

ASSISTENTE INTELIGENTE PARA CONFIGURAÇÃO DO SAGE BASEADO EM GRANDES MODELOS DE LINGUAGEM

Pablo Mascarenhas de Araújo (*)
ELETROBRAS

RESUMO

Este trabalho apresenta um assistente inteligente para configuração de base de dados do SAGE, utilizando processamento de linguagem natural utilizando grandes modelos de linguagem. Buscou-se criar um assistente capaz de fornecer respostas a perguntas sobre a estrutura da base de dados, atributos, relacionamentos, e exemplos de configuração, além de auxiliar na criação, edição e correção de arquivos fonte. Outro objetivo é oferecer orientações para a configuração da base de acordo com as necessidades específicas dos protocolos de comunicação. O desenvolvimento deste assistente visa facilitar o trabalho de especialistas, tornando o processo mais eficiente e intuitivo.

PALAVRAS-CHAVE

SAGE, Inteligência Artificial, Processamento de Linguagem Natural, Sistemas Supervisórios, Grandes Modelos de Linguagem

1.0 - INTRODUÇÃO

O sistema supervisório SAGE, desenvolvido pelo CEPEL, é amplamente utilizado no setor elétrico brasileiro para a gestão de sistemas de energia. A configuração de sua base de dados é uma etapa crítica que define a estrutura e os parâmetros do sistema a ser supervisionado e controlado. No entanto, essa tarefa é complexa, exigindo expertise técnica e domínio da extensa documentação, o que a torna suscetível a erros que podem afetar a operação do sistema.

Os avanços recentes em Inteligência Artificial (IA), especialmente no Processamento de Linguagem Natural (PLN) através de Grandes Modelos de Linguagem (Large Language Model - LLM), como o GPT-4, possibilitam a compreensão e geração de linguagem natural sofisticada. Esses modelos oferecem potencial para aplicações em sistemas complexos como o SAGE, incluindo assistência na configuração e interpretação de documentação técnica.

Este artigo propõe o desenvolvimento de um assistente inteligente para a configuração da base de dados do SAGE, utilizando LLMs para interpretar a documentação técnica, responder a perguntas sobre configuração, auxiliar na escolha de parâmetros, detectar e corrigir erros, além de automatizar tarefas repetitivas, visando facilitar o trabalho dos especialistas.

2.0 - FUNDAMENTAÇÃO TEÓRICA

2.1 Grandes Modelos de Linguagem

Grandes Modelos de Linguagem (LLM) como o GPT4 representam um grande avanço na inteligência artificial, especialmente no processamento de linguagem natural. Eles são capazes de compreender e gerar texto de forma coerente e contextualizada, sendo utilizados em diversas aplicações como assistentes virtuais, tradução automática e resumo de textos. Para entender como os LLMs são usados no contexto deste trabalho, alguns conceitos são fundamentais:

- Tokens*: são as menores unidades de texto processadas pelo modelo, podendo ser palavras inteiras, subpalavras ou caracteres, conforme a tokenização empregada. Por exemplo, a frase "Configuração do SAGE" pode ser tokenizada em "Configuração", "do" e "SAGE". A tokenização é o processo que converte o texto em *tokens*, influenciando a interpretação do modelo.
- Embeddings*: São representações numéricas (vetores) dos *tokens*, capturando características semânticas e sintáticas das palavras. Cada token é convertido em um embedding, permitindo que o modelo processe o texto matematicamente. No espaço vetorial de alta dimensão, *tokens* com significados semelhantes têm *embeddings* próximos; por exemplo, "transformador" e "reator" estariam próximos nesse espaço.
- Pesos da rede neural: são parâmetros ajustados durante o treinamento do modelo, determinando como os *embeddings* são combinados para gerar a saída. O ajuste dos pesos minimiza a diferença entre as previsões do modelo e os dados reais, utilizando métodos de otimização como o gradiente descendente.

(*) Rua Delmiro Gouveia, nº 333, Sala A304 – CEP 50.761-901 Recife, PE – Brasil. Tel: (+55 81) 99434-8677 – E-mail: pablom@eletrobras.com

- d. Janela de atenção (janela de contexto): Define a quantidade de *tokens* que o modelo processa simultaneamente. Enquanto modelos tradicionais possuem limites de alguns milhares de *tokens*, avanços como o Gemini 1.5 permitem janelas de até 2 milhões de *tokens*.

Os LLMs modernos baseiam-se na arquitetura *transformer*, que utiliza mecanismos de atenção para focar em diferentes partes do texto de entrada, capturando relações contextuais complexas (1). Essa capacidade é crucial para compreender e gerar linguagem em domínios técnicos.

