como um pipeline em duas etapas principais. Na primeira, o texto bruto é tokenizado e gerado suas respectivas incorporações para o encaminhamento ao módulo de segmentação, o qual processa a sequência por meio das

Tabela 4 – Arquitetura e hiperparâmetros da rede densa para classificação

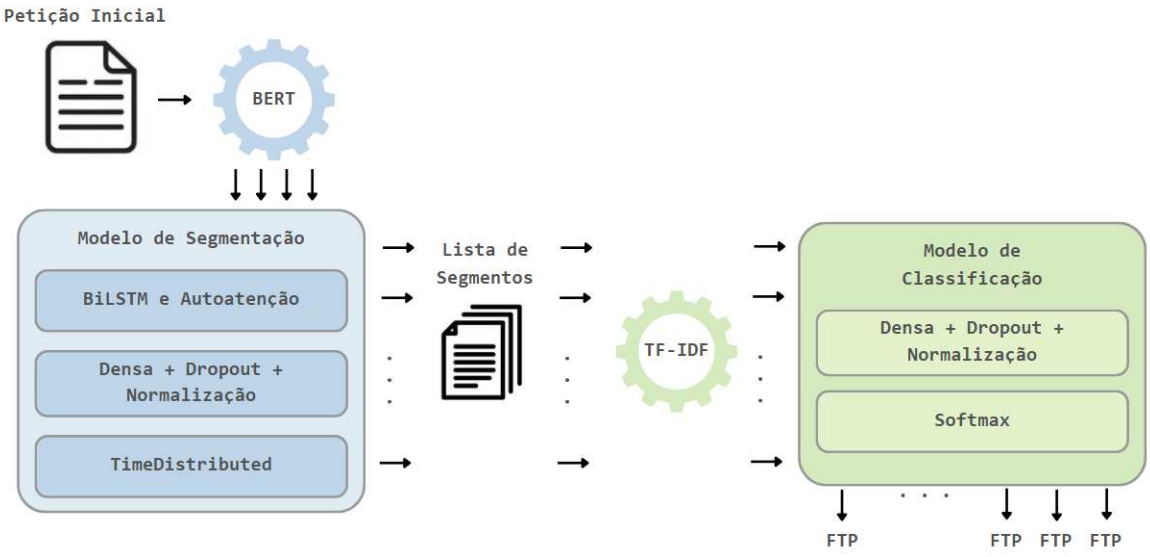
Camada	**Hiperparâmetros**
Entrada	formato=(1, 5.000)
Densa 1	unidades=128, ativação=ReLU, L2=0,01
Dropout 1	taxa=0,5
Densa 2	unidades=32, ativação=ReLU
Dropout 2	taxa=0,5
Normaização em lote 1	—
Densa 3 (Saída)	unidades=3, ativação=Softmax
Treinamensto	Adam, lr=0,001, perda=entropia cruzada categórica, métrica=acurácia

Fonte: Autores (2026).

camadas BiLSTM e do mecanismo de atenção para identificar, token a token, os trechos relevantes. Esses trechos são então extraídos e preparados para a etapa seguinte.

Na segunda etapa, os segmentos selecionados são transformados em vetores de características e enviados ao módulo de classificação. Esse módulo, composto por camadas densas com regularização, realiza a categorização de cada trecho em uma das classes de FTP definidas. O fluxo completo, conectando a segmentação à classificação, é apresentado na Figura 8.

Figura 8 – Arquitetura Completa



Fonte: Autores (2026).

Dessa forma, a arquitetura segue um encadeamento linear: entrada de texto; segmentação de trechos potenciais FTP; classificação em categorias FTP. Essa abordagem modular permite melhorias pontuais em cada etapa sem comprometer o funcionamento geral do pipeline.

4.2.5 AVALIAÇÃO

Para avaliar o desempenho do modelo proposto, foram utilizadas as métricas de acurácia, precisão, revocação e F1-score, sendo esta última também adaptada para melhor adequação à rede de segmentação. Adicionalmente, o modelo proposto foi confrontado com métodos clássicos de classificação e LLMs comuns de mercado.

4.2.5.1 MÉTRICAS DE DESEMPENHO

A acurácia é definida como a proporção de previsões corretas em relação ao total de amostras:

$$\text{Acurácia} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.4}$$

Onde TP representa os verdadeiros positivos, TN os verdadeiros negativos, FP os falsos positivos e FN os falsos negativos.

A precisão mede a proporção de previsões positivas que são corretas e é calculada como:

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{4.5}$$

A revocação, também conhecida como sensibilidade, quantifica a capacidade do modelo de identificar corretamente as amostras positivas:

$$\text{Revocação} = \frac{TP}{TP + FN} \tag{4.6}$$

O F1-score é a média harmônica entre precisão e revocação, garantindo um balanço entre essas métricas:

$$F1 = 2 \times \frac{\text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}} \tag{4.7}$$

Adicionalmente, para a rede de segmentação, cada valor inferido para os tokens é submetido a um limiar de decisão de 0,95 para decisão de tokens pertencentes a um trecho potencial FTP.

Para otimização do treinamento, foi utilizada a função de perda *Binary Cross-Entropy* (BCE) na rede de segmentação, uma vez que esta lida com a previsão de classes binárias para cada token:

$$\text{BCE} = -\frac{1}{N} \sum_{i=1}^{N} [y_i \log(p(y_i)) + (1 - y_i) \log(1 - p(y_i))] \tag{4.8}$$

Onde N indica o número de amostras, y_i o rótulo da instância i, $p(y_i)$ a probabilidade da instância pertencer à determinada classe que, para o caso binário, compreende a adesão à classe 1 ou 0.

Já para a rede de classificação dos trechos segmentados, a função de perda utilizada foi a *Categorical Cross-Entropy* (CCE), adequada para a classificação multiclasse dos segmentos identificados:

$$\text{CCE} = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p(y_{ij})) \tag{4.9}$$

Onde N indica o número de amostras, C o número de classes, y_ij um valor binário que indica se a amostra i pertence à classe j em sua representação one-hot, $p(y_{ij})$ a probabilidade da instância pertencer à classe j.

4.2.5.2 MODELOS CLÁSSICOS E LLMS PARA COMPARAÇÃO

Por fim, a rede foi é submetida a um comparativo de modelos clássicos produzidos por uma combinatória dos parâmetros descritos na Tabela 5.

Os modelos clássicos listados na Tabela 5 são algoritmos comuns em tarefas de classificação textual, a saber: *Logistic Regression*, *Naïve Bayes*, *Random Forest* (ASHBAUGH; ZHANG, 2024), *Support Vector Machine* (SVM) (POOMKA; KERDPRASOP; KERDPRASOP, 2021) *K-Nearest Neighbors* (KNN) (LI et al., 2022). Esses algoritmos foram configurados a partir de diferentes combinações de hiperparâmetros também descritos na Tabela 5. O modelo selecionado foi produto dessa combinatória com o maior valor de acurácia, métrica definida na em 4.2.5.

