LUAN ALECXANDER KRZYZANIAK

EXTRAÇÃO DE ESQUEMAS EM DOCUMENTOS LEGAIS NÃO ESTRUTURADOS UTILIZANDO LLMS

Trabalho de Conclusão de Curso apresentado ao Curso de Ciência da Computação da Universidade Federal da Fronteira Sul (UFFS), como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Denio Duarte

CHAPECÓ
2025

LUAN ALECXANDER KRZYZANIAK

EXTRAÇÃO DE ESQUEMAS EM DOCUMENTOS LEGAIS NÃO ESTRUTURADOS UTILIZANDO LLMS

Trabalho de Conclusão de Curso apresentado ao Curso de Ciência da Computação da Universidade Federal da Fronteira Sul (UFFS), como requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Denio Duarte

Este Trabalho de Conclusão de Curso foi avaliado e aprovado pela banca avali- adora em: 10/12/2025.

BANCA AVALIADORA

Dr. Denio Duarte - UFFS

Prof. Dr. Geomar Schreiner - UFFS

Prof. Dr. Giancarlo Dondoni Salton - UFFS

SBC Computing Reviews, 202X, XX:1
ISSN: 2966-3938 • doi: 10.5753/rocs.202X.XXXXXX

Revisão da Literatura

Extração de Esquemas em Documentos Legais Não Estruturados Utilizando LLMs

Schema Extraction from Unstructured Legal Documents Using Large Language Models

Luan Alecxander Krzyzaniak [Universidade Federal da Fronteira Sul | *luan.krzyzaniak@estudante.uffs.edu.br*]
Denio Duarte [Universidade Federal da Fronteira Sul | *duarte@uffs.edu.br*]

Universidade Federal da Fronteira Sul, Rodovia SC 484, km 02, Bairro Fronteira Sul, Chapecó, SC CEP 89815-899 Tel. (49) 2049-2600 CNPJ 11.234.780/0007-46, Brasil.

Resumo. Este trabalho explora o uso de Large Language Models (LLMs) na extração de esquemas a partir de documentos legais não estruturados. O estudo propõe uma abordagem que integra técnicas de coleta, pré-processamento textual, engenharia de prompt e aplicação de *embedding* para estudar a aplicação de LLMs na identificação e organização eficiente de informações jurídicas. Para avaliação, foi utilizado o modelo Mistral 7B Instruct v0.2 em 471 Atas de Registro de Preços (ARPs), segmentadas em blocos de 2.000 tokens. Os resultados da análise por documento indicam alto desempenho na classificação de tipos de dados (*Type Accuracy*: 0,946; *Type Precision*: 0,972), mas desempenho semântico moderado (*Semantic Accuracy*: 0,412; *Semantic Coverage*: 0,714), revelando consistência na tipagem, porém limitações na correspondência semântica. Uma segunda etapa avaliou um JSON unificado construído a partir de todos os esquemas extraídos, no qual as métricas semânticas atingiram valores máximos, mostrando que o modelo recupera todos os campos esperados ao menos uma vez, embora com aumento de ruído. Os achados sugerem que LLMs podem atuar como ferramentas auxiliares na extração de esquemas jurídicos, reduzindo esforços manuais, mas ainda dependem de estratégias adicionais de verificação semântica e consolidação estrutural para aplicação prática.

Abstract. This work explores the use of Large Language Models (LLMs) for schema extraction from unstructured legal documents. The study proposes an approach that integrates data collection techniques, text preprocessing, prompt engineering, and embedding-based evaluation to examine the applicability of LLMs in identifying and organizing legal information efficiently. For evaluation, the Mistral 7B Instruct v0.2 model was applied to 471 Price Registration Records (ARPs), segmented into 2,000-token blocks. Results from the per-document analysis indicate strong performance in data type classification (Type Accuracy: 0.946; Type Precision: 0.972), but moderate semantic performance (Semantic Accuracy: 0.412; Semantic Coverage: 0.714), revealing consistent typing but limitations in semantic correspondence. A second evaluation stage examined a unified JSON constructed from all extracted schemas, in which semantic metrics reached maximum values, showing that the model successfully recovers all expected fields at least once, although with increased noise. The findings suggest that LLMs can serve as auxiliary tools for legal schema extraction, reducing manual effort, but still require additional semantic verification and structural consolidation strategies for practical deployment.

Palavras-chave: LLMs, Esquemas, JSON, Documentos Não Estruturados, Documentos Legais, Embedding, Mistral

Keywords: Large Language Models, Schemas, JSON, Unstructured Documents, Legal Documents, Embedding, Mistral

Recebido/Received: DD Month YYYY • **Aceito/Accepted:** DD Month YYYY • **Publicado/Published:** DD Month YYYY

1 Introdução

O sistema jurídico brasileiro gera diariamente um volume massivo de documentos legais, como petições, leis e contratos, que são armazenados em formatos predominantemente não estruturados [Aquino *et al*., 2024]. A análise desses documentos requer mão de obra significativa, tanto para a identificação de informações específicas quanto para a organização dos dados, o que frequentemente resulta em atrasos na tramitação e conclusão de processos judiciais [Aquino *et al*., 2024]. Embora técnicas de NLP já tenham sido empregadas em tarefas similares, ainda existem limitações quando se trata de lidar com documentos complexos, despadronizados e fortemente contextuais [Dagdelen *et al*., 2024] como estes. Assim, a automatização desses processos se mostra um desafio aberto e de importante resolução.

Por consequência, este estudo têm relevância ao buscar possibilitar a automatização destas tarefas no âmbito jurídico. Esta necessidade é reforçada especialmente pelo avanço que o desenvolvimento de técnicas capazes de processar e estruturar informações jurídicas pode representar na otimização do trabalho de profissionais da área e na redução do congestionamento de processos [Aquino *et al*., 2024]. A automatização de extração de esquemas, em particular, apresenta alto potencial de impacto por possibilitar a organização de informações de forma escalável e consistente [Dagdelen *et al*., 2024]. Portanto, o uso de *Large Language Models* (LLMs), que possuem grande capacidade de análise contextual, se mostra uma alternativa promissora para tarefas de *Natural Language Processing* (NLP) e, por consequência, para a tarefa da extração de esquemas em documentos legais não estruturados.

Portanto, este trabalho tem como objetivo principal avaliar o desempenho de LLMs para a extração de esquemas a partir de documentos não estruturados, com ênfase em docu-

mentos legais, elaborando uma *pipeline* eficaz capaz de converter um conjunto desses documentos em um esquema útil. Busca-se, com isso, reduzir o tempo de análise e organização de informações, aumentar a eficiência em processos jurídicos e demonstrar a eficácia da aplicação de LLMs em tarefas de processamento de linguagem natural voltadas a textos não estruturados.

Para este fim, o trabalho foi iniciado pela a seleção de um modelo base de LLM, fundamentada tanto no seu uso no artigo dos autores Zhang and Soh [2024] quanto em testes pessoais voltados a tarefas de extração. Em seguida, foi construido um dataset de avaliação, composto por documentos legais obtidos por meio de *web scrapping* em portais oficiais do governo e em repositórios de modelos jurídicos. Então, este conjunto de dados foi processado por uma *pipeline* de pré-tratamento textual que inclui filtragem, limpeza e normalização. Após essa etapa, foi elaborado um prompt mínimo e funcional para a LLM, juntamente a um *golden schema* que é utilizado como base comparativa para medir o desempenho da extração nos testes. Por fim, foram realizadas as avaliações do modelo e a coleta das métricas estruturais, calculadas com a ajuda de técnicas de *embedding*.

Os testes conduzidos com uma coleção de Atas de Registros de Preços (ARPs) utilizando o modelo Mistral 7B Instruct v0.2 demonstraram que a abordagem proposta apresenta desempenho satisfatório na extração de esquemas. De modo geral, o modelo foi capaz de identificar a maior parte dos campos relevantes e manter a consistência na atribuição de tipos de dados, alcançando índices elevados de acurácia e precisão tipológica. Embora as métricas de similaridade semântica tenham evidenciado variação significativa entre os documentos, a cobertura média obtida indica que a estratégia adotada ainda é capaz de recuperar, de forma satisfatória, uma parcela substancial das informações do *golden schema*.

O restante do trabalho está organizado da seguinte forma: a próxima seção apresenta os trabalhos que serviram como base de pesquisa para sua realização. Em sequência, a Seção 3 introduz os temas centrais à compreensão do trabalho. A Seção 4 estabelece a metodologia adotada para a realização do estudo, e a Seção 5 apresenta os resultados obtidos. Na seção 6 os resultados são analisados e discutidos, e direções de pesquisa para possíveis trabalhos futuros são apontadas.

2 Trabalhos Relacionados

Nesta seção serão apresentados trabalhos cujo tema, ferramentas, técnicas ou conceitos foram aplicáveis ou relevantes para a realização deste trabalho. Os artigos englobam temas relacionados a textos não estruturados, técnicas de extração de esquemas, aplicações de técnicas de melhoramento de LLMs como *fine-tuning* e *Retrieval-Augmented Generation* (RAG), além de estudos a respeito de documentos legais.

O trabalho de Dagdelen *et al*. [2024] trata da automação da extração de informações em textos científicos com foco em compostos químicos e seus atributos. O maior interesse é expandir sobre as técnicas de processamento de NLP de reconhecimento de entidades nomeadas (NER) e extração de relações (RE) que podem ser aplicadas para esta tarefa, mas que não apresentam resultados satisfatórios por si só. A utilização de LLMs como GPT-3 e Llama-2 também é explorada, mas apresenta resultados insatisfatórios devido ao menor treinamento em campos científicos menos populares, segundo o autor, resultado da baixa quantidade de conteúdo computacionalmente acessível disponível sobre estes temas.

