

Contrato360: uma aplicação de perguntas e respostas usando modelos de linguagem, documentos e bancos de dados

Antony Seabra de Medeiros^{1,2}, Claudio Cavalcante^{1,2}, João Nepomuceno¹,
Lucas Lago¹, Nicolaas Ruberg¹ and Sérgio Lifschitz² *

¹ BNDES - Área de Tecnologia da Informação, Rio de Janeiro, RJ

²PUC-Rio - Departamento de Informática, Rio de Janeiro, RJ

{amede, cfrag, jonep, lulag, nic}@bndes.gov.br, sergio@inf.puc-rio.br

Resumo. Apresentamos uma aplicação de perguntas e respostas (Q&A) para o processo de gestão de contratos. Conjugando as informações capturadas de contratos (PDF) e dados provenientes do sistema de apoio, obtivemos informações que são enviadas para o GPT-4. A relevância das respostas é melhorada pelas técnicas de Recuperação Aumentada (RAG) e text-to-SQL, dispensando o retreinamento do modelo de linguagem. Exploramos também a Engenharia de Prompt para melhor o foco das respostas. Ao longo do trabalho, observamos que tais técnicas conjugadas aumentam a relevância das respostas. Destacamos o potencial dos LLMs na construção de sistemas, pavimentando o caminho para sistemas de informação que utilizam linguagem natural como interface.

Abstract. We present a question-and-answer (Q&A) application for the contract management process. Combined information captured from contracts (PDF) and data from the contractual process support system is sent to GPT-4 to provide accurate answers. The relevance of the responses is improved through Augmented Retrieval (RAG) and text-to-SQL techniques, eliminating the need to retrain the language model. We also explored Prompt Engineering to focus the responses better. Throughout our work, we observe that these combined techniques increase the relevance of the answers. We highlight the potential of LLMs in building systems, paving the way for information systems that use natural language as the primary interface.

1. Introdução

Em grandes corporações, a gestão de contratos abrange o processo de contratação de serviços ou produtos, a garantia de sua execução correta e a avaliação contínua do desempenho operacional e financeiro da prestação do serviço ou produto. Nas empresas públicas, esses desafios são ampliados, conforme estipulado pela Lei nº 14.133/2021, a gestão de contratos abrange o gerenciamento de todas as atividades relacionadas à execução do contrato. Isso inclui a fiscalização técnica, administrativa e setorial, bem como procedimentos necessários para a prorrogação, reequilíbrio econômico-financeiro, alteração e aplicação de sanções.

*A.S. Medeiros, C. Cavalcante e S. Lifschitz são parcialmente financiados pela CAPES (institucional) e CNPq (auxílio individual).

Para gerenciar eficazmente todas essas atividades, unidades organizacionais dedicam-se a apoiar o processo de gestão de contratos. Em muitos casos, são necessárias subunidades com conhecimento especializado, como, por exemplo, em contratos de serviços e produtos de tecnologia da informação e comunicação (TIC), serviços e materiais de administração predial e patrimonial, entre outros.

Para essa atividade, é fundamental contar com o apoio de um sistema de gerenciamento de contratos. Nesse contexto, as empresas públicas ou desenvolvem seus sistemas ou utilizam soluções no mercado, como o *SAP Contract Lifecycle Management*, *IBM Emptoris Contract Management*, entre outros.

Embora esses sistemas gerenciem dados gerais dos contratos, como data de assinatura, vencimento e gestor, existem detalhes que são obtidos apenas nos documentos. Estes incluem questões sobre penalidades, descontos por atrasos, ou ainda questões que envolvem diversos contratos, por exemplo, quais contratos de manutenção elétrica ou de elevadores são aplicáveis a um prédio público.

Nosso objetivo neste trabalho é desenvolver uma aplicação que apoie as equipes de gestão de contratos nessas atividades. Utilizamos as mais recentes tecnologias de Modelos de Linguagem de Grande Escala (LLM) para analisar os textos dos contratos. Indo além, nossa aplicação enriquece a qualidade das respostas ao integrar informações dos sistemas tradicionais de gestão de contratos.

Desenvolvemos um sistema de perguntas e respostas (Q&A) focado em contratos de Tecnologia da Informação e Comunicação (TIC), utilizando como fontes de dados os arquivos PDF dos contratos e os dados dos Sistemas de Gerenciamento de Contratos. Com o objetivo de aumentar a relevância das informações sobre os contratos empregamos a técnica de Geração com Recuperação Aumentada (RAG); técnicas de extração *text-to-SQL* para obter informações pertinentes dos sistemas de contratos; e técnicas de Engenharia de *Prompt* para padronizar e garantir maior precisão nas respostas produzidas.

Um dos principais desafios está relacionado à padronização dos formatos e à redação de documentos contratuais. Esta uniformização é um desafio para os LLMs pois existe uma grande similaridade textual, que não necessariamente se reflete em relevância. Por meio de técnicas, desenvolvemos uma solução que minimiza o impacto da uniformização e fornece respostas relevantes. Esta abordagem possibilitou a elaboração de uma solução sem a necessidade do tradicional *fine-tuning* ou re-treinamento dos modelos de linguagem.