Além das métricas definidas para quantificar o desempenho dos modelos desenvolvidos, foi construído um conjunto de agentes LLMs com o auxílio do framework LangChain para a segmentação de petições. Um total de quatro modelos foram utilizados nessa tarefa: GPT-3.5, Maritalk-Sabiá-3, LLaMA 3.

Cada LLM foi configurado com um prompt[2] elaborado a partir de experimentações para garantir aderência às características de segmentação definidas neste trabalho, o qual segue a seguinte estrutura: (1) definições das categorias de anotação (fatos, teses, pedidos), (2) instruções de extração (marcar 5 primeiras e últimas palavras de um segmento), (3) critérios de priorização (trechos longos e coerentes), e (4) exemplo completo de entrada/saída.

Em especial, os modelos LLaMA foram executados localmente em uma GPU NVIDIA GeForce GTX 1650, possibilitando uma comparação mais justa com os modelos

[2] https://github.com/ThyagoFRTS/llm-text-segmentation/blob/main/app/langconf/prompt.py

Tabela 5 – Modelos e hiperparâmetros utilizados para classificação

Modelo	**Hiperparâmetros**
Logistic Regression	C = [0,01, 0,1, 1, 10] max_iter = [300] penalty = [l2, l1] solver = [lbfgs, liblinear, saga]
Naïve Bayes	alpha = [0,01, 0,1, 0,5, 1, 2, 5] fit_prior = [True, False] force_alpha = [True, False]
Random Forest	criterion = [gini, entropy, log_loss] max_depth = [None, 10, 20, 30, 50] min_samples_split = [2, 5, 10] min_samples_leaf = [1, 2, 3, 5, 10] max_features = [sqrt, log2, None] n_estimators = [10, 20, 30, 50, 100]
Support Vector Machine	C = [0,1, 1, 10] kernel = [linear, poly, rbf, sigmoid] gamma = [scale, auto] degree = [3, 4, 5] coef0 = [0, 1, 2] max_iter = [300] decision_function_shape = [ovo, ovr]
K-Nearest Neighbors	n_neighbors = [3, 5, 7, 9] weights = [uniform, distance] algorithm = [auto, ball_tree, kd_tree, brute] leaf_size = [5, 10, 20, 30, 50] p = [1, 2]

Fonte: Autores (2026).

hospedados em nuvem, como o GPT e o Sabiá. A comparação entre as segmentações geradas por cada modelo foi realizada com base na métrica Alpha de Krippendorff (KRIPPENDORFF, 2011), que permite avaliar a consistência entre anotações categóricas, sendo adequada para medir a concordância entre as saídas dos modelos e as anotações de referência:

$$\alpha = 1 - \frac{D_o}{D_e} \tag{4.10}$$

Onde D_o representa a discordância observada entre os anotadores (ou modelos, no caso), e D_e a discordância esperada ao acaso. Valores de α próximos de 1 indicam alta concordância, enquanto valores próximos de 0 (ou negativos) indicam baixa ou nenhuma confiabilidade. Na literatura, valores de $\alpha \geq 0,800$ são considerados confiáveis, enquanto valores abaixo de $0,670$ indicam fraca confiabilidade (HAYES; KRIPPENDORFF, 2007). A

implementação completa do programa de segmentação encontra-se disponível no repositório GitHub do projeto[3].

4.2.6 IMPLANTAÇÃO

A última etapa do processo CRISP-DM consiste na disponibilização dos modelos desenvolvidos para uso prático. Visando a transparência e o compartilhamento de conhecimento, tanto o código-fonte quanto os modelos treinados serão disponibilizados no repositório GitHub do autor [4].

[3] https://github.com/ThyagoFRTS/llm-text-segmentation
[4] https://github.com/ThyagoFRTS/petition-segmentation-ftp

5 RESULTADOS E DISCUSSÕES

Nesta seção serão apresentados os resultados obtidos a partir da avaliação dos modelos desenvolvidos para as tarefas de segmentação e classificação de texto. A análise será dividida em duas partes principais, cada uma focada em uma das redes, de forma a possibilitar uma discussão sobre o desempenho de cada modelo em relação às suas respectivas funções.

5.1 IDENTIFICAÇÃO DE TRECHOS

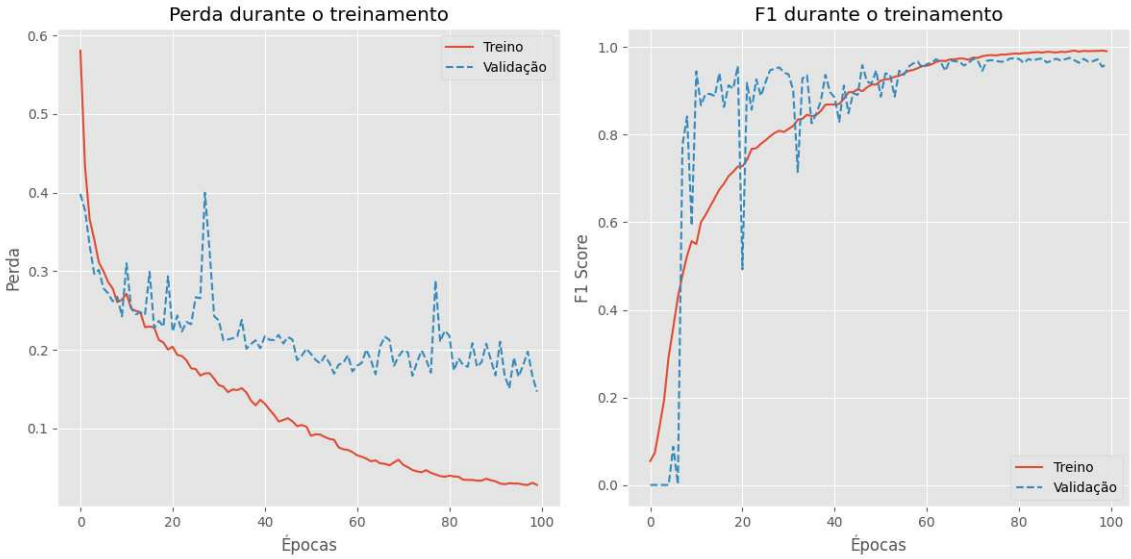
O treinamento do modelo foi realizado com uma fração de 80% dos dados e a fração restante foi destinada para validação. O processo de treinamento foi conduzido com um gerador de dados que permite que os exemplos sejam processados em lotes e carregados em tempo de execução. A escolha do gerador também teve influência do formato de entrada da rede por constituir-se de duas dimensões variáveis (número de documentos e quantidade de blocos por documento) e uma fixa (tamanho do bloco). Para tratar essa característica, os blocos foram tratados como independentes durante o treinamento, removendo-se a noção de documentos no conjunto de dados. Essa abordagem permite o embaralhamento dos dados sem quaisquer perdas de contexto, uma vez que os blocos já carregam o contexto do bloco anterior com aplicação do janelamento.