Para resolver o problema, os autores propõem a integração da extração da técnica NER com relações de entidade (apelidado NERRE) ao uso de LLMs, nomeado LLM-NERRE (Figura 1). Nesse processo, ambas as extrações de entidades e relações são feitas com o auxílio em tempo real de uma LLM, ao contrário de métodos antigos onde o seu uso era realizado em uma etapa posterior. Quanto à implementação da LLM, os autores utilizaram documentos categorizados e tratados por profissionais como base de dados para realizar um treinamento *fine-tuning* nos modelos GPT-3 e Llama-2. O *fine-tuning* é realizado pelos autores com o objetivo de melhorar a capacidade da LLM de identificar os compostos e suas relações.

Após a aplicação do LLM-NERRE, os resultados são exportados como objetos JSON ou texto corrido, como exemplificado na Figura 1. Aqui, o autor aplica a técnica em um trecho que descreve a performance de uma bateria de lítio. O primeiro passo é a identificação de entidades nomeadas, representadas com cores pelos autores Dagdelen *et al*. [2024]. Em seguida, o texto e as entidades são entregues à LLM, que identifica suas relações e por fim elabora um documento JSON estruturado. A entidade *lithium titanate*, em azul, é identificada como o nome de um composto, e é atribuída a ela a fórmula *Li4Ti5O12*, em rosa. Suas aplicações também são identificadas, como *Li-ion battery* e *positive electrode*, representadas em amarelo.

O autor utilizou a técnica com os modelos GPT-3 e Llama-2 para realizar testes em três tarefas de extração de compostos químicos. As métricas de precisão para qualquer uma das três atividades atingiram 0.87 no pior caso, resultados satisfatórios que demonstram a usabilidade do LLM-NERRE, segundo o autor.

Já no trabalho dos autores Zhang and Soh [2024], é proposta uma solução para automatizar a criação de *knowledge graphs* para grandes volumes de textos sem esquemas estabelecidos (isto é, dados não estruturados) utilizando LLMs. Para solucionar esse problema, é proposto um *framework* que implementa a LLM em uma *pipeline* de três passos nomeada *Extract-Define-Canonicalize* ou *EDC*.

O primeiro passo é o uso da LLM para a extração de *triplets* do texto bruto. Essas estruturas representam uma relação entre dois objetos textuais, tal qual na fase 1 apresentada na Figura 2, em que o nome `Alan Shepard` e o nome `Apollo 14` são relacionados pelo termo `participatedIn`. Em seguida, a LLM é utilizada para analisar os termos relacionais dos *triplets* e buscar definições atributivas mais detalhadas. Na Figura 2, o termo `participatedIn` é descrito pela LLM como a participação da entidade em uma missão especificada pela entidade objeto. Por fim, os *triplets* são revisitados pela LLM que, baseando-se na definição atribuída na etapa 2, altera os termos relacionais de acordo com as definições atributivas. No caso da Figura 2 [Zhang and Soh, 2024], o termo *participatedIn* é substituído pelo termo *mission* na fase 3 seguindo a definição elaborada na fase 2.

O *framework* foi aplicado em três *datasets* distintos e,

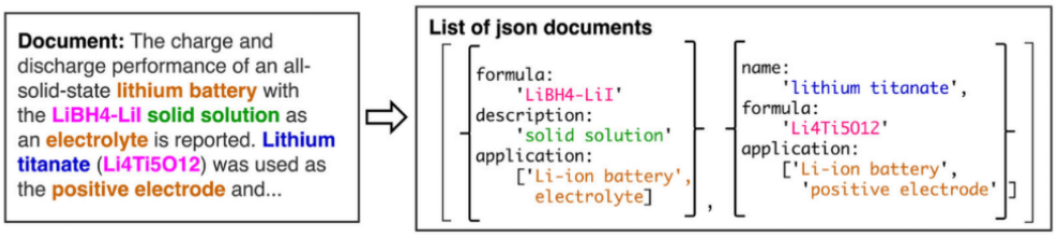


Figura 1. Aplicação do NERRE em um enxerto de documento científico. [Dagdelen *et al.*, 2024]

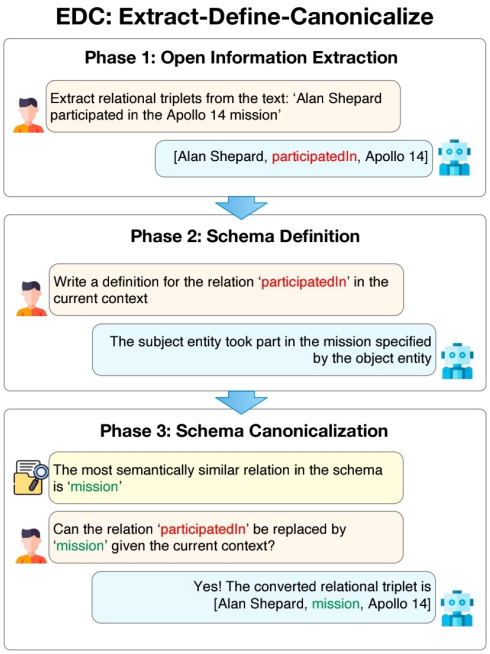


Figura 2. Exemplo de aplicação do framework *Extract-Describe-Canonicalize*.

segundo o autor, demonstrou resultados superiores ou em par com técnicas modernas. Os testes foram realizados nas LLMs GPT-4, Mistral-7b e GPT-3.5-Turbo, sendo a aplicação com GPT-4 aquela com melhores resultados.

A aplicação do LLM-NERRE, portanto, pode se mostrar útil pela capacidade de identificação e análise de entidades e relações. Sua principal aplicação a este tema seria na extração de pessoas físicas e jurídicas que tomam parte em documentos legais, além de seus cargos e funções.

Já no trabalho de Lewis *et al*. [2021], os autores comentam que as LLMs são ferramentas poderosas na interpretação de temas de conhecimento geral e bem difundido mas que, por outro lado, o seu desempenho ao responder perguntas em temas que exigem uma análise extensa e complexa, como áreas de especialização médica, é insatisfatório. Dessa maneira, seu trabalho busca ampliar a capacidade das LLMs em lidar com conhecimento fortemente especializado por meio da implementação do método conhecido como *Retrieval-Augmented Generation* (ou RAG).

Os autores Lewis *et al*. [2021] exploram a implementação do RAG de duas formas: o RAG-*sequence*, que realiza a pesquisa externa uma vez para cada sequência completa de *tokens*, e o RAG-*token* realiza uma pesquisa única para cada *token* analisado.

Neste artigo, os modelos foram testados por uma bateria de perguntas classificadas como *knowledge-intensive*, ou seja, que exigem a análise de grande quantidade de informações para serem respondidas. Segundo os autores, os resultados de ambas as arquiteturas RAG nestes testes superaram tanto modelos que utilizam apenas LLMs quanto implementações baseadas em recuperar e extrair, demonstrando sua aplicabilidade.

Em sequência, o trabalho de Aquino *et al*. [2024] apresenta o uso do RAG em documentos legais brasileiros. Segundo os autores, o sistema jurídico brasileiro produz uma enorme quantidade de documentos legais regularmente e, por isso, procurar, coletar e utilizar os dados presentes nestes documentos é um trabalho que demanda muita mão de obra e planejamento. Portanto, os autores exploram uma alternativa para a extração de dados específicos de documentos legais utilizando LLMs auxiliadas pelo método `RAG`.

A proposta de uso do RAG pelos autores se deve à dificuldade apresentada pelas LLMs em lidar com tarefas de conhecimento amplo ou de contextualização complexa, além da sua tendência a produzir alucinações. Dessa maneira, é proposto que o uso do RAG auxilie na precisão do modelo ao lidar com tarefas de extração. Por outro lado, a implementação efetiva dessa técnica exige um difícil e complexo ajuste de parâmetros, o que dificulta o seu uso. Para resolver esse problema, o projeto introduz uma *pipeline* LLM-RAG que realiza testes de parâmetros na etapa anterior à geração auxiliada por RAG, a fim de encontrar valores satisfatórios para a extração de cada tipo de dado.

A extração foi realizada em dois conjuntos de dados específicos propostos, e os resultados atingidos se mostraram satisfatórios, segundo os autores. A *pipeline* demonstrou um resultado de 90% de acurácia, superando modelos que utilizam expressões regulares, que atingiram média de 58,75% de acerto.

A abordagem aqui proposta se assemelha aos trabalhos apresentados, porém com o foco na extração de dados específicos em documentos legais brasileiros. As implementações de técnicas como RAG, NERRE e EDC se mostraram relevantes por oferecerem abordagens eficientes para a extração de dados não estruturados. O trabalho de Lewis *et al*. [2021], especialmente, mostrou-se útil pela sua eficiência em lidar com documentos que exigem alto nível de conhecimento técnico, característica presente nos documentos legais.

Em comum, os trabalhos demonstraram que a aplicação de LLMs sem adaptação ou integração com métodos auxiliares tende a produzir resultados limitados. Por outro lado, as suas aplicações com as mais diversas técnicas de NLP, grafos de conhecimento ou métodos de recuperação expandiram as suas capacidades, apresentando-se como estratégias promissoras para este trabalho.

3 Referencial Teórico

Esta seção apresenta, brevemente, os conceitos necessários para a compreensão da proposta destes trabalho. Inicialmente, é apresentado o conceito de documentos legais. Em seguida, alguns conceitos de extração dos esquemas são discutidos. A seção é finalizada apresentando a arquitetura *transformer* e LLM, bem como a noção de *embeddings*.