O artigo está organizado da seguinte maneira: A Seção 2 oferece uma contextualização técnica sobre LLMs, *text-to-SQL* e engenharia de *prompt*. A Seção 3 aborda trabalhos correlatos na área, enquanto a Seção 4 detalha a metodologia para a integração de textos de contratos e dados estruturados com modelos de linguagem. A Seção 5 descreve aspectos da implementação e experimentação da aplicação de perguntas e respostas. Por fim, a Seção 6 conclui nosso estudo e propõe direções para futuras pesquisas neste campo.

2. Referencial Teórico

A disseminação de várias aplicações na área de Processamento de Linguagem Natural (NLP) foi possibilitada pelos Modelos de Linguagem de Grande Escala (LLMs), incluindo os sistemas de perguntas e respostas (Q&A). Estes sistemas são atrativos como interface para recuperar informação de domínio de conhecimento específico. Para melho-

rar as respostas geradas pelos LLMs, algumas técnicas se destacam, como: Recuperação Aumentada (RAG), *text-to-SQL* e engenharia de *prompt*.

2.1. Grandes Modelos de Linguagem

Os Modelos de Linguagem de Grande Escala (LLMs) revolucionaram o campo do processamento de linguagem natural com sua capacidade de entender e gerar texto semelhante ao humano. Em sua arquitetura, eles utilizam uma estrutura específica de rede neural, os *Transformers*, que permite ao modelo ponderar a influência de diferentes partes dos textos de entrada em momentos distintos [Vaswani et al. 2017].

As aplicações de conversação, um uso específico dos LLMs, são especializadas na geração de texto que é coerente e contextualizado. Isto é alcançado através de treinamento, no qual os modelos são alimentados com vastas quantidades de dados de conversação, permitindo que eles aprendam as nuances do diálogo [OpenAI 2023a].

Dessa forma, por meio da paralelização e dos mecanismos de atenção, os LLMs estabeleceram um novo paradigma para o processamento da linguagem natural. Ao expandir o espaço de busca com dados externos ou se especializar por meio de ajustes finos, os LLMs se tornam plataformas para a construção de aplicativos especializados e ricos em conhecimento. Esses aplicativos podem apresentar informações de maneira semelhante a um diálogo, apoiando a busca por informações relevantes e gerando *insights* para a tomada de decisões.

2.2. Recuperação aumentada (RAG)

De acordo com [Chen et al. 2024], os LLMs enfrentam desafios significativos, como alucinação factual, conhecimento desatualizado e falta de *expertise* específica de domínio. Em resposta a estes desafios, o RAG representa uma mudança de paradigma na forma como os LLMs processam e geram textos. O princípio por trás do RAG envolve o uso do armazenamento de vetores para recuperar trechos de texto similares à consulta de entrada [Gao et al. 2023b]. Essa técnica converte tanto o texto da consulta quanto o banco de dados de informações em vetores de alta dimensão, permitindo que se recupere informações similares, que posteriormente são fornecidas para um LLM.

[Gao et al. 2023b] e [Feng et al. 2024] descrevem *frameworks* que exploram as vantagens dessa técnica ao fornecer dados adicionais ao LLM sem re-treinar o modelo [Li et al. 2022]. Ao dividir o texto disponível em pedaços gerenciáveis e incorporar esses pedaços em espaços vetoriais de alta dimensão, é possível realizar a rápida recuperação de informações contextualmente relevantes em resposta a uma consulta, o que subsidia as próximas etapas de processamento. Conforme mostrado pela Figura 1, a primeira etapa (1) envolve a leitura do conteúdo textual dos documentos PDF em pedaços gerenciáveis (*chunks*), que são então transformados (*embedding*) (2) em vetores de alta dimensão. O texto em formato vetorial captura as propriedades semânticas do texto, um formato que pode possuir 1536 dimensões.

Estes vetores de *embeddings* são armazenados em uma *vectorstore* (3), um banco de dados especializado em vetores de alta dimensão. O armazenamento de vetores permite a consulta eficiente de vetores através de suas similaridades, utilizando a distância para comparação (seja *Manhattan*, euclidiana ou cosseno).