O modelo foi treinado por 100 épocas, com monitoramento do desempenho nos dados de validação e treinamento. A melhor métrica F1 no conjunto de treinamento foi de 0,990, alcançada na época 93, enquanto a melhor F1 no conjunto de validação foi de 0,975, registrada na época 92. Em termos de perda, o menor valor no treinamento foi de 0,028, ocorrido na época 100, e no conjunto de validação, a menor perda foi de 0,147, alcançada na época 98. No final do treinamento, os valores de F1 foram 0,990 no conjunto de treinamento e 0,961 no conjunto de validação, enquanto as perdas finais foram de 0,028 e 0,14, respectivamente. A Figura 9 mostra o comportamento da função de perda e da métrica F1 ao longo de todo o treinamento.

Durante o treinamento do modelo de segmentação, observou-se que a função de perda BCE diminuiu progressivamente ao longo das épocas, enquanto o F1-score aumentou e se estabilizou em um intervalo entre 0,8 e 1 pontos. No conjunto de validação, a loss também apresentou uma queda inicial, mas posteriormente passou a oscilar entre 0,2 e 0,4, enquanto o F1-score seguiu uma tendência crescente e se estabilizou em um nível semelhante ao do conjunto de treino.

O fato do F1-score nos dados de validação se estabilizar em um valor alto sugere que o modelo está conseguindo generalizar bem para novos dados. No entanto, a variação

Figura 9 – Treinamento e Validação do Segmentador



Fonte: Autores (2026).

na loss de validação pode indicar a presença de instâncias mais difíceis dentro do conjunto de validação, que causam maior incerteza na predição, em especial nos casos onde há sequências longas de trechos anotados, onde o modelo tende a inserir tokens abaixo do limiar (0,95) no meio desses trechos, produzindo a subdivisão desses trechos. Como a BCE penaliza mais fortemente previsões erradas com alta confiança, mesmo que o modelo esteja fazendo boas predições globais, algumas incertezas em exemplos específicos podem resultar nessa oscilação.

Ademais, observa-se uma queda gradual na perda durante as primeiras épocas, indicando uma dificuldade da rede em identificar os limites dos segmentos e a quantidade de tokens pertencentes corretamente. Esse comportamento, embora melhor tratado nas épocas subsequentes, pode ser observado na Figura 10.

Figura 10 – Trecho Anotado x Saída do Modelo (Decodificado)

- Trecho real

```
['art 186 aquele que por acao ou omissao voluntaria negligencia ou imprudencia violar direito ou causar prejuizo a outrem ainda que
exclusivamente moral comete ato ilicito',
 'art 6º vi a efetiva prevencao e reparacao de danos patrimoniais e morais individuais coletivos e difusos',
 'fornecedor de servicos responde independentemente da existencia de culpa pela reparacao dos danos causados aos consumidores por
defeitos relativos a prestacao dos servicos bem como por informacoes insuficientes ou inadequados sobre sua fruicao e ricos conforme
discriminado patente esta o direito pleiteado uma vez que o requerido infringiu todos os preceitos legais acima destacados ademais a
iletrada que nao sabe ler nem escrever somente se obriga contratualmente por instrumento publico sendo nulo o contrato que se vincula
analfabeto atraves de impressao digital e testemunhas instrumentarias porque o documento particular para ser valido deve ser redigido e
firmado ou somente firmado pelo signatario cc art 221 1ª parte',
 'pois a requerida ainda continua efetuando os indevidos descontos do']
```

- Trecho inferido

```
['##e que por acao ou omissao voluntaria negligencia ou imprudencia violar direito ou causar prejuizo a outrem ainda que exclusivamente
moral comete at',
 'efetiva prevencao e reparacao de danos patrimoniais e morais individua',
 '##or de servicos responde independentemente da existencia de culpa pela reparacao dos danos causados aos consumidores por defeitos
relativos a prestacao dos servicos bem como por informacoes insuficientes ou inadequados sobre sua fruicao e ricos conforme
discriminado patente esta o direito pleiteado uma vez que o requerido infringiu todos os preceitos legais acima des',
 'iletrada que nao sabe ler nem escrever somente se obriga contratualmente por instrumento publico sendo nulo o contrato que se vincula
analfabeto atraves de impressao digital e testemunhas instrumentarias porque o documento particular para ser valido deve ser redigido e
firmado ou somente firmado pelo signatario cc',
 '##rida ainda continua efetuando os indevidos des']
```

Fonte: Autores (2026).

A utilização do BERT para tokenização foi eficaz na identificação de palavras complexas, com a marcação "##" indicando palavras que foram divididas em dois ou mais tokens, o que é característico deste modelo. No entanto, observou-se uma tendência ao corte abrupto das palavras e dos segmentos, especialmente quando o modelo encontrou palavras que exigiam múltiplos tokens. Por exemplo, trechos como "fornecedor de servico" e "a requerida ainda continua efetuando" foram segmentados de forma incompleta, resultando em palavras truncadas, como "##or" e "##rida", que indicam uma quebra inadequada nas palavras. Além disso, o modelo apresentou dificuldades ao segmentar o início das frases de forma precisa, perdendo o contexto de palavras-chave importantes, como "art 186", o que pode afetar a compreensão e a precisão da segmentação.

As segmentações inferidas foram contrastadas com as produzidas pelos LLMs e estão apresentadas na Tabela 6, a qual mostra a média do valor α das anotações de 20 documentos de avaliação. Na literatura, valores de $\alpha \geq 0,800$ são considerados confiáveis, enquanto valores abaixo de $0,667$ indicam fraca confiabilidade (KRIPPENDORFF, 2011). Para fins de análise, a segmentação denominada SegNetFTP corresponde à fusão da rede de segmentação e da rede de classificação desenvolvidas neste trabalho. A Tabela 6 apresenta os valores de α obtidos ao comparar anotações geradas por LLMs (GPT, Sabiá, LLaMA), por sua combinação, e pela arquitetura proposta neste trabalho (SegNetFTP), todas em relação à anotação manual de referência:

Tabela 6 – Média de valores de Krippendorff's Alpha para diferentes combinações de anotadores comparadas à anotação manual

Comparação	α
GPT vs. Manual	-0,002
Sabiá vs. Manual	-0,001
LLaMA vs. Manual	0,015
GPT + Sabiá + LLaMA	0,034
GPT + Sabiá + LLaMA + SegNetFTP	0,029
GPT + Sabiá + LLaMA + SegNetFTP vs. Manual	0,074
SegNetFTP vs. Manual	**0,634**

Fonte: Autores (2026).

Observa-se que todos os modelos de linguagem natural (LLMs), tanto isoladamente quanto em combinação, apresentaram níveis de concordância extremamente baixos em relação à anotação humana, com alphas próximos de zero ou até negativos. O LLaMA apresentou o valor mais alto entre os LLMs (0,0146), mas ainda muito distante dos níveis mínimos aceitáveis.