3.1 Documentos legais

Documentos legais englobam todos os documentos utilizados por operadores do direito, como advogados, juízes e tribunais, além de incluir legislação, doutrina e jurisprudência. Englobam-se a Constituição Federal, leis, mandatos, contratos, petições, procurações e inúmeros outros modelos que possuam valor legal [Lima and Flores, 2016].

O enorme volume de procurações públicas, cerca de 200 mil entre 2020 e 2023 e, segundo o autor Aquino *et al*. [2024], justifica a necessidade de padronização de documentos legais. A facilidade na leitura e compreensão agiliza o andamento de processos no judiciário e, além disso, a padronização de formatação das leis reduz a necessidade de interpretações subjetivas, garantindo que sejam "redigidas com clareza, precisão e ordem lógica", conforme determina a Lei Complementar nº 95/1998, que reforça a Constituição Federal de 1988.

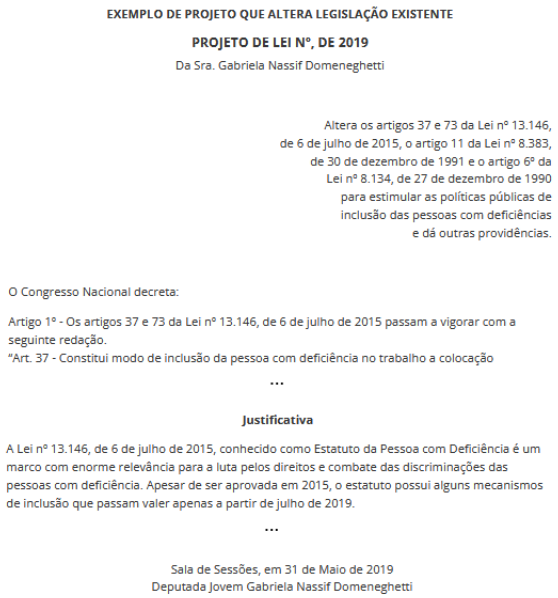
No âmbito digital, a padronização de documentos também é relevante pois facilita os processos de automação, permitindo maior eficiência no tratamento da grande quantia de documentos legais que passam pelo judiciário.

3.1.1 Estrutura

A estrutura de um documento legal depende estritamente do tipo de documento. Leis, contratos, procurações e outros possuem estruturas diversas, com entidades, termos e relações próprias.

A estrutura de uma lei, por exemplo, é rígida: na Figura 3, pode-se observar sua composição. É possível identificar no topo da imagem a frase `Projeto de Lei Nº, de 2019` como sendo o título da lei, seguido da autora `Sra. Gabriela Nassif Domeneghetti`. Logo após a autora, à direita, encontra-se a ementa: um resumo curto dos conteúdos e das pretensões do projeto. Em sequência, temos o corpo textual da lei: contém capítulos, artigos ou incisos de acordo com a proposta do projeto. No caso de projetos de lei, tal qual a figura, contamos também com uma justificativa que busca explicar a necessidade da sua implementação. Por fim, é apresentada a assinatura.

O conhecimento da estrutura de cada documento legal é necessário para a compreensão, análise e identificação de informações importantes e relevantes aos processos da União. Sua importância se estende ao âmbito digital, onde uma estrutura formal e respeitada facilita a extração e tratamento destes textos, uma atividade demorada que pode ser automatizada dado um documento bem estruturado.



EXEMPLO DE PROJETO QUE ALTERA LEGISLAÇÃO EXISTENTE

PROJETO DE LEI Nº, DE 2019

Da Sra. Gabriela Nassif Domeneghetti

Altera os artigos 37 e 73 da Lei nº 13.146, de 6 de julho de 2015, o artigo 11 da Lei nº 8.383, de 30 de dezembro de 1991 e o artigo 6º da Lei nº 8.134, de 27 de dezembro de 1990 para estimular as políticas públicas de inclusão das pessoas com deficiências e dá outras providências.

O Congresso Nacional decreta:

Artigo 1º - Os artigos 37 e 73 da Lei nº 13.146, de 6 de julho de 2015 passam a vigorar com a seguinte redação.

"Art. 37 - Constitui modo de inclusão da pessoa com deficiência no trabalho a colocação

...

Justificativa

A Lei nº 13.146, de 6 de julho de 2015, conhecido como Estatuto da Pessoa com Deficiência é um marco com enorme relevância para a luta pelos direitos e combate das discriminações das pessoas com deficiência. Apesar de ser aprovada em 2015, o estatuto possui alguns mecanismos de inclusão que passam valer apenas a partir de julho de 2019.

...

Sala de Sessões, em 31 de Maio de 2019

Deputada Jovem Gabriela Nassif Domeneghetti

Figura 3. Exemplo de projeto de lei[1]

3.1.2 Armazenamento digital

O início do interesse pela regularização de documentos legais digitais começou ainda em 2001, quando o Decreto n.º 3.865 e a Infraestrutura de Chaves Públicas Brasileira (ICP-Brasil) foram publicados [Lima and Flores, 2016]. Porém, a autenticidade da digitalização e armazenamento de documentos legais, públicos ou privados, só foi regulamentada pela Lei nº 12.682/2012, que também confere aos documentos digitais a equivalência a documentos físicos. Dessa maneira, documentos legais podem ser físicos ou digitais, desde que sua autenticidade seja comprovada por assinatura eletrônica ou certificado digital.

Uma ampla gama de documentos legais são armazenados na internet tanto por órgãos oficiais quanto por organizações jurídicas, sendo espalhados em diversos portais. O JusBrasil[2], por exemplo, é uma plataforma que centraliza decisões judiciais, petições, leis e notícias jurídicas. O Portal da Legislação[3], por outro lado, é o repositório oficial das leis federais brasileiras, como a constituição, códigos civil e penal, e legislações complementares. Aqui muitas das leis são formatadas em HTML ou em textos estruturados de maneira simples, como os utilizados em publicações de notícias, o que facilita a sua extração. Por fim, os portais dos tribunais superiores, como STF, STJ, TST e afins possuem decisões judiciais; enquanto o Portal da Transparência oferece contratos executados e anonimizados.

No que tange a extração de esquemas nestes documentos, a falta de padronização entre órgãos e tipos documentais, com variação na linguagem e terminologias, acaba a transformando em uma tarefa complexa e que demanda muita mão de obra [Aquino *et al*., 2024]. O uso predominante dos PDFs, que apresentam os dados de maneira não estruturada, dificulta a extração de informações relevantes destes documentos. Além disso, uma grande parte dos documentos legais é escaneada e armazenada em imagens JPEG, de forma em que a própria extração textual se torna uma tarefa complexa.

Por outro lado, é importante reforçar que existem nor-

[1]Câmara Legislativa. Disponível em: `https://www2.camara.leg.br/a-camara/programas-institucionais/experiencias-presenciais/parlamentojovem/sou-estudante/material-de-apoio-para-estudantes/modelo-de-projeto-de-lei`

[2]https://www.jusbrasil.com.br

[3]https://www4.planalto.gov.br/legislacao

mativas que auxiliam na padronização, mesmo que parcialmente. Segundo o autor Lima and Flores [2016], a legislação brasileira vem evoluindo para estabelecer diretrizes claras para a digitalização, gestão e preservação de documentos digitais no âmbito federal. A Lei nº 12.682/2012, citada anteriormente, é uma das analisadas pelo autor; e juntamente a decretos como o Decreto nº10.278/2020 buscam assegurar a padronização destes documentos.

A efetividade destas implementações para os fins de extração de esquemas, no entanto, ainda se mostram limitadas. Como observado previamente, mesmo dentre os próprios sites da União a formatação de documentos legais não é fortemente padronizada. Alguns portais lidam somente com PDFs, enquanto outros como o Portal da Transparência permitem exportar seus dados em CSV. Por consequência, a coleta de dados para treinamento demanda um tratamento individual a cada portal e formato de arquivo. A falta de padronização acarreta, portanto, em inúmeros problemas que dificultam a automatização de qualquer tarefa jurídica.

3.2 Esquemas

Esquemas são estruturas formais utilizadas para descrever a organização dos dados, incluindo atributos, tipos e relações [Benson and Grieve, 2021]. Constituem a base da representação de dados moderna: em bancos de dados relacionais, os esquemas definem tabelas, colunas e restrições; em formatos semi-estruturados, como XML ou JSON, descrevem a estrutura esperada dos documentos [Benson and Grieve, 2021]. Assim, os esquemas atuam como metadados que permitem compreender o formato e o conteúdo armazenado.

A Tabela 1 apresenta um extrato de um conjunto de dados representado no modelo relacional. Os rótulos `ID`, `NOME`, `IDADE`, `CIDADE` e `PAÍS` compõem o esquema da tabela, descrevendo como os dados devem ser estruturados e quais tipos devem assumir. Esses metadados também garantem que quaisquer valores inseridos nas colunas respeitem os tipos especificados (como números inteiros, números reais ou cadeias de caracteres), além de atuarem como referência para interpretação dos dados nelas contidos.

Por exemplo, os valores `Maria` e `José`, presentes nas linhas 1 e 2, são associados ao atributo `NOME` e restritos ao tipo *string*, enquanto os valores 45 e 32 se referem ao atributo `IDADE` e seguem o tipo *integer*. Dessa forma, o esquema facilita a leitura e análise da tabela, eliminando a necessidade de rotular individualmente cada entrada.