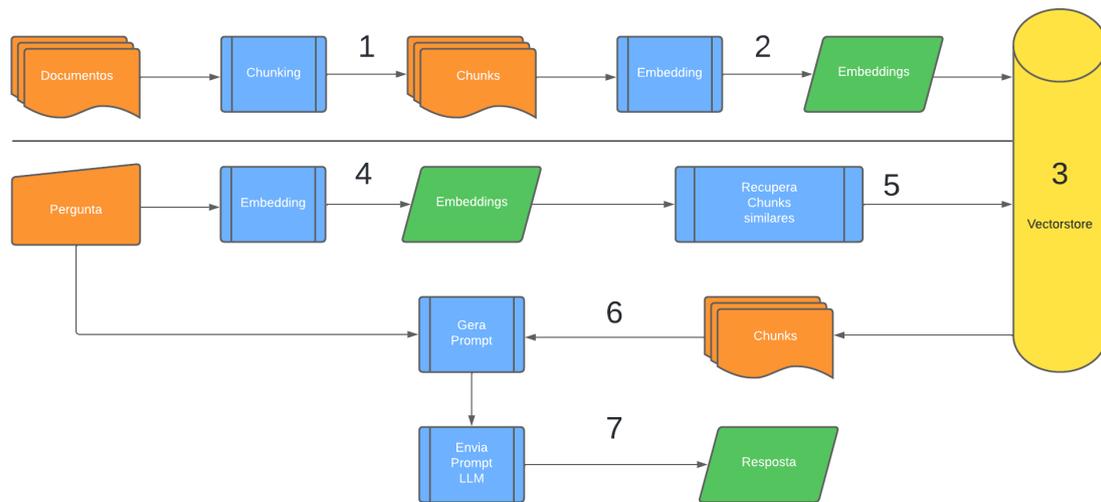


Figure 1. Retrieval Augmented Generation

Uma vez estabelecida a métrica de similaridade, é realizado o *embedding* da consulta no mesmo espaço vetorial (4); e isso permite uma comparação direta entre a consulta vetorizada e os vetores dos trechos armazenados, recuperando os trechos mais similares (5), que são então integrados de forma transparente ao contexto do LLM para gerar um *prompt* (6). O *prompt* é então composto pela pergunta, pelos textos recuperados da *vectorstore*, pelas instruções específicas e, opcionalmente, pelo histórico do *chat*, tudo enviado para o LLM que gera a resposta final (7).

No RAG, a estratégia de *chunking* é importante pois influencia diretamente a qualidade das informações recuperadas. Uma divisão dos *chunks* em trechos bem concebida garante que a informação seja coesa e semanticamente completa, capturando sua essência.

Um aspecto-chave no RAG é a diferença entre similaridade e relevância. Trechos similares podem não conter as informações relevantes para responder a uma pergunta, representando um desafio para recuperar informações com precisão, especialmente em casos em que os dados vêm de vários documentos com estrutura semelhante. Em tais contextos, os documentos podem compartilhar um alto grau de similaridade estrutural e léxica, tornando difícil para os algoritmos de recuperação distinguir entre conteúdo meramente similar em forma e o conteúdo verdadeiramente relevante para uma consulta.

2.3. Text-to-SQL

Text-to-SQL é uma tecnologia que possibilita a conversão de perguntas em linguagem natural em comandos SQL baseados exclusivamente no esquema do banco de dados, eliminando a necessidade de conhecimento dos dados subjacentes [Liu et al. 2023]. Essa abordagem aproveita as capacidades dos LLMs para entender e interpretar a linguagem humana, permitindo aos usuários recuperarem dados em bancos de dados por meio de entradas de texto simples, sem a necessidade de conhecimento especializado da sintaxe da linguagem SQL [Gao et al. 2023a].

Ao traduzir linguagem natural em consultas SQL, *text-to-SQL* aproxima estruturas complexas de banco de dados e usuários finais, tornando o acesso mais intuitivo e

eficiente. Essa técnica é particularmente útil porque permite que usuários leigos tenham acesso aos bancos de dados através de perguntas em linguagem natural. Ela melhora a acessibilidade aos dados, reduz a curva de aprendizado associada à consulta de bancos de dados e acelera os processos de análise de dados, permitindo um maior número de usuários a tomar decisões orientadas por dados.

A principal distinção entre as técnicas RAG e *text-to-SQL* reside na forma em que a informação é buscada. RAG baseia-se em recuperar segmentos de texto de um armazenamento de vetores que são semelhantes à pergunta do usuário, usando esses segmentos para gerar uma resposta coerente e contextualmente relevante. Este método é eficaz para perguntas onde a resposta pode ser sintetizada a partir de texto existente. Mas nem sempre é possível identificar as informações esperadas como resposta. Em outro aspecto, o *text-to-SQL* traduz consultas em linguagem natural em comandos SQL, como demonstrado em [Pinheiro et al. 2023], que são então executados em um banco de dados estruturado para recuperar correspondências exatas de dados. Isso garante que, se a tradução de texto para SQL for precisa, o usuário receberá uma resposta altamente específica diretamente dos campos de banco de dados.

Portanto, enquanto RAG opera com base no princípio de similaridade textual e capacidades generativas, *text-to-SQL* oferece um mecanismo mais intrusivo para recuperação de dados ao executar consultas que correspondem diretamente à intenção do usuário, tornando-o particularmente eficaz para investigações de dados.

2.4. Engenharia de *Prompt*

A engenharia de *prompt* é a arte de projetar e otimizar *prompts* para orientar os LLMs na geração de saídas desejadas. O objetivo da engenharia de *prompt* é maximizar o potencial dos LLMs fornecendo-lhes instruções e contexto [OpenAI 2023b].

No escopo da engenharia de *prompt*, as instruções são parte fundamental. Através delas, os engenheiros podem detalhar o roteiro para uma resposta, especificando o estilo e o formato desejados para a resposta do LLM [White et al. 2023] [Giray 2023]. Por exemplo, para definir o estilo de uma conversa, um *prompt* poderia ser formulado como "Utilize linguagem profissional e trate o cliente com respeito" ou "Utilize linguagem informal e emojis para transmitir um tom amigável". Para especificar o formato das datas nas respostas, uma instrução de *prompt* poderia ser "Utilize o formato americano, MM/DD/YYYY, para todas as datas".