A combinação entre os três LLMs não trouxe ganhos ($\alpha = 0,0400$), e a inclusão da SegNetFTP nessa combinação tampouco resultou em melhorias significativas ($\alpha = 0,0287$). Mesmo a inclusão da anotação manual ao grupo elevou o alpha para apenas 0,0736, um

incremento modesto e ainda insuficiente.

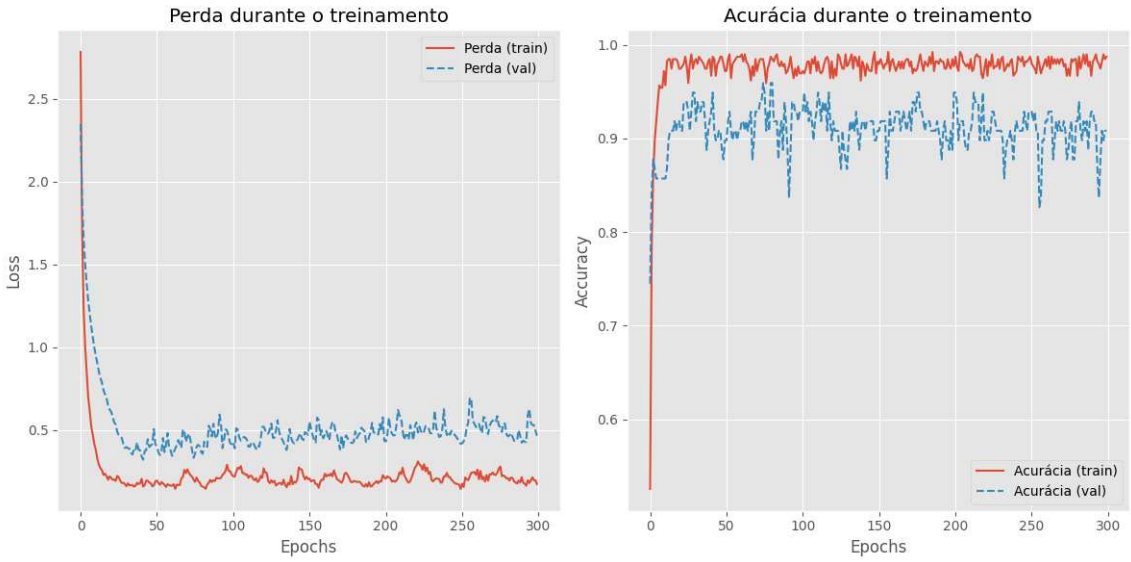
O único resultado que se destaca é o obtido pela comparação direta entre a arquitetura SegNetFTP e a anotação manual ($\alpha = 0,6337$), indicando um grau de concordância consideravelmente superior ao observado com os LLMs. Embora este valor ainda esteja abaixo do limiar de 0,800 recomendado, representa um avanço notável em relação às abordagens baseadas exclusivamente em LLMs.

Além disso, o valor intermediário de α também aponta para limitações que podem estar relacionadas a fatores como a ambiguidade intrínseca da tarefa, eventuais inconsistências na própria anotação de referência, ou limitações na abrangência e balanceamento no conjunto de treinamento.

5.2 CLASSIFICAÇÃO DE TRECHOS

De maneira similar à rede de segmentação, o treinamento do modelo de classificação também foi realizado com a distribuição de dados de 80% para treino e e 20% validação. Dentre as 300 épocas de treinamento, a maior acurácia observada foi de 0,995 para dados de treinamento e 0,949 para validação, enquanto as menores perdas foram de 0,112 para treino e 0,335 para validação. A Figura 11 exibe o comportamento da perda e acurácia ao longo das épocas.

Figura 11 – Treino e Validação do Classificador

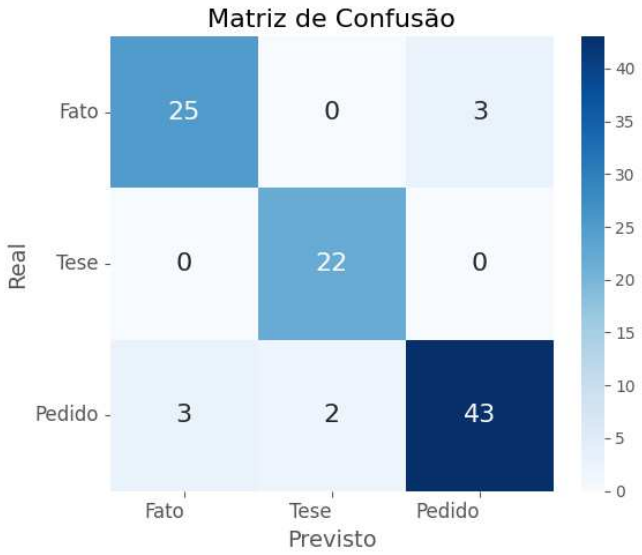


Fonte: Autores (2026).

De acordo com a Figura 11, a curva de aprendizado mostra que o modelo começa a convergir nas primeiras 20 épocas, sugerindo que o treinamento adicional não resultaria em melhorias significativas. No entanto, a menor perda de validação só ocorre na época 64, sugerindo que a acurácia estabilizou rapidamente, mas a função de perda ainda não se encontrava em seu mínimo local.

Esse comportamento é refletido também na matriz de confusão apresentada na Figura 12 e 13, as quais fornecem uma visão detalhada dos desempenhos dos modelos em relação às classes Fato, Tese e Pedido. Apesar da SegNetFTP ter atingido um bom nível de acurácia em seu treinamento, os erros de classificação, especialmente entre Fato e Pedido, sugerem que há nuances que o modelo ainda precisa aprender, o que pode estar relacionado à complexidade do próprio problema, uma vez que os modelos clássicos também demonstraram as mesmas dificuldades, e à rapidez com que o modelo se estabilizou.

Figura 12 – Matriz de Confusão



Fonte: Autores (2026).