ID	NOME	IDADE	CIDADE	PAÍS
100	Maria	45	New York	EUA
101	José	32	São Paulo	Brasil
102	Paulo	78	Maringá	Brasil
103	Beto	17	Santos	Brasil
104	Julia	22	Lisboa	Portugal
105	Amanda	25	Paris	França

Tabela 1. Extrato de uma tabela no formato relacional representando um indivíduo.[4]

[4]DIO. Disponível em: `https://www.dio.me/articles/descomplicando-banco-de-dados-e-seus-tipos-1dab874518c0`

3.2.1 Níveis de estruturação de dados

A identificação de esquemas em dados estruturados, como tabelas relacionais, tende a ser direta devido ao formato rigidamente padronizado. Entretanto, essa tarefa se torna mais complexa em dados semi-estruturados e ainda mais desafiadora em dados não estruturados [Christopher *et al*., 2021].

Em dados semi-estruturados, como JSON e XML, os metadados estão embutidos nos próprios valores. Essa característica proporciona flexibilidade, mas também aumenta a variabilidade estrutural. Um mesmo arquivo JSON pode conter objetos completamente distintos, dificultando a inferência automática de um esquema global.

Nos dados não estruturados, a complexidade é ampliada. Esses dados são amplamente predominantes na internet — artigos, contratos, jornais, postagens e documentos administrativos. A extração de informações estruturais nesses cenários requer interpretação contextual, pois os metadados podem aparecer de formas variadas ou mesmo sem rótulo explícito. Além disso, diferentes documentos podem expressar a mesma categoria de informação de maneiras heterogêneas (por exemplo, ocupações como `Representante`, `Estudante` ou `Escritor`).

A Figura 4 ilustra uma cláusula de contrato de compra de imóvel que evidencia a dificuldade de identificar esquemas em texto corrido. Os cômodos do imóvel (`quartos`, `banheiros`, `sala`, `cozinha`), mencionados na linha 4, aparecem imersos em narrativa textual, sem lista ou estrutura explícita que os categorize como atributos do imóvel. Da mesma forma, o endereço (`Rua das Margaridas, nº 120, Bairro Vila Nova, Estado de São Paulo`), distribuído entre as linhas 4 e 5 e introduzido por "*situado à*", não apresenta rótulo formal que facilite sua identificação.

1. **CLÁUSULA PRIMEIRA**
2. O VENDEDOR, na qualidade de legítimo proprietário do imóvel situado à Rua
3. das Margaridas, nº 120, Bairro Vila Nova, Estado de São Paulo, constituído por
4. uma casa térrea com 3 quartos, 2 banheiros, sala, cozinha, área de serviço e
5. garagem, adquirido através de escritura definitiva registrada no 2º Registro de
6. Imóveis desta Capital sob o nº de matrícula 123456, resolve vendê-lo à
7. COMPRADORA, pelo valor de R$ 350.000,00 (trezentos e cinquenta mil reais),
8. que deverá ser pago da seguinte forma:

Figura 4. Exemplo da cláusula de um contrato de compra de uma casa com dados fictícios.

3.2.2 Automatização da extração de esquemas

Diante da predominância de dados não estruturados como método de armazenamento digital, automatizar a identificação de esquemas torna-se crucial tanto para reduzir o tempo e o esforço humano necessários na análise documental, quanto para facilitar a conversão desses dados para um modelo estruturado. Para esses fins, diversas técnicas de NLP têm sido aplicadas para tratar textos e permitir que sistemas computacionais identifiquem padrões estruturais. No entanto, apesar de seu avanço, estas técnicas ainda enfrentam dificuldades em extrair corretamente metadados quando estes aparecem de forma implícita.

O desenvolvimento e a popularização dos LLMs, especialmente após a adoção da arquitetura transformer, apresentaram novas possibilidades na tarefa de extração estrutural. LLMs são capazes de reconhecer padrões textuais comple-

xos, definir categorias implícitas e relacionar informações dispersas pelo documento, oferecendo capacidades semânticas que complementam métodos clássicos de NLP.

Combinadas, as técnicas tradicionais de NLP e as capacidades semânticas das LLMs representam uma abordagem promissora para automatizar a extração de esquemas em documentos não estruturados e ricos em contexto, como contratos e documentos legais. Esses conceitos serão discutidos nas próximas seções.

3.3 *Large Language Models*

As LLMs são modelos de aprendizado de máquina em grande escala capazes de processar linguagem humana e gerar respostas através da predição sequencial de *tokens*. São modelos treinados em enormes datasets com capacidade de gerar textos coerentes, responder perguntas e realizar tarefas complexas de compreensão textual, segundo artigo da empresa Achiam *et al*. [2024].

A popularização das LLMs, possibilitada pelo aumento de poder computacional e arquiteturas novas como o Transformer [Achiam *et al*., 2024], iniciou em 2022 com o lançamento público do GPT, da OpenAI. Nos anos seguintes, diversos modelos com arquiteturas semelhantes foram desenvolvidos, como o LLAMA (da Meta), o DeepSeek (da DeepSeek AI) e o Claude (da Anthropic).

Atualmente os LLMs são popularmente utilizados por meio de interfaces de interações conversacionais, que permitem que os usuários insiram *prompts* que orientam a geração das respostas do modelo. O ChatGPT, por exemplo, é a interface online pelo qual é disponibilizado o acesso aos diferentes modelos GPT [OpenAI, 2024]. De forma semelhante, a LLM chinesa DeepSeek também mantém uma interface de mesmo nome. No entanto, alguns modelos como o LLaMA não possuem hospedagem online oficial: este foi proposto e lançado como um modelo base de código aberto que pode ser utilizado para criar aplicações e interfaces, com foco no uso por pesquisadores e acadêmicos [Touvron *et al*., 2023].

Vale a pena citar que mesmo os principais modelos do mercado ainda diferem em aspectos centrais. Esses produtos diferem em características como o tamanho do *prompt*, *datasets* de treinamento, quantia de parâmetros e especializações. Ao se tratar de treinamento, por exemplo, o GPT-3 foi treinado com aproximadamente 300 bilhões de *tokens*, enquanto o LLaMA foi treinado por 1,4 trilhão de *tokens* - quase quatro vezes mais. De maneira semelhante, a capacidade de *prompt* varia: enquanto o GPT-4 analisa até cerca de 8 mil *tokens*[OpenAI, 2024] e o LLaMA 2 processa até 4 mil [Meta AI, 2023], o Claude Pro aceita até 200 mil *tokens*, segundo documentação oficial [Anthropic, 2024].

3.3.1 Arquitetura

Os LLMs são fundamentalmente modelos de aprendizado profundo (*Deep Learning*), que utilizam redes neurais para o processamento de dados e a detecção de padrões a partir de grandes volumes de dados. As redes neurais são estruturas computacionais formadas por camadas de neurônios interconectados que permitem a troca de informações e o ajuste de parâmetros ao longo do treinamento do modelo. Como consequência, esses modelos possuem capacidade de reconhecimento de padrões e generalização eficientes e robustas [Bai *et al*., 2021].

As redes neurais são compostas por três seções principais: a camada de *input*, as camadas ocultas (ou *hidden layers*) e as camadas de *output*. A camada de *input* é composta pelos dados brutos que serão processados pela rede. Em prompts de texto, cada neurônio da camada será um *token*, enquanto no processamento de imagens a coloração de cada pixel formará os neurônios.

As *hidden layers* são um contribuem para o processamento de dados e o ajuste de parâmetros da rede neural. Aqui, cada neurônio recebe dados das camadas anteriores, aplica parâmetros como pesos e bias, e utiliza funções de ativação para gerar uma saída para a próxima camada. As funções de ativação têm um papel importante: evitar a linearidade. Segundo Rasamoelina *et al*. [2020], a variação que elas trazem aos neurônios permite que a rede represente padrões complexos, evitando que se comporte como uma simples função linear.

E é a partir do constante processamento, ajuste e propagação de dados pelas camadas que o treinamento do modelo acontece: a cada dado processado os neurônios são capazes de ajustar seus parâmetros, como o peso de cada entrada (que representa sua importância na decisão do neurônio) e o bias, um valor constante que desloca a ativação. Dessa maneira, são capazes de se adaptarem aos dados fornecidos, reconhecendo padrões complexos que auxiliam na formulação de respostas.

Por último, os dados chegam à camada de *output*. É aqui que a resposta é gerada: é atribuída a cada possível *token* uma probabilidade de ser utilizado na resposta, e os *tokens* são selecionados sequencialmente. Esta escolha também pode ser controlada através de técnicas de seleção de *tokens*, como o *greedy decoding* - que sempre escolhe o *token* com maior probabilidade; ou o *Top-K Sampling*, que escolhe aleatoriamente entre os K *tokens* mais prováveis. É importante notar que a forma da saída final está diretamente ligada ao tipo de dado de entrada. A rede pode gerar, por exemplo, texto ou imagens de forma sequencial, com uma palavra ou cor de cada vez.

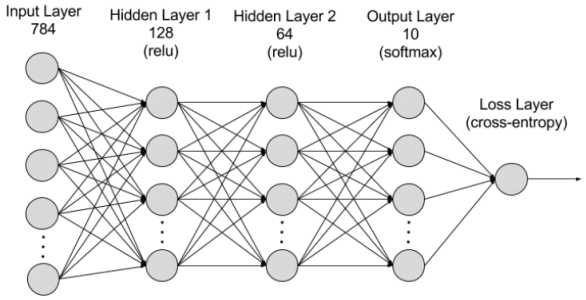


Figura 5. Arquitetura simplificada de uma rede neural.[5]

3.3.2 Treinamento

Apesar dos avanços trazidos pela arquitetura *transformer*, o treinamento de modelos massivos como os LLMs ainda requer grandes volumes de dados e enorme poder computacional para o ajuste de parâmetros [Achiam *et al*., 2024].