Por outro lado, o contexto refere-se às informações fornecidas aos LLMs juntamente com as instruções principais. O aspecto mais importante de um contexto é que ele pode fornecer informações adicionais para apoiar a resposta dada pelo LLM, sendo muito útil ao implementar sistemas de perguntas e respostas. Esse contexto suplementar pode incluir detalhes de fundo relevantes, exemplos específicos e até mesmo trocas de diálogo anteriores, que coletivamente ajudam o modelo a gerar respostas mais precisas, detalhadas e contextualmente apropriadas. De acordo com [Wang et al. 2023], os *prompts* fornecem orientação para garantir que o modelo gere respostas alinhadas com a intenção do usuário. Como resultado, *prompts* bem elaborados melhoram significativamente a eficácia e a adequação das respostas.

Estudos recentes começaram a explorar a integração sinérgica dessas técnicas com LLMs para criar sistemas de perguntas e respostas mais sofisticados. Por exem-

plo, [Jeong 2023] reforça a importância do uso de Engenharia de Prompt com RAG para melhorar a recuperação de documentos relevantes, que são então usados para gerar respostas tanto contextualmente relevantes quanto ricas em informações. Da mesma forma, [Gao et al. 2023a] explora a integração de *text-to-SQL* com Engenharia de Prompt para aprimorar a capacidade do modelo de interagir diretamente com bancos de dados relacionais, expandindo assim o escopo de consultas que podem ser respondidas com precisão.

3. Metodologia

Conforme apresentado na Seção 1 para superar os desafios atuais enfrentados pelas equipes gestoras de contratos, desenvolvemos um sistema de perguntas e respostas (Q&A), o Contrato360. O sistema é suportado pelo GPT-4, focado nos contratos de TIC do BNDES, que utiliza um conjunto de técnicas que melhoram a relevância das respostas e reduzem o risco decorrente da uniformização textual dos contratos.

Para atingir este objetivo de ampliar a relevância das respostas obtidas pelo Contrato360, conjugamos três técnicas: 1) Geração com Recuperação Aumentada (RAG) para aumentar a relevância das informações sobre os contratos; 2) técnicas de extração *text-to-SQL* para obter informações pertinentes dos sistemas de contratos; e 3) técnicas de Engenharia de Prompt para padronizar e garantir maior precisão nas respostas produzidas.

3.1. Aplicando o RAG

Na construção do Contrato360, a primeira técnica aplicada foi RAG. Como explicado na Subseção 2.2, a primeira decisão a ser tomada é a escolha da melhor estratégia para segmentar o documento, ou seja, como realizar o *chunking* dos arquivos PDFs. Uma estratégia comum de *chunking* envolve a segmentação de documentos com base em um número específico de *tokens* e uma sobreposição (*overlap*). Isto é útil ao tratar textos sequenciais onde é importante manter a continuidade do contexto entre os *chunks*.

Como mencionamos, os contratos possuem uma estrutura textual padronizada, organizada em seções contratuais. Portanto, suas seções com a mesma numeração ou na sua vizinhança descrevem o mesmo aspecto contratual, isto é, possuem semântica semelhante. Por exemplo, na primeira seção dos documentos dos contratos, sempre encontramos o objeto do serviço ou produto. Neste cenário, podemos pressupor que a melhor estratégia de *chunking* é separar os *chunks* por seção do documento. Nesse caso, o *overlap* entre os *chunks* ocorre por seção, dado que as perguntas serão respondidas por informações contidas na própria seção ou em seções anteriores ou posteriores. Para a página do contrato no exemplo da Figura 2, teríamos um *chunk* para a seção do objeto do contrato, outro *chunk* para a seção da vigência do contrato, ou seja, um *chunk* para cada cláusula do contrato e suas adjacências. Essa abordagem garante que cada trecho represente uma unidade semântica, tornando as recuperações mais precisas alinhadas com as consultas.

Tendo a seção do contrato como limite dos *chunks* melhora a relevância das respostas dentro de um único contrato. Entretanto, ao ampliar o número de contratos que o Contrato360 pretende responder, observamos o problema em determinar corretamente o contrato a ser tratado, no exemplo a seguir, detalhamos este aspecto:

Considere os documentos de contratos mostrados na Figura 2. A "CLÁUSULA PRIMEIRA – OBJETO" detalha o objeto do contrato e uma pergunta frequente é: "Qual