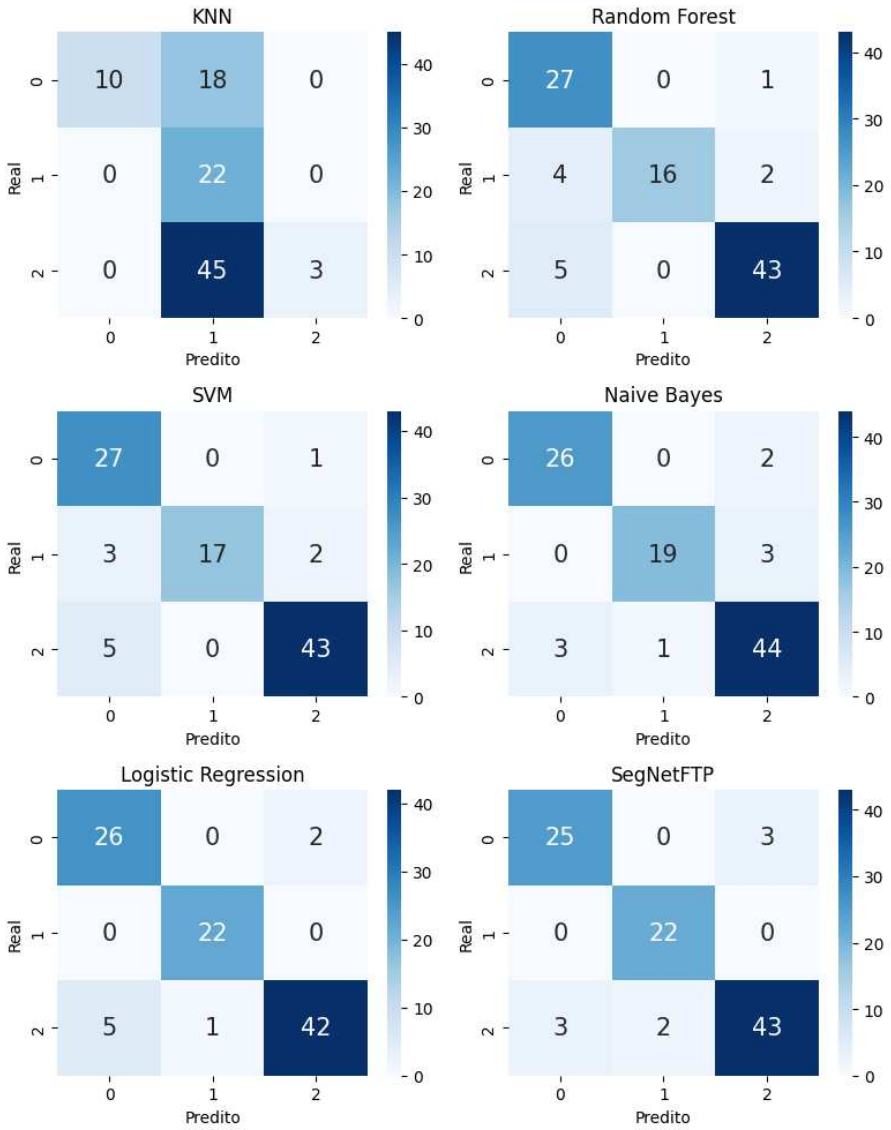
A Figura 13 apresenta as matrizes de confusão de todos os modelos clássicos utilizados para comparação, bem como a do modelo proposto. É possível observar que, embora métodos como SVM, Naïve Bayes e Logistic Regression obtenham desempenhos próximos, não apresentam a mesma consistência do SegNetFTP na classificação correta das três classes. Ao analisar os valores das matrizes, nota-se que o KNN apresentou alta taxa de erros na classe Fato (18 trechos classificados incorretamente como Tese), possivelmente em razão de sua dependência de medidas de distância, que tornam o algoritmo sensível a representações textuais de alta dimensionalidade e à sensibilidade de conjunto de palavras que comumente aparecem diferentes classes. Por outro lado, o Random Forest, apesar de reduzir drasticamente os erros na classe Fato em relação ao KNN, ainda apresentou confusões entre Fato e Pedido. Esse comportamento pode estar relacionado à forma como o modelo combina previsões de várias árvores de decisão. Em trechos cujo conteúdo apresenta características mistas das duas classes, diferentes árvores podem chegar a conclusões distintas, e a votação final pode resultar na classificação incorreta.

O SVM e o Naïve Bayes apresentaram desempenho intermediário, com o SVM errando principalmente entre Fato e Pedido (5 ocorrências) e o Naïve Bayes apresentando

poucos erros na classe Tese (3 casos), o que pode ser explicado pela suposição de independência entre as palavras, limitando sua capacidade de capturar dependências contextuais mais complexas. Já a Logistic Regression apresentou resultados próximos aos do modelo proposto, com confusões restritas às classes Fato e Pedido, sugerindo que modelos lineares conseguem capturar padrões relevantes quando as classes são bem separáveis, mas não necessariamente as nuances semânticas mais sutis.

Por fim, o SegNetFTP, embora apresente alguns erros entre Fato e Pedido (3 e 2 ocorrências, respectivamente), mantém um desempenho mais equilibrado nas três classes, indicando que arquiteturas baseadas em redes podem lidar melhor com a variabilidade linguística dos textos jurídicos e a ambiguidade de termos que aparecem em contextos múltiplos.

Figura 13 – Matrizes de Confusão



Fonte: Autores (2026).

A partir da matriz de confusão 12, é possível observar que a SegNetFTP tem um

bom desempenho geral, com alta taxa de acerto nas classes Tese (22 de 22) e Pedido (43 de 45). No entanto, existem algumas dificuldades em classificar corretamente a classe Fato, com 3 erros de classificação para Pedido e 3 para Tese. Esse comportamento pode estar relacionado à similaridade entre trechos mais longos, os quais podem apresentar tópicos de mais de uma classe, onde certas palavras ou frases podem ser ambíguas. A classe Pedido, por exemplo, tende a apresentar termos relacionados a serviços ou pagamentos, que o modelo pode associar erroneamente ao Fato, como mostrado na Tabela 7.

Tabela 7 – Exemplos de Erros de Classificação

Texto	**Classe Real**	**Classe Predita**
ressaltese conforme explicado ligacao autora so contratou plano virtude aparelho ortodontico atendente confirmou garantiu cobertura toda colocacao tratamento plano.	Fato	Pedido
evidente ilegalidade propaganda promovida re alem agressiva coercitiva relacao clientes evidencia risco vazamento dado pode evitado caso cliente promova pagamento taxa em servicos site questao...	Fato	Pedido
parte demandada responsavel legal imovel numero condominiorequerente nesta condicao responsavel pelao pagamento despesas contribuicoes condominiais...	Fato	Pedido
manutencao referida constricao contas desta requerente deve classificada indevida sendo assim medida rigor deflagracao provimento jurisdicional venha possibilitar liberacao contas questao.	Pedido	Fato
enquanto magistrado servidor autorizado nao determinarem desbloqueio transferencia valores permanecem bloqueados contas aplicacoes financeiras atingidas ressalvada hipotese vencimento contrato aplicacao financeira...	Pedido	Fato

Fonte: Autores (2026).

Nos exemplos apresentados, o modelo classifica erroneamente como Pedido trechos que descrevem fatos geralmente quando há palavras associadas a transações financeiras ou serviços. Isso ocorre no trecho 1, onde o modelo confunde o fato de um contrato com um pedido de cobertura de tratamento. Esse padrão sugere que o modelo tem uma tendência a associar palavras-chave como “plano”, “pagamento” e “cobertura” a solicitações, o que reforça a hipótese de que o modelo pode estar influenciado por um viés semântico relacionado ao contexto dos serviços jurídicos. Por outro lado, nos casos em que a classe Pedido é classificada como Fato, como no exemplo 4, o modelo pode estar confundindo termos relacionados a ordens judiciais ou solicitações com declarações de fato, evidenciando

a complexidade da tarefa de discriminar entre os dois tipos de segmentos. Já o exemplo 5 representa uma falha na recuperação de seu rótulo não corrigido na fase de validação, o trecho representa um parágrafo do regulamento BACENJUD[1] e serve para embasar o entendimento da tese defendida pelo autor.