[5]AMAZON WEB SERVICES. Disponível em: `https://aws.amazon.com/pt/what-is/neural-network/`

O GPT-3, desenvolvido pela OpenAI, é um exemplo: o modelo possui 175 bilhões de parâmetros [Brown *et al*., 2020]. Os bilhões de parâmetros são constantemente atualizados durante o processo de treinamento que, no caso do GPT-3, contou com aproximadamente 300 bilhões de *tokens* [Achiam *et al*., 2024].

Os *tokens* utilizados pelas grandes empresas para o treinamento de LLMs estão divididos entre inúmeras bases de dados. Os *datasets* são essenciais para o treinamento dos modelos, e sua escolha determina as capacidades e especialidades de cada LLM [Achiam *et al*., 2024].

Os 300 bilhões de tokens utilizados no GPT-3, por exemplo, estão divididos entre enormes bases de dados. A Common Crawl - uma vasta coleção de páginas web coletadas desde 2008, foi utilizada no treinamento juntamente ao WebText, que é composto por textos extraídos de links compartilhados na rede social Reddit. Para dados literários, os conjuntos Books1 e Books2, que são compostos por livros de domínio público e obras licenciadas, foram utilizados. Além destes, grande parte da enciclopédia online Wikipedia foi utilizada de maneira fundamental para a geração de saídas estruturadas.

Modelos como o GPT-3 são treinados em infraestruturas computacionais de alto desempenho, muitas vezes contendo milhares de GPUs e TPUs, com técnicas aplicadas para otimizar o uso dessas máquinas. Por esse motivo, o desenvolvimento de LLMs acaba por ser uma tarefa custosa e, por consequência, limitada a centros de pesquisa, empresas multimilionárias ou universidades com infraestrutura modernas [Touvron *et al*., 2023].

Devido a esta limitação, empresas como a Google através da Google Cloud introduziram o serviço de aluguel de GPUs. Como o treinamento de uma LLM de larga escala é impossível sem uma enorme infraestrutura, esta é uma alternativa (ainda que custosa) para a elaboração por indivíduos, organizações ou pequenas empresas. Além disso, lançamento de modelos open-source como o LLaMA e o Mistral, aliado a técnicas de *fine-tuning*, torna possível a adaptação de modelos já treinados de maneira barata e exponencialmente mais rápida [Touvron *et al*., 2023]

3.4 *Transformer*

O transformer é uma arquitetura de rede neural proposta em [Vaswani *et al*., 2023]. Sua arquitetura é baseada na implementação de mecanismos de atenção, desenvolvido principalmente para tarefas de tradução em sequência e de maneira a ser facilmente paralelizada e escalável, segundo os autores.

A arquitetura transformer é separada entre *encoder* e *decoder*, representados na Figura 6. Enquanto o *encoder* é responsável por processar a entrada e gerar representações contextuais, o *decoder* gera as saídas de acordo com as representações do *encoder*.

O encoder é composto por N camadas idênticas (sendo seis camadas na implementação original) contendo implementações de *Multi-Head Attention*, *Feed Forward* e, após cada uma dessas, etapas de *Add & Norm*

O *Multi-Head Attention* é o principal mecanismo de atenção proposto pela arquitetura. Este mecanismo permite que a rede neural preste atenção em múltiplos *tokens* da entrada durante o processamento de cada *token*, isto é, utilize o restante da entrada como contexto para cada análise. Esta atenção múltipla é referenciada como as diversas "cabeças"do mecanismo. Dessa maneira, em cada *token* tratado pelo modelo é realizada uma análise de contexto que o carrega com informações auxiliares que, posteriormente, impactam na resposta gerada. Segundo os autores Manning and Schütze [1999], essa técnica é um avanço em relação às redes neurais RNN, que utilizavam somente o *token* anterior como ponto de referência.

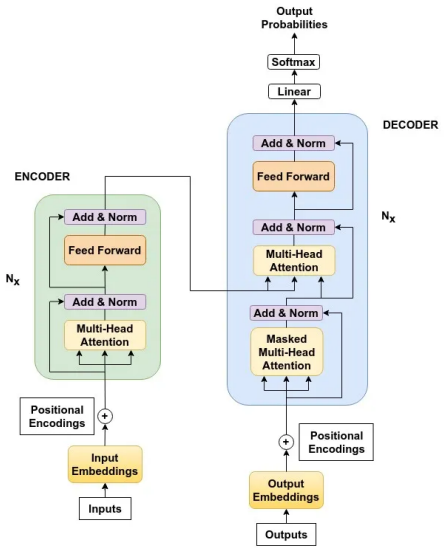


Figura 6. Arquitetura *transformer* apresentada no artigo [Vaswani *et al*., 2023].

Em seguida, cada *token* passa por uma rede neural *Feed Forward*, que é responsável por aplicar uma transformação não-linear através de uma função de ativação, como citado previamente na explicação de redes neurais. Esta etapa não utiliza um sistema de atenção, como o *Multi-Head Attention*, mas aplica seu tratamento *token* a *token* levando em consideração as informações de contexto já inseridas na etapa com o mecanismo. A aplicação da *Feed Forward*, segundo o autor Manning and Schütze [1999], evita a linearidade que limitaria os resultados do *transformer*.

Após a execução de cada um dos mecanismos (tanto *Multi-Head Attention* quanto *Feed Forward*), a arquitetura *transformer* aplica a etapa de *Add & Norm*, ou conexão residual e normalização em camadas. Aqui, a entrada x de um mecanismo é somada à sua saída $f(x)$, gerada após a sua execução. Dessa maneira, o modelo é capaz de aprender através de correções graduais ao permitir que os novos dados mantenham informações importantes já contidas nos dados de entrada. Após a soma, ainda é realizada uma normalização por camadas, onde os valores são ajustados para terem média zero e variação unitária, o que, como descrito por Vaswani *et al*. [2023], estabiliza o treinamento e melhora o desempenho do modelo.

Por fim, a saída do *encoder* é uma sequência de vetores, um para cada *token*, carregados contextualmente, ou seja, contendo informações sobre outros *tokens* juntamente às informações do *token* principal. Os dados vão, então, para o *decoder*, que é responsável pela geração sequencial de *tokens* da resposta.

O *decoder*, assim como o *encoder*, é composto por N camadas idênticas (sendo seis utilizadas na implementação

original). Aqui também estão presentes os mecanismos de *Multi-Head Attention*, *Feed Forward* e as etapas de *Add & Norm*, porém com a adição do *Masked Multi-Head Attention* no início de sua *pipeline*.

O mecanismo de *Masked Multi-Head Attention* é responsável implementar uma máscara triangular, que impede o modelo de levar *tokens* futuros em consideração na geração sequencial da saída. Dessa maneira, cada *token* é gerado utilizando o mecanismo de atenção apenas nos tokens já gerados. Na geração do terceiro *token*, por exemplo, o *Masked Multi-Head Attention* impede o modelo de considerar os possíveis *token* 4 ou 5, e permite apenas a atenção aos *token* 1 e 2, com o objetivo de garantir uma sequência fluida de geração.

3.5 *Embeddings*

Os *embeddings* são representações numéricas de elementos linguísticos, como palavras, sentenças ou documentos inteiros, utilizados para capturar relações semânticas de maneira efetiva. Essa técnica surgiu para lidar com as limitações de métodos tradicionais de representação textual baseados em contagem, como o *bag-of-words* e o *TF-IDF*, que tratavam cada termo de forma independente e não eram capazes de expressar relações de similaridade ou contexto. Com o avanço dos modelos neurais voltados para a linguagem, os *embeddings* passaram a ocupar papel central em sistemas de processamento de texto por permitirem que computadores lidem com significado de forma aproximada à interpretação humana [Chersoni *et al*., 2021].

Segundo Mikolov *et al*. [2013], uma das primeiras contribuições expressivas para essa abordagem foi o Word2Vec, que introduziu representações vetoriais capazes de posicionar palavras semanticamente semelhantes próximas em um espaço multidimensional. Posteriormente, técnicas mais robustas surgiram, como GloVe, FastText e, mais recentemente, modelos baseados em transformadores, que ampliaram a capacidade dos *embeddings* ao incorporarem contexto dinâmico a cada *token* representado.

O avanço dos modelos pré-treinados, como BERT e RoBERTa, contribuiu para o uso dos *embeddings* orientados a sentenças. Diferentemente das versões que representavam apenas palavras isoladas, os modelos recentes conseguem codificar unidades textuais maiores, permitindo medir similaridade entre frases, parágrafos ou documentos completos. Os autores Reimers and Gurevych [2019] destacam que essa evolução permitiu o uso de medidas de distância vetorial como instrumento para tarefas mais complexas, como detecção de paráfrases, recuperação de informação e avaliação semântica de respostas geradas por LLMs.

A função principal dos *embeddings* consiste em transformar textos em vetores em que proximidade espacial representa proximidade de significado. Duas frases distintas, mas semanticamente equivalentes, tendem a ser mapeadas para regiões próximas no espaço vetorial, enquanto textos sem relação ficam afastados. Essa característica se tornou fundamental para este trabalho, pois os resultados gerados pela LLM frequentemente apresentam variações lexicais ou estruturais que não podem ser avaliadas por métodos puramente textuais. A utilização de *embeddings* possibilita comparar elementos do *golden schema* com os campos produzidos pelo modelo de forma robusta, considerando não apenas igualdade literal, mas grau de similaridade conceitual.