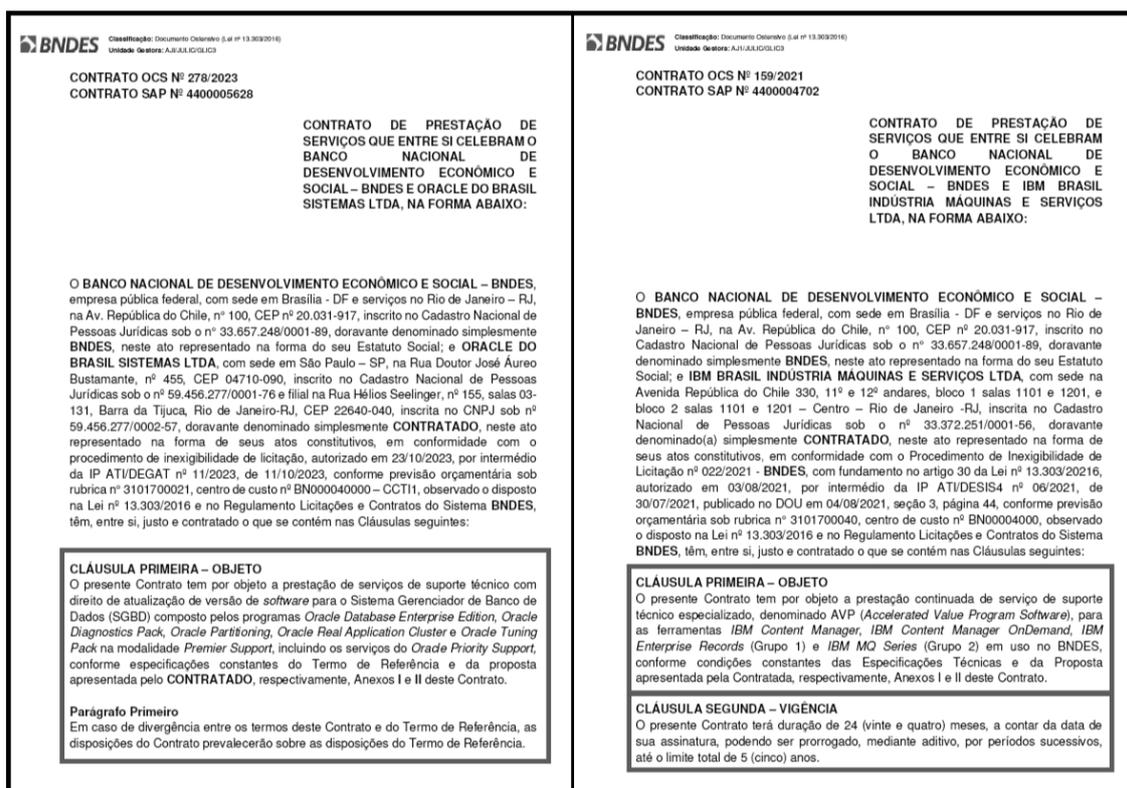


Figure 2. Chunking aplicado a Contratos

o objeto do contrato OCS 278/2023?”. Neste exemplo, o RAG irá armazenar vetores contendo as seções dos dois contratos, dado que esta cláusula é comum a ambos. Mas, ao inspecionarmos o que está expresso no *chunk*, seu conteúdo não contém o número do contrato, Figura 2. Desta forma, com grande probabilidade, uma consulta sobre um contrato específico, pode retornar um segmento (*chunk*) não relacionado ao contrato, por exemplo OCS 159/2021, seja recuperado ao invés do contrato que desejamos. No caso do nosso exemplo o *chunk* referente a pergunta que deve ser retornado é relativo ao contrato OCS 278/2023.

Para superar essa questão, é necessário adicionar semântica aos *chunks*, através da inclusão de metadados dos documentos. E ao acessar a *vectorstore* utilizar estes metadados para filtrar a informação retornada. Dessa forma, melhoramos a relevância dos textos recuperados. A Figura 3 exibe os metadados mais relevantes para os contratos (fonte, contrato e cláusula). Onde fonte é o nome do arquivo PDF do contrato), contrato é o número OCS e cláusula é o título da seção. Assim, para a pergunta “Qual o objeto do contrato OCS 278/2023?”, são recuperados os *chunks* do contrato OCS 278/2023 e, em seguida, aplicado o cálculo da similaridade recuperando os segmentos de texto para enviado ao LLM.

3.2. Aplicando o *text-to-SQL*

Os contratos são dinâmicos e passam por alterações e ajustes ao longo de sua vida. Para lidar com essa complexidade, as organizações utilizam sistemas de acompanhamento de contratos, como o *SAP Contract Lifecycle Management*. Esses sistemas controlam diversos aspectos, como o responsável técnico do contrato, alterações no preposto da con-

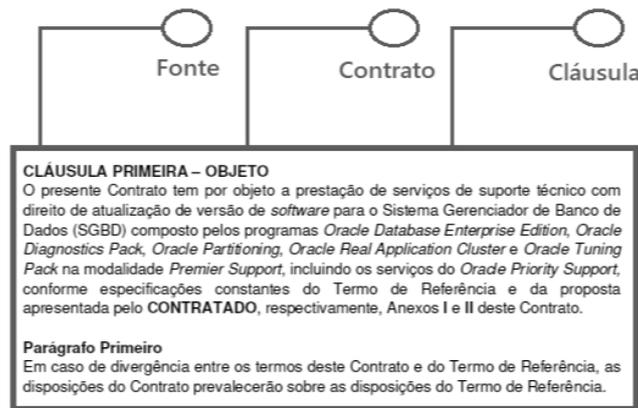


Figure 3. Metadados de Contratos

tratada e o término da prestação de serviços. Durante a vigência do contrato, esses eventos podem ocorrer e afetar significativamente a gestão contratual.