Tabela 8 – Acurácia dos Modelos de Classificação

Modelo	**Acurácia**
KNN	0,357
Random Forest	0,878
SVM	0,888
Naïve Bayes	0,908
Logistic Regression	0,918
SegNetFTP	0,949

Fonte: Autores (2026).

Os resultados quantitativos indicam que, embora o modelo SegNetFTP tenha obtido a melhor acurácia geral, a diferença para métodos tradicionais, mesmo que superior, não foi expressiva. Esse comportamento sugere que, para a tarefa de classificação de trechos jurídicos, modelos mais simples podem capturar padrões relevantes de forma competitiva, especialmente em cenários com dados limitados. No entanto, a maior flexibilidade do SegNetFTP pode oferecer vantagens em bases de dados mais extensas ou heterogêneas, onde a complexidade semântica tende a ser ainda maior.

[1] https://www.tjsc.jus.br/web/corregedoria-geral-da-justica/bacenjud

6 CONSIDERAÇÕES FINAIS

O presente trabalho teve como objetivos: (i) realizar um levantamento das principais técnicas de segmentação textual na literatura; (ii) preparar um conjunto de dados específico de petições iniciais com anotações para os segmentos fato, tese e pedido; (iii) implementar e avaliar modelos de redes neurais para segmentação textual; (iv) implementar e avaliar modelos para classificação dos segmentos em FTP; e (v) comparar o desempenho dos modelos propostos com abordagens clássicas e modelos de linguagem de grande porte (LLMs).

Os resultados obtidos indicam que os objetivos foram plenamente alcançados. A arquitetura proposta para a segmentação de petições iniciais em fato, tese e pedido demonstrou ser capaz de se adaptar à tarefa, apresentando resultados consistentes na delimitação dos segmentos textuais. O modelo alcançou um F1-score de $0,975$ nos dados de validação, confirmando sua viabilidade para tarefas de segmentação semântica em textos jurídicos para essas 3 categorias em petições iniciais. Esses resultados indicam que a SegNetFTP é capaz de identificar de forma consistente os limites entre os segmentos, mesmo em cenários de variação estilística e estrutural.

A análise dos padrões probabilísticos revelou que a probabilidade de uma palavra ser o início de um segmento aumenta nas proximidades das marcações de início e diminui ao se aproximar do término do segmento, o que sugere que o modelo consegue captar indícios estruturais da segmentação semântica. Observou-se também que, em alguns casos, o início e o término de segmentos coincidiram com palavras representadas por mais de um token, sugerindo que a tokenização adotada pode influenciar na precisão das marcações.

Além disso, os vieses dos anotadores influenciaram na granularidade do modelo. Diferenças na interpretação dos critérios de anotação, em especial na granularidade dos textos, levaram a variações nos padrões de segmentação, refletindo a subjetividade intrínseca ao processo manual de anotação e influenciando o modelo a produzir sequências de baixa granularidade. Esse fator deve ser considerado em futuras melhorias do modelo, seja por meio de refinamentos nas diretrizes de anotação, como na definição de procedimentos para selecionar trechos dentro de um parágrafo, ou pela adoção de técnicas que reduzam a dependência desses vieses, especialmente no que diz respeito à granularidade das segmentações.

A comparação com LLMs populares demonstrou que os modelos de linguagem capturam bem os tópicos e argumentos do texto; no entanto, possuem um alto caráter interpretativo, sugerindo que não performam bem para tarefas em que a fidelidade do texto é importante. A SegNetFTP, por outro lado, demonstrou viabilidade para esse tipo

de tarefa.

O uso de janelas deslizantes com sobreposição contribuiu diretamente para a manutenção da coerência entre segmentos, permitindo ao modelo explorar padrões internos ao texto e preservar a continuidade da segmentação, mesmo sem mecanismos explícitos para considerar blocos anteriores.

No tocante à classificação dos segmentos em fato, tese e pedido, o modelo neural alcançou acurácia de 0,949, representando um aumento de 0,030 pontos em relação ao Logistic Regression, modelo clássico de classificação mais bem ajustado à tarefa de classificação, confirmando sua viabilidade para tarefas de categorização textual em documentos jurídicos.

Por fim, o modelo ainda apresenta possibilidades de aprimoramento. Futuras versões podem explorar abordagens que levem em conta o contexto global do documento, integração de técnicas de pós-processamento para ajuste fino das marcações e estratégias para mitigar a influência da tokenização na segmentação. Essas melhorias podem contribuir para a obtenção de resultados mais precisos e alinhados com a estrutura dos documentos jurídicos analisados.

REFERÊNCIAS

ABDUSALOMOVNA, T. D. et al. Text mining. *European Journal of Interdisciplinary Research and Development*, v. 13, p. 284–289, 2023. Citado na página 21.

ABUBAKAR, H. D.; UMAR, M.; BAKALE, M. A. Sentiment classification: Review of text vectorization methods: Bag of words, tf-idf, word2vec and doc2vec. *SLU Journal of Science and Technology*, Sule Lamido University Kafin Hausa, v. 4, n. 1, p. 27–33, 2022. Citado na página 21.

ALAMMARY, A. S. Bert models for arabic text classification: a systematic review. *Applied Sciences*, MDPI, v. 12, n. 11, p. 5720, 2022. Citado na página 22.

ARTIFEX. *PyMuPDF documentation*. [S.l.], 2025. Acesso em: 2 mar. 2025. Disponível em: <https://pymupdf.readthedocs.io>. Citado na página 38.

ASHBAUGH, L.; ZHANG, Y. A comparative study of sentiment analysis on customer reviews using machine learning and deep learning. *Computers*, MDPI, v. 13, n. 12, p. 340, 2024. Citado 2 vezes nas páginas 23 e 46.

AUMILLER, D. et al. Structural text segmentation of legal documents. In: *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*. [S.l.: s.n.], 2021. p. 2–11. Citado 2 vezes nas páginas 17 e 35.

BARION, E. C. N.; LAGO, D. Mineração de textos. *Revista de ciências exatas e tecnologia*, v. 3, n. 3, p. 123–140, 2008. Citado na página 21.

BRAGANÇA, F. O progresso da justiça digital no brasil: da urna eletrônica ao programa 4.0. *Revista Eletrônica de Direito Processual*, v. 24, n. 3, 2023. Citado na página 15.

BUSSAB, W. d. O.; MORETTIN, P. A. Estatística básica. In: *Estatística básica*. [S.l.: s.n.], 2010. p. xvi–540. Citado na página 37.

CNJ, C. N. de J. *A Justiça em Números - Relatório Analítico 2024*. Brasília: [s.n.], 2024. Citado na página 14.

DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. Citado 3 vezes nas páginas 22, 23 e 38.

FALCÃO, H. S. et al. Classificação de vagas de estacionamento com utilização de rede perceptron multicamadas. *Revista de Sistemas de Informação da FSMA, Visconde de Araújo*, v. 12, p. 41–48, 2013. Citado 2 vezes nas páginas 23 e 40.