Pertinente a este trabalho, o uso de representações vetoriais permite o cálculo de métricas semânticas através de técnicas como a similaridade do cosseno, amplamente empregada em tarefas de comparação textual. Essa abordagem permite quantificar a proximidade entre dois textos mesmo quando expressos de maneiras diferentes, como discutido anteriormente, uma característica essencial na avaliação de saídas de modelos de linguagem que lidam com textos não estuturados. Estudos recentes reforçam esse potencial, como o de Gao *et al*. [2024], que investiga como *embeddings* extraídos de LLMs podem ser usados para classificação, interpretação e comparação semântica, demonstrando que tais vetores capturam relações de significado de forma eficaz.

Apesar das vantagens, o método apresenta limitações. Expressões numéricas, códigos formais e sequências não textuais podem não ser representadas adequadamente no espaço vetorial, gerando incerteza nestes casos. Ainda assim, as vantagens em relação ao tratamento textual por não depender de similariedade sintática tornam essa técnica especialmente valiosa para avaliar a coerência semântica das extrações e identificar divergências relevantes entre o esquema extraído pela LLM e o *golden schema* esperado.

4 Metodologia

A metodologia adotada neste trabalho inicia com as etapas de coleta, pré-processamento e preparação do conjunto de dados, além da escolha e preparação do modelo, e por último segue para detalhar os testes e uso de *embeddings* para a coleta dos resultados. As seções a seguir detalham estas etapas.

4.1 Elaboração do Dataset

A construção do conjunto de dados teve início com a coleta de documentos legais disponibilizados em portais oficiais do governo, particularmente Atas de Registro de Preço (ARP) do Portal de Transparência[6] e documentos relacionados, normalmente publicados em formato PDF. As ARPs possuem esquemas semelhantes, mas diferem em estrutura e representação de dados, o que as torna adequadas para o estudo da tarefa de extração de esquemas e, sobretudo, permitem comparação via *embedding* utilizando um *golden schema* como base.

A pesquisa e coleta inicial ocorreu de forma manual, com o objetivo de compreender a estrutura e a distribuição dos documentos nos diferentes portais. Após essa etapa exploratória, a automação foi implementada para ampliar significativamente o volume de dados e garantir maior controle sobre os critérios de seleção. Para isso, foi utilizado o framework Selenium[7], escolhido por sua capacidade de simular interações humanas com navegadores e lidar com páginas que dependem fortemente de JavaScript ou carregamento dinâmico de conteúdo.

Com o Selenium, foi construída uma rotina capaz de navegar por listas de publicações, interagir com elementos da página, percorrer paginações e realizar o download direto dos arquivos PDF. Essa abordagem permitiu contornar limi-

[6]https://contratos.sistema.gov.br/transparencia
[7]https://www.selenium.dev/

tações comuns de raspagem tradicional, como páginas que bloqueiam requisições diretas ou que exigem a execução de scripts *client-side* para exibir documentos. Além disso, a automação possibilitou estabelecer regras de priorização, como baixar apenas arquivos compatíveis com os objetivos da pesquisa. O resultado dessa etapa foi um conjunto de documentos legais composto por 560 arquivos, reunidos de forma sistemática e reprodutível.

Após concluída a coleta automatizada, os documentos passaram por um processo de pré-seleção para garantir sua usabilidade. Arquivos duplicados ou compostos majoritariamente por imagens digitalizadas foram excluídos, uma vez que não estão contidos no escopo do trabalho, resultando em um conjunto de 471 documentos. Os arquivos resultantes passaram posteriormente por etapas de limpeza, normalização e preparação textual, detalhadas na próxima seção.

4.2 Pré-Tratamento Textual

O pré-tratamento textual buscou tratar a heterogeneidade dos documentos coletados via web scraping. As Atas de Registro de Preços (ARPs) apresentam formatos variados, problemas de digitalização, padrões inconsistentes e textos com ruídos típicos de PDFs governamentais. Para garantir uma qualidade uniforme nos dados enviados ao modelo de linguagem, foi utilizada uma *pipeline* que limpa, padroniza e organiza o texto sem perda de dados relevantes, implementada por meio de um script elaborado na linaguagem Python.

O processo inicia com o carregamento e filtragem dos PDFs. Arquivos duplicados, corrompidos ou vazios são removidos, assim como documentos compostos apenas por imagens, já que o estudo não utiliza *Optical Character Recognition (OCR)*. Esse filtro evita arquivos que não poderiam ser processados e mantém o *dataset* consistente.

Após a filtragem, o texto de cada página é extraído e submetido a uma limpeza que remove quebras de linha excessivas, espaçamentos irregulares e caracteres residuais produzidos por conversões internas do PDF. Em seguida, o conteúdo passa por normalizações textuais. Datas são padronizadas para reduzir variações entre formatos distintos, facilitando o reconhecimento posterior pela LLM. Abreviações comuns, como “v. total” ou “obs.”, são substituídas por suas formas completas (”valor total”, ”observação”) para tornar o texto mais claro e reforçar consistência.

Uma parte importante do processo foi a tentativa de normalização de valores monetários. Inicialmente, buscou-se padronizar preços e valores financeiros, já que são elementos centrais nas ARPs. No entanto, essa abordagem mostrou-se inviável. Muitos números presentes nos documentos possuem o padrão de escrita similares a valores monetários, como CPFs, CNPJs, telefones, códigos de itens e percentuais. A ausência frequente do prefixo “R$” impediu diferenciar, de forma segura, valores financeiros de outras entidades numéricas. Mesmo tentando restringir a normalização pelo contexto, os documentos apresentaram variações demais para uma detecção confiável. Para evitar modificações incorretas em dados sensíveis, especialmente identificadores pessoais e referências administrativas, esta etapa do processamento foi abandonada e estes valores foram preservados em suas formas originais.

Depois da limpeza e normalização, o texto das páginas foi reunido em um único bloco contínuo, garantindo que a LLM tenha acesso sequencial completo ao conteúdo do documento. Com essa pipeline, foi possível padronizar os documentos sem comprometer informações críticas, reduzindo ruídos e mantendo a consistência necessária para a análise e para a avaliação do desempenho da LLM.

4.3 Escolha do Modelo de Linguagem (LLM)

A escolha do modelo de linguagem utilizado neste trabalho recaiu sobre o *Mistral 7B Instruct v0.2*, um modelo de código aberto projetado para ser eficiente em tarefas de *finetuning* [Jiang *et al*., 2023]. Durante testes isolados, como no exemplo apresentado na Figura 7, o modelo apresentou desempenho satisfatório na identificação e extração dos esquemas propostos no *dataset* elaborado, superando outras alternativas testadas, como o T5-base da Google.

```
{
  "document": {
    "type": "document",
    "registration_number": {
      "type": "integer",
      "number": "001.2025"
    },
    "date": {
      "type": "date",
      "year": "2025",
      "month": "abril"
```

Figura 7. Testes prévios utilizando o Mistral 7B

A escolha do modelo foi fundamentada em referência literária e experimentos preliminares que avaliaram a capacidade das LLMs em extrair estruturas complexas a partir de documentos legais estruturados e semiestruturados. Nessas avaliações, o Mistral 7B apresentou um comportamento particularmente consistente, produzindo saídas organizadas e com poucos desvios estruturais. Esse desempenho sugere uma capacidade inerente do modelo em reconhecer padrões hierárquicos e representações esquemáticas desses textos, o que o torna adequado para a tarefa proposta [Wang *et al*., 2025].

A decisão também se estende ao tokenizador utilizado. O Mistral 7B emprega um tokenizador baseado em *SentencePiece* com *Byte-Pair Encoding* (BPE). Esse tipo de tokenização, semelhante à adotada pela família de modelos LLaMA, opera no nível de subpalavras, permitindo que o modelo represente eficientemente tanto termos jurídicos específicos quanto estruturas numéricas e padrões textuais. Essa propriedade contribui para a melhor interpretação e segmentação dos documentos legais presentes no dataset, reduzindo ambiguidades e auxiliando na eficiência da etapa de extração de campos. O uso do tokenizador nativo foi particularmente interessante devido à diversidade de terminológica encontrada nos documentos legais, e favoreceu a interpretação contextual.

Por fim, a escolha do Mistral 7B Instruct v0.2 também se justifica por sua boa relação entre desempenho e custo computacional [Jiang *et al*., 2023]. Diferentemente de mo-

delos de maior porte, como o Llama 3 70B ou o GPT-4, o Mistral 7B oferece boa precisão com requisitos computacionais mais modestos, tornando viável sua utilização no contexto desta pesquisa, que opera sob limitações de hardware que impedem a utilização de modelos computacionalmente custosos.

4.4 Preparação do modelo

4.4.1 Golden Schema

Para possibilitar a avaliação do modelo na extração de informações estruturadas de documentos legais, foi necessário definir previamente um *golden schema* e um *prompt* padronizado. O *golden schema* consiste em uma estrutura JSON que identifica os campos de interesse presentes nas Atas de Registro de Preços (ARPs) e define os tipos de dados esperados para cada campo. Sua elaboração foi realizada manualmente, com base na análise de quinze documentos representativos coletados da mesma fonte do dataset, permitindo identificar informações recorrentes e relevantes, como número da ata, processo administrativo, datas, órgão gerenciador, fornecedores e itens registrados. Campos complexos, como listas de fornecedores e itens, foram modelados como listas de objetos, enquanto informações compostas, como dados do órgão gerenciador ou períodos de vigência, foram representadas como objetos aninhados. Como resultado, o *golden schema* ficou composto por 25 chaves. A construção do *golden schema* permitiu criar uma referência consistente para avaliação do modelo, fornecendo tanto a identificação dos campos quanto os tipos de dados esperados, essenciais para o cálculo de métricas como *Type Accuracy*, *Type Precision* e similaridade semântica. O Listing 1 apresenta uma versão reduzida do *golden schema* utilizado neste trabalho, destacando apenas os diferentes tipos estruturais presentes no conjunto completo. O exemplo inclui: chaves simples, representando campos simples como `string`; objetos, que unem propriedades internas organizadas em uma hierarquia; e listas (*arrays*), utilizadas para representar coleções de elementos estruturados, como conjuntos de fornecedores ou itens registrados.