Para incorporar esses eventos ao Contrato360, e melhorar a relevância das respostas, foi necessário recuperar essas informações do sistema de acompanhamento de contratos. E para tanto, o uso da técnica *text-to-SQL* foi natural, pois viabiliza respostas mais precisas e atualizadas aos gestores dos contratos buscando informações nas bases de dados que suportam o sistema de acompanhamento de contratos, não se limitando apenas ao documento do contrato.

O uso tradicional do *text-to-SQL* para recuperar conteúdo de um banco de dados, é traduzir a pergunta do usuário, feito em linguagem natural, para código SQL e, em seguida, consultar o banco de dados. Podemos observar que no contexto do sistema de Q&A proposto, para poder responder ao usuário, a avaliação da pergunta no sistema deve decidir entre recuperar do sistema ou buscar na *vectorstore*, ou seja, o fluxo teria as seguintes etapas: 1) tradução da consulta em SQL e recuperação dos dados do sistema de acompanhamento; ou 2) busca e recuperação dos *chunks* na *vectorstore*; e 3) submissão do resultado ao GPT-4. No entanto, determinar a melhor estratégia (entre 1 ou 2), no momento em que o usuário submete suas perguntas, não é trivial.

Considerando essas dificuldades, a abordagem escolhida foi a de enriquecer a *vectorstore* com consultas pré-estabelecidas pelos gestores de contratos e previamente submetidas ao sistema de contratos. Consequentemente, se evita a decisão de em tempo de execução escolher a fonte de resposta, a informação estará sempre na *vectorstore*. Além disso, este enriquecimento extrai diversas dimensões textuais dos dados dos sistemas, aumentando a riqueza semântica dos dados armazenados na *vectorstore*. Essa escolha arquitetural se mostrou robusta nos relatos dos gestores de contrato, pois enriquece semanticamente as informações dos contratos, conforme mostra a Figura 4.

Desta forma, o sistema requer apenas uma única *vectorstore* para recuperar textos tanto dos PDFs quanto do banco de dados. As informações pesquisadas no banco de dados foram geradas a partir das perguntas mais frequentes dos usuários. Quando uma pergunta é feita pelo usuário, o sistema de recuperação seleciona apenas os *chunks* mais similares à pergunta na *vectorstore*. Em seguida, gera o prompt a ser enviado para o LLM.

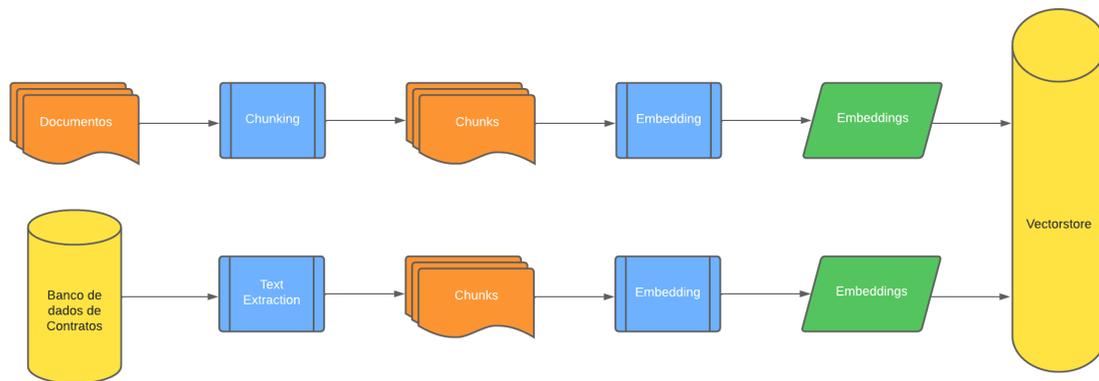


Figure 4. vectorstore de Contratos

A partir desse ponto, o processo de construção da resposta segue a abordagem clássica.

3.3. Aplicando a Engenharia de Prompt

A técnica engenharia de *prompt* fornece um padrão (*pattern*) para o estilo das respostas e o reuso da solução no acesso ao LLM, pois provê instruções e contexto [White et al. 2023]. Partindo dessas observações, na aplicação, instruções foram elaboradas para melhorar as respostas. As instruções com orientações básicas, como “*Não utilizar conhecimento prévio*”, que garante que as respostas sejam baseadas apenas nos contratos da *vectorstore*, e instruções específicas, como “*Toda vez que responder a uma pergunta sobre um contrato, informar o número da OCS.*” Assim, a pergunta “*Temos algum contrato de Suporte Oracle?*” teria como uma possível resposta “*Sim, temos um contrato de Suporte ao Oracle Database. O número da OCS é 278/2023.*”.