FILHO, M. S. M.; JUNQUILHO, T. A. Projeto victor: perspectivas de aplicação da inteligência artificial ao direito. *Revista de Direitos e Garantias Fundamentais*, Faculdade de Direito de Vitória, v. 19, n. 3, p. 218–237, 2018. Citado na página 15.

GAUTAM, S. et al. Leveraging text mining techniques for automated information analysis. In: . [S.l.: s.n.], 2023. p. 1–6. Citado na página 21.

GAYEN, A. et al. Destiny of legal petition: Accept or reject?: A machine learning approach to predict the legal petition's initial decision. In: *2024 IEEE Region 10 Symposium (TENSYMP)*. [S.l.: s.n.], 2024. p. 1–6. Citado na página 35.

GONZALEZ, M.; LIMA, V. L. S. Recuperação de informação e processamento da linguagem natural. In: SN. *XXIII Congresso da Sociedade Brasileira de Computação*. [S.l.], 2003. v. 3, p. 347–395. Citado na página 16.

GRANGEIA, M. A. D. A crise de gestão do poder judiciário: o problema, as consequências e os possíveis caminhos para a solução. *Brasília, DF: Escola Nacional de Formação e Aperfeiçoamento de Magistrados. Recuperado em*, v. 18, 2011. Disponível em: <url(https://www.enfam.jus.br/wp-content/uploads/2013/01-/2099_Des_Marcos_Alaor_Artigo_ENFAM_28_4_2011_editado.pdf)>. Citado na página 14.

GRAVES, A.; GRAVES, A. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, Springer, p. 37–45, 2012. Citado na página 24.

GUIMARÃES, S. A. d. S.; ROCHA, E. S. S.; MUGNAINI, R. Estudo cientométrico da atividade acadêmica sobre as temáticas de humanidades digitais e big data nas universidades estaduais paulistas. *Encontros Bibli*, SciELO Brasil, v. 28, p. e90566, 2023. Citado na página 14.

HAYES, A. F.; KRIPPENDORFF, K. Answering the call for a standard reliability measure for coding data. *Communication methods and measures*, Taylor & Francis, v. 1, n. 1, p. 77–89, 2007. Citado na página 47.

HAYKIN, S. *Neural networks and learning machines, 3/E*. [S.l.]: Pearson Education India, 2009. Citado na página 23.

JÚNIOR, A. P. de C.; CALIXTO, W. P.; CASTRO, C. H. A. de. Aplicação da inteligência artificial na identificação de conexões pelo fato e tese jurídica nas petições iniciais e integração com o sistema de processo eletrônico. *CNJ*, p. 9, 2020. Citado 5 vezes nas páginas 17, 19, 29, 31 e 35.

KIDO, G. S.; JUNIOR, S. B.; MORIGUCHI, S. N. *Comparação entre TF-IDF e LSI para pesagem de termos em micro-blog. n.* [S.l.]: May, 2014. Citado 2 vezes nas páginas 21 e 40.

KRIPPENDORFF, K. *Computing Krippendorff's alpha-reliability*. [S.l.]: Citeseer, 2011. Citado 2 vezes nas páginas 47 e 51.

LATTISI, T. et al. Semantic segmentation of text using deep learning. *Computing and Informatics*, SVK, v. 2022, n. 1, p. 78–97, 2022. Citado na página 17.

LI, F. et al. A segment enhanced span-based model for nested named entity recognition. *Neurocomputing*, Elsevier, v. 465, p. 26–37, 2021. Citado na página 16.

LI, J. et al. Neural text segmentation and its application to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, IEEE, v. 34, n. 2, p. 828–842, 2020. Citado 3 vezes nas páginas 17, 29 e 31.

LI, Q. et al. A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, ACM New York, NY, v. 13, n. 2, p. 1–41, 2022. Citado na página 46.

LOURENÇO, M. d. V. N. S. *A argumentação na Petição Inicial*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Norte, 2008. Citado na página 19.

LUKASIK, M. et al. Text segmentation by cross segment attention. *arXiv preprint arXiv:2004.14535*, 2020. Citado 2 vezes nas páginas 29 e 31.

MACHADO, F. de V.; COLOMBO, C. Inteligência artificial aplicada à atividade jurisdicional: desafios e perspectivas para sua implementação no judiciário. *Revista da Escola Judicial do TRT4*, v. 3, n. 5, p. 117–141, 2021. Disponível em: <url(https://rejtrt4.emnuvens.com.br/revistaejud4/article/view/113/95)>. Citado na página 15.

MAGALHÃES, G. P. G. de; LUCAS, E. R. G. de. A importância de uma boa elaboração da petição inicial. *Revista Eletrônica da OAB-RJ*, 2025. Citado na página 19.

MAGALHÃES, R. A.; FREITAS, F. O. A morosidade do poder judiciário e sua interferência nas relações contratuais. *Revista Jurídica Cesumar-Mestrado*, v. 23, n. 3, p. 701–711, 2023. Disponível em: <url(https://periodicos.unicesumar.edu.br/index.php/revjuridica/article/view/10707/7505)>. Citado na página 14.

MALIK, V. et al. Ildc for cjpe: Indian legal documents corpus for court judgment prediction and explanation. *arXiv preprint arXiv:2105.13562*, 2021. Citado na página 30.

MARINATO, M. S. et al. Classificação automática de petições iniciais usando classificadores combinados. In: SBC. *Brazilian e-Science Workshop (BreSci)*. [S.l.], 2022. p. 89–96. Citado na página 20.

MELCARNE, A.; RAMELLO, G. B.; SPRUK, R. Is justice delayed justice denied? an empirical approach. *International Review of Law and Economics*, v. 65, p. 105953, 2021. ISSN 0144-8188. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0144818820301666>. Citado na página 14.

MORAIS, E. A. M.; AMBRÓSIO, A. P. L. Mineração de textos. *Relatório Técnico–Instituto de Informática (UFG)*, 2007. Citado na página 21.

NETTO, S. S. M.; CAMPAGNOLI, A. d. F. F.; GARCIA, A. S. O uso da tecnologia no poder judiciário em busca da razoável duração do processo e da eficiência na administração pública, com ênfase no robô judiciário 1 do trt da 9ª região. *Humanidades & Inovação*, v. 8, n. 48, p. 175–186, 2021. Citado 2 vezes nas páginas 14 e 15.

OLIVEIRA, A. F. de; FERNANDES, A.; KERSCHER, D. Petição inicial. *JICEX*, v. 4, n. 4, 2014. Citado na página 19.

OLIVEIRA, L. E. d. R. M. Petição inicial. *Petição inicial*, 2023. Citado na página 19.

PAK, I.; TEH, P. L. Text segmentation techniques: a critical review. *Innovative Computing, Optimization and Its Applications: Modelling and Simulations*, Springer, p. 167–181, 2017. Citado na página 16.