Listing 1: Exemplo simplificado do *golden schema*

```
{
  "numero_ata": {"type": "string"},
  ...
  "vigencia": {
    "type": "object",
    "properties": {
      "data_inicio": { "type": "date" },
      "data_fim": { "type": "date" }}
  },
  ...
  "fornecedores_registrados": {
    "type": "array",
    "items": {
      "type": "object",
      "properties": {
        "razao_social": {"type": "string"},
        "cnpj": { "type": "string" }}}}
  ...
}
```

4.4.2 Prompt

O prompt de extração foi desenvolvido para guiar a LLM na identificação e estruturação das informações mais relevantes presentes nos documentos, respeitando a forma definida pelo *golden schema*. Ele contém instruções a respeito da identificação de campos recorrentes, a padronização de tipos de dados e a representação de listas e objetos aninhados, além de orientar a LLM a não preencher valores e a retornar exclusivamente o esquema JSON. A elaboração do *prompt* focou em garantir consistência nos resultados produzidos pelo modelo, tornando possível a comparação direta com o *golden schema* e permitindo uma avaliação objetiva da capacidade da LLM em extrair informações estruturadas de documentos legais complexos.

Listing 2: Prompt utilizado nos testes

```
Analise o texto abaixo e extraia um esquema JSON que descreva os dados mais importantes encontrados no documento.
O esquema deve seguir o formato chave-valor, com o nome do campo e o tipo de dado (inteiro, string, lista, etc.).
Regras:
- Identifique apenas informações relevantes e recorrentes.
- Use tipos de dado genéricos: "string", "integer", "float", "date", "boolean", "percent".
- Não preencha valores, apenas defina o tipo esperado.
- Se o texto sugerir listas ou tabelas, represente-as com "array" e especifique o tipo dos elementos.
- Use o formato JSON Schema (com "type" e "properties" quando aplicável).
- Não repita este prompt nem adicione comentários, explicações ou texto além do JSON.
- Retorne apenas o JSON puro e válido.

Exemplo de estruturas esperadas:
{ "documento": { "type": "string" },
  "fornecedor": {
    "type": "object",
    "properties": {
      "nome": { "type": "string" },
      "valor": { "type": "integer" }}},
  "itens": {
    "type": "array",
    "items": {
      "type": "object",
      "properties": {
        "descricao": { "type": "string" },
        "quantia": { "type": "integer" }}}}
}
```

4.5 Testes

A etapa de testes teve como objetivo avaliar o desempenho do modelo Mistral 7B Instruct v0.2 na extração de esquemas a partir dos documentos legais previamente coletados e submetidos ao processo de pré-tratamento textual. Conforme descrito nas seções anteriores, o conjunto de dados utilizado foi composto por Atas de Registro de Preços (ARPs), obtidas

por meio de *web scraping* e filtradas para garantir que apenas documentos únicos e textualmente legíveis fossem incluídos, resultando em uma base de dados composta por 471 documentos.

Para viabilizar a execução da LLM em ambiente local, os testes foram realizados em uma máquina equipada com uma GPU NVIDIA RTX 4070 Ti, com 8 GB de memória dedicada. Devido ao tamanho do modelo e às limitações de hardware, foi empregada quantização em 4 bits durante o carregamento da LLM, permitindo reduzir o custo de memória do modelo. Essa técnica tornou possível o processamento dos documentos em tempo hábil e dentro da capacidade computacional disponível.

O Mistral 7B possui um limite de contexto aproximado de 8.000 *tokens*, o que inviabiliza o envio de documentos extensos em uma única chamada ao modelo. Para contornar essa restrição, o conteúdo de cada documento foi segmentado em blocos de 2.000 *tokens*, gerados por meio do tokenizador nativo do próprio Mistral. Essa abordagem garantiu que cada parte do documento fosse processada integralmente, ao mesmo tempo em que evitou truncamentos automáticos decorrentes de alto consumo do contexto da LLM pela junção do *prompt* e do texto do documento, o que poderia comprometer a extração das informações.

Cada *chunk* foi enviada individualmente ao modelo utilizando o mesmo *prompt* de extração previamente definido. O uso de um *prompt* único para todos os blocos buscou assegurar consistência na forma como a LLM interpretou e estruturou as informações ao longo do documento. O resultado do procesamento gera um JSON estruturado para cada bloco, contendo os campos identificados e seus respectivos valores.

Após a execução completa do processo de extração, cada documento teve seu esquema reconstruído a partir dos chunks processados pela LLM, resultando em um único JSON por documento. Esse JSON preserva as chaves e estruturas de todas as chunks, integrando todas as informações extraídas ao longo do documento, garantindo uma representação completa e unificada do esquema produzido pela LLM.

Com os JSONs individuais prontos, foi realizada a primeira etapa de avaliação, na qual cada documento foi comparado diretamente com o golden schema. Nessa fase, foram calculadas métricas de acurácia semântica, cobertura semântica, F1 semântico, acurácia de tipos e precisão de tipos, permitindo analisar, em uma média por documento, o quão próximo o modelo chegou da estrutura esperada.

Além dessa avaliação individual, foi conduzida uma segunda análise baseada na união de todos os JSONs gerados. Para isso, todos os esquemas reconstruídos foram unidos em um único arquivo, agregando as estruturas identificadas em todos os documentos do conjunto. Esse JSON global também foi comparado com o *golden schema*, possibilitando avaliar o desempenho do modelo em nível agregado. A intenção dessa segunda análise foi verificar se a fusão de todos os documentos capturava uma visão mais completa do esquema esperado, superando eventuais lacunas presentes em documentos isolados, ou se, ao contrário, acumulava inconsistências que prejudicavam a fidelidade ao esquema de referência. A Listing 3 apresenta um trecho do esquema resultante.

Listing 3: Exemplo simplificado do esquema unificado gerado pelo modelo

```
{"orgao_gerenciador": {
   "type": "object",
   "properties": {
     "email": { "type": "string" },
     "telefone": { "type": "string" },
     "endereco": {
       "type": "object",
       "properties": {
         "bairro":   { "type": "string" },
         "cep":      { "type": "string" }}
     }}},
  ...
 "fornecedores": {
   "type": "array",
   "items": {
     "type": "object",
     "properties": {
      "registro_de_precos": {
        "type": "object",
        "properties": {
          "cancelado": { "type": "boolean"
              },
          "precos": {
            "type": "array",
            "items": {
              "type": "object",
              "properties": {
                "item": { "type": "string"},
                "preco": { "type": "float"}
               }
              }}}}}
  },
  ...
 "cancelamento": {
   "type": "object",
   "properties": {
     "motivo": { "type": "string" }}}
```

Dessa forma, a avaliação contemplou tanto o desempenho por documento quanto o desempenho consolidado da união de todos eles, permitindo compreender o comportamento da LLM em diferentes granulações e observar como a integração dos resultados influencia a proximidade com o golden schema.

5 Resultados

A análise qualitativa do desempenho do modelo foi realizada comparando os JSONs extraídos com o *golden schema* de referência, utilizando métricas de similaridade semântica e acurácia de tipos. Para isso, aplicou-se um modelo de *embeddings* local (*all-MiniLM-L6-v2*) para calcular similaridades entre pares de campos esperados e extraídos, utilizando similariedade do cosseno. Essa abordagem permitiu identificar correspondências exatas (*semantic_good*) e parciais (*semantic_partial*), além de avaliar a correção dos tipos de dados previstos pelo modelo.

Os resultados gerais obtidos na primeira etapa, que engloba a média resultante da comparação individual do JSON de cada documento, encontram-se resumidos na Tabela 2. A distribuição do seu desempenho nesta tarefa é ilustrada na Figura 8.

Tabela 2. Métricas gerais de desempenho do Mistral 7B Instruct v0.2 na extração de esquemas de ARPs por documento.

Métrica	Valor médio
Semantic Accuracy	0.412
Semantic Coverage	0.714
Semantic F1	0.519
Type Accuracy	0.946
Type Precision	0.972
Campos previstos (média)	30.9

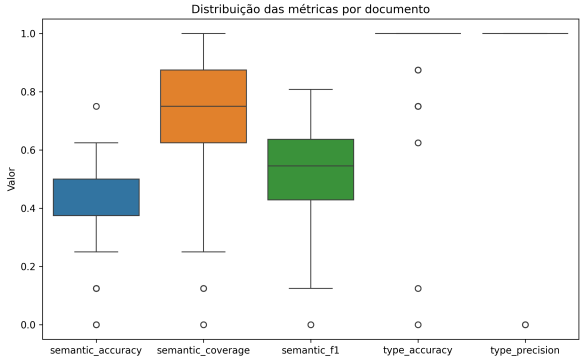


Figura 8. Box Plot da distribuição das métricas de desempenho do Mistral 7B Instruct v0.2 por documento.

A *Semantic Accuracy* de 0.412 indica que aproximadamente 41% dos campos previstos pelo modelo correspondiam semanticamente aos campos do *golden schema*, considerando apenas correspondências com similaridade igual ou superior a 0,75. Já a *Semantic Coverage* de 0.714 revela que cerca de 71% dos campos do esquema de referência foram cobertos, incluindo correspondências parciais com similaridade entre 0,60 e 0,75. O valor intermediário da *Semantic F1* (0.519) evidencia um equilíbrio entre precisão semântica e cobertura.