A manutenção e as orientações de como utilizar o contexto do chat também foram aplicadas para garantir uniformidade e coerência. Por exemplo, informamos o estilo esperado para as respostas: “*Você deve usar um tom formal e objetivo.*”, determinando o estilo das respostas do LLM. Outra orientação instrui o LLM: “*Dado o histórico de bate-papo e a pergunta realizada, construa a resposta de forma completa, sem que o usuário precise rever o histórico*”.

Por fim, o contexto passado para o LLM pode ser útil para estabelecer o estilo das respostas em função do papel do usuário do sistema de Q&A. No caso do Contrato360, temos três papéis: 1) gestor do contrato; 2) apoio a gestão de contratos; e 3) gerente da unidade de apoio a gestão de contratos. Para cada um desses papéis foi definido um contexto específico, por exemplo para o papel 3 temos: “*Você é um assistente especializado em responder perguntas sobre contratos administrativos, que fornece informações gerenciais e sumarizadas sobre os contratos.*”

Com estas três técnicas obtivemos respostas mais relevantes. Na seção a seguir, detalhamos a implementação e os componentes utilizados no desenvolvimento do sistema.

4. Arquitetura da aplicação e Experimento

A arquitetura da aplicação é composta por três principais componentes: a camada de backend para processamento e armazenamento de dados, a camada de integração com

modelos de linguagem, e a camada de interface do usuário. Na camada de backend, utilizamos Python e a biblioteca Langchain para gerenciar a integração com os modelos de linguagem da OpenAI. A funcionalidade central envolve a preparação e indexação dos documentos que serão consultados pela aplicação. Criamos uma classe Document em Python para encapsular informações sobre cada documento, incluindo o conteúdo da página e metadados associados.

Para permitir a busca eficiente e a recuperação de informações, utilizamos a biblioteca Chroma para criar um banco de dados de vetores, utilizando embeddings gerados por modelos da OpenAI. Inicializamos um objeto vectordb usando a biblioteca Chroma, que cria um índice de busca de vetores a partir de uma coleção de documentos. Utilizamos embeddings fornecidos pelo modelo text-embedding-ada-002 da OpenAI. Para implementação da recuperação de respostas usando a abordagem RAG, utilizamos o módulo RetrievalQA da Langchain. Este módulo recupera os trechos mais similares do armazenamento de vetores e passa esses *chunks* para o modelo de linguagem responder à pergunta.

Para a interface do usuário, utilizamos o Streamlit 6, que permite aos usuários interagir com o sistema de forma intuitiva e eficiente. Através dessa interface, os usuários podem fazer perguntas, receber respostas e exportar conversas. A aplicação suporta um histórico de chat, permitindo que perguntas anteriores sejam consideradas nas respostas futuras. Por exemplo, se um usuário pergunta "Quem é o gestor desse contrato?" após perguntar sobre um contrato específico, a aplicação é capaz de contextualizar e transformar a pergunta para obter uma resposta precisa.

O experimento para validar a aplicação foi conduzido por dois especialistas em contratos de TIC do BNDES. O sistema foi preparado com 75 contratos (PDFs e dados do sistema de contratos). E para validar a relevância das respostas, foram preparadas questões de *benchmark*, de dois grupos distintos: perguntas "diretas" e "indiretas". Perguntas "diretas" são aquelas que podem ser respondidas através dos PDFs e seus metadados. Perguntas "indiretas" são aquelas que obtêm uma melhor relevância quando buscadas nos dados do sistema de contratos. Nas Tabelas 1 e 2 apresentamos a percepção dos usuários sobre a qualidade das respostas. Na avaliação, a relevância da respostas foi categorizada em "Certa" e "Incompleta".¹

Podemos observar que para as perguntas "diretas" o sistema apresenta respostas relevantes para todos experimentos. Porém, nas perguntas "indiretas", apesar de satisfatórios, os resultados em duas questões específicas foram limitadas e incompletas. Na nossa avaliação, estas questões necessitam uma avaliação semântica mais elaborada, no primeiro caso, percebemos que o conceito de "Dispensa de Licitação", não foi bem cap-

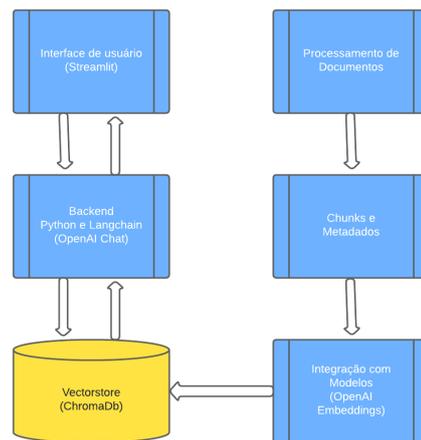


Figure 5. Arquitetura da aplicação

¹Uma terceira categoria seria "Incorreta", mas essa opção não foi obtida em nenhuma das perguntas.

turado; e no segundo caso temos uma implicação lógica dupla: gestor → contrato → empresa, que também não foi capturado pela aplicação. Acreditamos que um ajuste nas consultas no sistema possam agregar este tipo de semântica.