PARDO, T. A. S. *SENTER: um segmentador sentencial automático para o português do Brasil*. [S.l.]: Série de Relatórios do NILC. NILC-TR-06-01. São Carlos-SP, Janeiro, 6p, 2006. Citado na página 16.

PARDO, T. A. S.; NUNES, M. d. G. V. Segmentaçao textual automática: Uma revisao bibliográfica. 2003. Citado na página 16.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 40.

POLO, F. M. et al. Legalnlp-natural language processing methods for the brazilian legal language. In: SBC. *Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional*. [S.l.], 2021. p. 763–774. Citado 2 vezes nas páginas 16 e 35.

POOMKA, P.; KERDPRASOP, N.; KERDPRASOP, K. Machine learning versus deep learning performances on the sentiment analysis of product reviews. *International Journal of Machine Learning and Computing*, v. 11, n. 2, p. 103–109, 2021. Citado na página 46.

PORTO, F. R. A "corrida maluca" da inteligência artificial no poder judiciário. *Inteligência artificial e aplicabilidade prática no direito. Brasília: Conselho Nacional de Justiça*, p. 103–130, 2022. Disponível em: <url(https://www.cyberleviathan.com-.br/_files/ugd/212c00_d35929a913c741a191814de41a7c2143.pdf)>. Citado na página 14.

QUEIROZ, A. M. de; BUENO, P. L. N.; LISBINO, J. K. T. O impacto da inteligência artificial na advocacia brasileira: benefícios e desafios no setor jurídico. *Revista Ibero-Americana de Humanidades, Ciências e Educação*, v. 10, n. 11, p. 2697–2712, 2024. Citado na página 19.

RAMOS, J. L. C. et al. Crisp-edm: uma proposta de adaptação do modelo crisp-dm para mineração de dados educacionais. In: SBC. *Anais do XXXI Simpósio Brasileiro de Informática na Educação*. [S.l.], 2020. p. 1092–1101. Citado na página 34.

RAMPIM, T.; IGREJA, R. L. Acesso à justiça e transformação digital: um estudo sobre o programa justiça 4.0 e seu impacto na prestação jurisdicional. *Direito Público*, v. 19, n. 102, 2022. Citado na página 15.

REZENDE, S. O. *Sistemas inteligentes: fundamentos e aplicações*. [S.l.]: Editora Manole Ltda, 2003. Citado na página 20.

REZENDE, S. O.; MARCACINI, R. M.; MOURA, M. F. O uso da mineração de textos para extração e organização não supervisionada de conhecimento. Revista de Sistema de Informação da FSMA, Macaé, n. 7, p. 7-21, 2011., 2011. Citado na página 21.

RIBEIRO, R. et al. Predicting the tear strength of woven fabrics via automated machine learning: an application of the crisp-dm methodology. SCITEPRESS–Science and Technology Publications, 2020. Citado na página 34.

ROCHA, C. I. da; RIBEIRO, L. A.; JEVAUX, G. A. Acesso à justiça, o processo 4.0 e as novas tecnologias. 2023. Citado na página 15.

SAVELKA, J. et al. Sentence boundary detection in adjudicatory decisions in the united states. *Traitement automatique des langues*, v. 58, p. 21, 2017. Citado na página 30.

SCHNEIDER, A. F.; MOREIRA, A. C. S. A justiça 4.0 como ferramenta de eficiência: O caso ambiental. *Revista da EMERJ*, v. 25, n. 2, p. 22–30, 2023. Citado na página 15.

SHAHID, F.; ZAMEER, A.; MUNEEB, M. Predictions for covid-19 with deep learning models of lstm, gru and bi-lstm. *Chaos, Solitons & Fractals*, Elsevier, v. 140, p. 110212, 2020. Citado 2 vezes nas páginas 23 e 25.

SHEARER, C. The crisp-dm model: the new blueprint for data mining. *Journal of data warehousing*, THE DATA WAREHOUSE INSTITUTE, v. 5, n. 4, p. 13–22, 2000. Citado na página 34.

SHEIK, R.; GANTA, S. R.; NIRMALA, S. J. Legal sentence boundary detection using hybrid deep learning and statistical models. *Artificial Intelligence and Law*, Springer, p. 1–31, 2024. Citado 2 vezes nas páginas 30 e 31.

SIAMI-NAMINI, S.; TAVAKOLI, N.; NAMIN, A. S. The performance of lstm and bilstm in forecasting time series. In: IEEE. *2019 IEEE International conference on big data (Big Data)*. [S.l.], 2019. p. 3285–3292. Citado 2 vezes nas páginas 26 e 40.

TAUK, C. S.; SALOMÃO, L. F. Inteligência artificial no judiciário brasileiro. *Diké-Revista Jurídica*, v. 22, n. 23, p. 2–32, 2023. Disponível em: <url(https:/-/periodicos.uesc.br/index.php/dike/article/view/3819/2419)>. Citado na página 15.

TEIXEIRA, J. A.; RÊGO, M. C. B. Inovação no sistema judiciário com a adoção do processo judicial eletrônico em um tribunal de justiça brasileiro. *Revista Ciências Administrativas*, v. 23, n. 3, p. 369–384, 2017. Citado na página 14.

VALE, R. F. do. Análise comparativa de métodos de simplificação de sentenças para sumarização de textos. 2019. Citado na página 16.

VASWANI, A. et al. Attention is all you need. *Advances in neural information processing systems*, v. 30, 2017. Citado na página 22.

VIEIRA, R.; LOPES, L. Processamento de linguagem natural e o tratamento computacional de linguagens científicas. *Em corpora*, p. 183, 2010. Citado na página 16.

VINOTHENI, C.; LAKSHMANAPANDIAN, S. Deep learning-based text segmentation in nlp using fast recurrent neural network with bi-lstm. *Smart Intelligent Computing and Communication Technology*, IOS Press, v. 38, p. 87, 2021. Citado 2 vezes nas páginas 30 e 31.

VINYALS, O.; FORTUNATO, M.; JAITLY, N. Pointer networks. *Advances in neural information processing systems*, v. 28, 2015. Citado 3 vezes nas páginas 26, 27 e 40.

WIRTH, R.; HIPP, J. Crisp-dm: Towards a standard process model for data mining. In: MANCHESTER. *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*. [S.l.], 2000. v. 1, p. 29–39. Citado na página 34.

XIA, J.; WANG, H. A sequence-to-sequence approach with mixed pointers to topic segmentation and segment labeling. In: *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. [S.l.: s.n.], 2023. p. 2683–2693. Citado na página 17.

YU, R. et al. Lstm-efg for wind power forecasting based on sequential correlation features. *Future Generation Computer Systems*, Elsevier, v. 93, p. 33–42, 2019. Citado 2 vezes nas páginas 23 e 24.

YU, Y. et al. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . , v. 31, n. 7, p. 1235–1270, 2019. Citado 2 vezes nas páginas 24 e 25.

ZHONG, M. et al. Pointer networks trained better via evolutionary algorithms. *arXiv preprint arXiv:2312.01150*, 2023. Citado na página 26.