Em termos de identificação dos tipos de dados, o modelo apresentou desempenho notavelmente elevado. A *Type Accuracy* média de 0,946 indica que 94,6% dos campos extraídos pelo modelo possuíam tipos que correspondiam aos do *golden schema*. A *Type Precision* de 0,972 demonstra que, entre os campos semanticamente bem mapeados (*semantic_good*), 97,2% apresentavam o tipo correto, evidenciando a consistência do modelo na atribuição de tipos.

Adicionalmente, o modelo gerou em média 30,9 campos por documento, valor próximo ao número de campos do *golden schema*, de 25. Isso indica que a LLM conseguiu extrair grande parte das informações relevantes, com valor aceitável de ruído, embora algumas correspondências semânticas não tenham atingido o limiar de similaridade máximo.

O Diagrama de Caixa (*Box Plot*) na Figura 8 fornece uma visão detalhada da variabilidade do desempenho do modelo Mistral 7B Instruct v0.2 em nível de documento.

Como pode ser observado, as métricas de similaridade semântica (*Semantic Accuracy*, *Coverage* e *F1*) demonstram uma alta dispersão entre os documentos, indicada pelas caixas alongadas. Isso sugere que o desempenho do modelo na correspondência semântica é inconsistente ao longo da base de 471 documentos, especialmente no caso da *Semantic Coverage*.

Tabela 3. Métricas gerais de desempenho do Mistral 7B Instruct v0.2 na extração de esquemas de ARPs no JSON unificado.

Métrica	Valor
Semantic Accuracy	0.0120
Semantic Coverage	1.0
Semantic F1	0.0237
Type Accuracy	0.958
Type Precision	0.958
Campos previstos (total)	3310

Por outro lado, as métricas de tipo (*Type Accuracy* e *Type Precision*) apresentam uma dispersão interquartil praticamente nula (caixas esmagadas em 1.0). Este padrão visual indica robustez do modelo na atribuição de tipos de dados. Além disso, a ocorrência de *outliers* mostra-se irrelevante diante do número total de documentos analisados, indicando que o modelo mantém consistência na extração de informações em torno das médias representadas pelo *box plot*.

A partir do comportamento observado na avaliação individual, é possível compreender o contraste nos dados quando se analisa o JSON unificado, formado pela junção de todos os esquemas extraídos ao longo do conjunto de documentos. Nesse cenário, todas as métricas semânticas atingiram valores aceitáveis, como pode ser visto na Tabela 3. Esse resultado mostra que a fusão dos esquemas elimina lacunas e assimetrias presentes em documentos isolados, unindo todas as chaves corretas em um único esquema, permitindo que ao menos uma ocorrência correta de cada campo do *golden schema* seja recuperada pelo modelo.

Apesar da performance semântica elevada, o volume total de campos previstos alcançou 3310, em contraste com os 25 campos existentes no esquema de referência. Esse aumento expressivo revela que a fusão agregou grande quantidade de ruído, incluindo redundâncias, variações nominais de chaves e interpretações alternativas produzidas pelo modelo. Ainda assim, mesmo com o grande número de chaves, o modelo manteve alta coerência na atribuição dos tipos de dados, com *Type Accuracy* e *Type Precision* em torno de 0,96.

De forma geral, os resultados apresentados fornecem uma visão consolidada do desempenho do Mistral 7B Instruct v0.2 na extração de esquemas de documentos não estruturados, servindo como base para as discussões e recomendações que serão abordadas no capítulo seguinte.

6 Conclusão

O presente trabalho avaliou a capacidade do modelo Mistral 7B Instruct v0.2 na extração de esquemas estruturados a partir de documentos legais não estruturados, especificamente Atas de Registro de Preços (ARPs).

Os testes mostraram que a LLM é capaz de identificar e organizar a maior parte das informações relevantes, produzindo JSONs consistentes e com uma tipagem correta entre 94,6% e 97,2% dos campos, dependendo da métrica considerada. A *Type Accuracy* e a *Type Precision*, elevadas em ambas as etapas do teste, apontam a robustez do modelo na classificação dos tipos de campos, mesmo quando o conteúdo dos documentos é complexo ou variado.

Por outro lado, a avaliação semântica, representada pela *Semantic Accuracy* e pela *Semantic Coverage*, demonstrou

limitações na correspondência entre os campos extraídos e o *golden schema* na primeira etapa dos testes, mesmo utilizando técnicas como *embedding*. Isso indica que, embora o modelo seja capaz de estruturar dados de forma coerente, a compreensão semântica completa ainda não é totalmente confiável para automação sem supervisão humana.

Os resultados observados na segunda etapa do teste, com o JSON unificado, ajudam a contextualizar esse cenário ao revelarem dois comportamentos complementares do modelo. Na análise por documento, o desempenho semântico se mostrou variável, evidenciando limitações na consistência da LLM ao lidar com textos complexos e pouco estruturados. Já na avaliação do JSON unificado, os valores máximos das métricas semânticas revelam que, ao longo do corpus, o modelo foi capaz de recuperar corretamente todos os campos esperados ao menos uma vez, embora não de forma uniforme entre os documentos.

Essa diferença ressalta um aspecto importante na avaliação de modelos para extração de esquemas: a fusão de previsões tende a ocultar falhas locais e consolidar apenas os acertos distribuídos ao longo dos documentos. Embora gere um esquema semanticamente completo, a junção também amplia significativamente o ruído e a redundância, o que reforça a necessidade de métodos adicionais de filtragem e normalização de chaves para tornar o resultado utilizável em aplicações reais.

Dessa forma, este estudo conclui que o Mistral 7B Instruct v0.2 pode agilizar a preparação de dados e reduzir parte do esforço manual na extração de esquemas de documentos não estruturados, especialmente no âmbito legal, atuando como ferramenta auxiliar. Entretanto, sua aplicação prática exige estratégias de verificação, limpeza e correção semântica, garantindo que as informações extraídas sejam não apenas estruturadas, mas também semanticamente corretas.

Como direção de trabalhos futuros, pode-se citar: explorar tanto aprimoramentos no modelo, por meio de RAG e *fine-tuning*, quanto o pós-processamento dos esquemas, visando aumentar a correspondência de chaves e reduzir a necessidade de intervenção manual. Essas abordagens têm potencial para tornar a aplicação de LLMs mais confiável na análise de documentos não estruturados complexos, contribuindo para a extração e padronização de dados.

Referências

Achiam, J. *et al*. (2024). GPT-4 technical report.

Anthropic (2024). Context windows – anthropic documentation. Acesso em: 25 jun. 2025.

Aquino, I., dos Santos, M. M., Dorneles, C., and Carvalho, J. T. (2024). Extracting information from brazilian legal documents with retrieval augmented generation. In *Anais Estendidos do XXXIX Simpósio Brasileiro de Bancos de Dados*, pages 280–287, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbbd_estendido.2024.244241.

Bai, X., Wang, X., Liu, X., Liu, Q., Song, J., Sebe, N., and Kim, B. (2021). Explainable deep learning for efficient and robust pattern recognition: A survey of recent developments. *Pattern Recognition*, 120:108102. DOI: https://doi.org/10.1016/j.patcog.2021.108102.

Benson, T. and Grieve, G. (2021). *UML, XML and JSON*, pages 399–426. Springer International Publishing, Cham. DOI: 10.1007/978-3-030-56883-2_21.

Brown, T. B. *et al*. (2020). Language models are few-shot learners.

Chersoni, E., Santus, E., Huang, C.-R., Lenci, A., *et al*. (2021). Decoding word embeddings with brain-based semantic features. *Computational Linguistics*, 47(3):663–698.

Christopher, C., Moore, K., and Liebowitz, D. (2021). Schemadb: Structures in relational datasets. DOI: 10.48550/arXiv.2111.12835.

Dagdelen, J., Dunn, A., Lee, S., Walker, N., Rosen, A. S., Ceder, G., Persson, K. A., and Jain, A. (2024). Structured information extraction from scientific text with large language models. *Nature communications*, 15(1):1418.

Gao, Y., Myers, S., Chen, S., Dligach, D., Miller, T. A., Bitterman, D., Churpek, M., and Afshar, M. (2024). When raw data prevails: Are large language model embeddings effective in numerical data representation for medical machine learning applications? *arXiv preprint arXiv:2408.11854*.

Jiang, A. Q. *et al*. (2023). Mistral 7b.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D. (2021). Retrieval-augmented generation for knowledge-intensive nlp tasks.

Lima, E. d. S. and Flores, D. (2016). A evolução da legislação relacionada à digitalização e aos documentos digitais no âmbito da administração pública federal. *Revista Sociais e Humanas*, 29(1):75–91. DOI: 10.5902/2317175821043.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA.

Meta AI (2023). Llama 2: Open foundation and fine-tuned chat models. Acesso em: 25 jun. 2025.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

OpenAI (2024). Gpt-4 models. Acesso em: 25 jun. 2025.

Rasamoelina, A. D., Adjailia, F., and Sinčák, P. (2020). A review of activation function for artificial neural network. pages 281–286. DOI: 10.1109/SAMI48414.2020.9108717.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics. DOI: 10.18653/v1/D19-1410.

Touvron, H. *et al*. (2023). Llama: Open and efficient foundation language models.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention is all you need.

Wang, D. Y.-B., Shen, Z., Mishra, S. S., Xu, Z., Teng, Y.,

and Ding, H. (2025). Slot: Structuring the output of large language models.

Zhang, B. and Soh, H. (2024). Extract, define, canonicalize: An llm-based framework for knowledge graph construction.