Pergunta	Certo	Incomp.
Qual o objeto do contrato OCS nnn/yy?	10	-
Temos algum contrato cujo objeto é xxxx?	9	1
Temos algum contrato com o fornecedor xxx?	10	-
Quem é o gestor do contrato OCS nnn/yy?	10	-
Qual o fornecedor do contrato nnn/yy?	10	-
Qual a vigência do contrato OCS nnn/yy?	10	-

Table 1. Perguntas Diretas

Pergunta	Certo	Incomp.
Quantos contratos ativos de TI temos no momento?	10	-
Liste os contratos que terão sua vigência encerrada no ano de yy?	10	-
Quantos contratos do fornecedor xxxx temos?	10	-
Quantos contratos temos celebrado por inexibilidade?	9	1
Quantas DLs (Dispensas de Licitações) foram contratadas em yy?	-	10
Quem são os gestores dos contratos que temos com a empresa xxxx?	-	10
O empregado xxxx tem quantos contratos sob sua gestão?	8	2
Mostre um resumo do contrato nnn/yy.	10	-

Table 2. Perguntas Indiretas

5. Conclusões e Trabalhos Futuros

Desenvolvemos uma aplicação de Q&A no domínio de contratos de serviços e produtos, utilizando como fontes de informação os contratos PDF e os dados do sistema de gerenciamento de contratos. Nós empregamos três técnicas para melhorar a relevância das respostas: 1) Recuperação Aumentada (RAG) aliada ao incremento semântico, usando metadados para recuperar informações dos PDFs; 2) *text-to-SQL*, agregando informações dinâmicas dos contratos disponibilizados no sistema de gerenciamento; e 3) Engenharia de *prompt* para contextualizar, instruir e direcionar as respostas produzidas pelo LLM.

Contrato360



Figure 6. Interface da aplicação de Q&A

Os resultados percebidos pelos usuários foram promissores e demonstram o potencial para a construção de sistemas de interação e acesso aos dados. No entanto, para ampliar a eficácia e a escalabilidade da solução proposta, é necessário aprofundar a compreensão das necessidades dos usuários além do conhecimento específico sobre contratos.

Os experimentos abordaram um conjunto inicial de questões capaz de produzir um sistema robusto que atende às atuais necessidades. Aprofundar outras questões enriquecerá os metadados e o conjunto de consultas que extraem informações dos sistemas tradicionais.

Por fim, para consolidar as técnicas desenvolvidas para atender à nossa aplicação, vislumbramos que a construção de um sistema em um outro domínio de aplicação poderá

trazer luz sobre limitações e a necessidade de refinamento ou adaptação. Essa exploração futura não apenas reforçará a confiança no uso dessas técnicas em cenários do mundo real, mas também abrirá caminho para sua otimização e possível personalização, aumentando, em última análise, a utilidade e o impacto dos LLMs em aplicações corporativas.

Disclaimer: Este artigo representa a opinião dos autores e é produto de sua pesquisa profissional. Não foi realizado para representar a posição ou as opiniões do BNDES ou de seus membros, nem tão pouco representa qualquer posição oficial da Instituição.

References

- Chen, J., Lin, H., Han, X., and Sun, L. (2024). Benchmarking large language models in retrieval-augmented generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17754–17762.
- Feng, Z., Feng, X., Zhao, D., Yang, M., and Qin, B. (2024). Retrieval-generation synergy augmented large language models. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11661–11665. IEEE.
- Gao, D., Wang, H., Li, Y., Sun, X., Qian, Y., Ding, B., and Zhou, J. (2023a). Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.
- Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., and Wang, H. (2023b). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.
- Giray, L. (2023). Prompt engineering with chatgpt: a guide for academic writers. *Annals of biomedical engineering*, 51(12):2629–2633.
- Jeong, C. (2023). A study on the implementation of generative ai services using an enterprise data-based llm application architecture. *arXiv preprint arXiv:2309.01105*.
- Li, H., Su, Y., Cai, D., Wang, Y., and Liu, L. (2022). A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*.
- Liu, A., Hu, X., Wen, L., and Yu, P. S. (2023). A comprehensive evaluation of chatgpt’s zero-shot text-to-sql capability. *arXiv preprint arXiv:2303.13547*.
- OpenAI (2023a). Chatgpt fine-tune description. <https://help.openai.com/en/articles/6783457-what-is-chatgpt>. Accessed: 2024-03-01.
- OpenAI (2023b). Chatgpt prompt engineering. <https://platform.openai.com/docs/guides/prompt-engineering>. Accessed: 2024-04-01.
- Pinheiro, J., Victorio, W., Nascimento, E., Seabra, A., Izquierdo, Y., Garcia, G., Coelho, G., Lemos, M., Leme, L. A. P. P., Furtado, A., et al. (2023). On the construction of database interfaces based on large language models. In *Proceedings of the 19th International Conference on Web Information Systems and Technologies - Volume 1: WEBIST*, pages 373–380. INSTICC, SciTePress.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- Wang, M., Wang, M., Xu, X., Yang, L., Cai, D., and Yin, M. (2023). Unleashing chatgpt’s power: A case study on optimizing information retrieval in flipped classrooms via prompt engineering. *IEEE Transactions on Learning Technologies*.
- White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., and Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*.