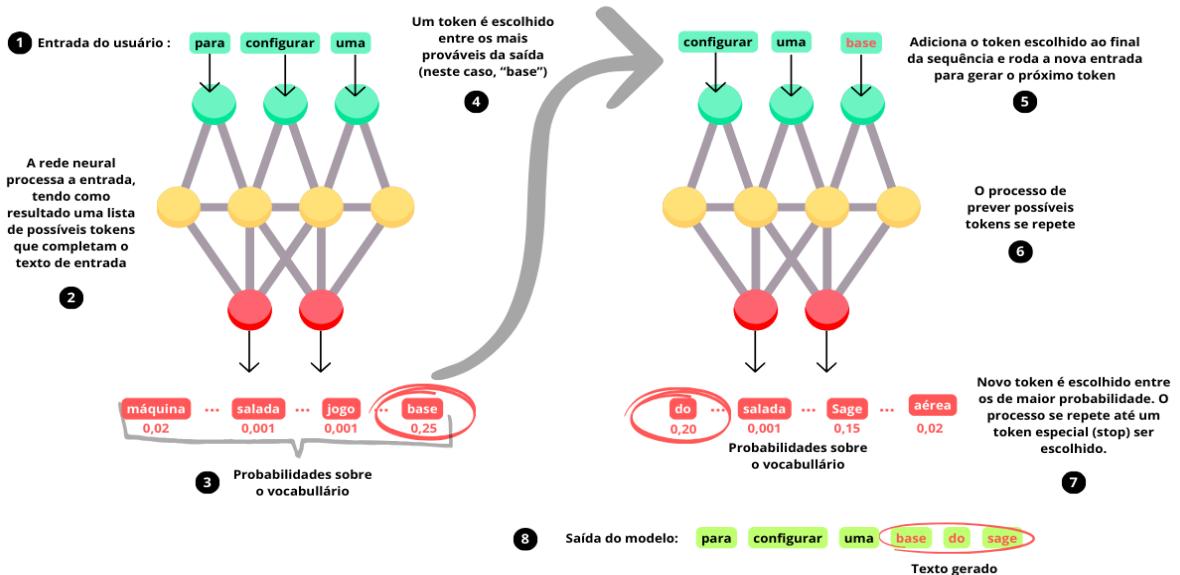


Figura 1 - Processo de inferência de um LLM

Durante o treinamento, os modelos são expostos a grandes quantidades de texto, aprendendo a prever o próximo token em uma sequência. Eles capturam padrões linguísticos sem armazenar informações específicas, gerando respostas dinamicamente com base nas probabilidades aprendidas, e não pela recuperação de trechos pré-armazenados (ver Figura 1). Assim, um LLM funciona como alguém que, após ler muitos livros, constrói respostas baseadas no conhecimento adquirido, sem recitar trechos específicos.

2.2 Capacidades em domínios técnicos

Para recuperar eficientemente informações específicas em grandes volumes de documentação, utilizam-se técnicas avançadas, como indexação semântica latente e modelos de linguagem pré-treinados. Essas abordagens permitem identificar conceitos-chave e suas inter-relações em documentos técnicos.

2.2.1 Técnicas de extração de informação

A recuperação eficiente de informações específicas em grandes volumes de documentação é viabilizada por técnicas avançadas de busca, como a indexação semântica latente e o uso de modelos de linguagem pré-treinados. Estas abordagens permitem que os usuários localizem rapidamente as informações necessárias, mesmo em meio a documentação extensa e complexa. As seguintes técnicas são geralmente utilizadas para identificar conceitos-chave e suas interligações dentro de um documento técnico:

2.2.1.1 Técnicas de *prompting*

A interação com LLMs ocorre por meio de *prompts*, que são instruções ou exemplos fornecidos ao modelo para direcionar suas respostas. O design eficaz de *prompts* influencia diretamente a qualidade, relevância e precisão das respostas. As principais abordagens de *prompting* incluem:

- a. *Zero-shot learning*: o modelo realiza uma tarefa sem exemplos prévios, testando sua capacidade de generalização. Por exemplo:

Entrada > Classifique o sentimento desta frase: "O dia está agradável."

Saída esperada > Sentimento: Positivo

Mesmo sem exemplos específicos sobre classificação de sentimentos, o modelo utiliza seu conhecimento linguístico para realizar a tarefa. Esta técnica pode ser particularmente útil para questões gerais sobre o SAGE, como "Explique o conceito de 'aquisição de dados' no contexto do sistema SAGE."

- b. *Few-shot learning*: fornecem-se alguns exemplos no prompt para guiar o modelo, útil em conceitos mais específicos ou técnicos (2). Por exemplo:

Entrada > Aqui estão alguns exemplos de poemas curtos sobre a natureza:

```
## EXEMPLO 1:  
O vento sussurra  
Segredos nas folhas verdes  
A dança da vida.
```

```
## EXEMPLO 2:  
O rio corre livre  
Refletindo o céu azul  
Calmaria e paz.
```

Agora, gere um novo poema:

Saída esperada >
A chuva cai suave
Lavando a alma da terra
Flores renascem.

Neste caso, o modelo recebe alguns exemplos de poemas sobre a natureza, o que o ajuda a entender o estilo e a temática desejada, facilitando a geração de um novo poema.

- c. *Chain-of-thought prompting (CoT)*: estimula o modelo a elaborar passos intermediários de raciocínio antes da resposta final, auxiliando na resolução de problemas complexos que exigem decomposição em etapas (3). Por exemplo:

Entrada (sem CoT)> Um caracol está no fundo de um poço de 10 metros de profundidade. Durante o dia, ele sobe 3 metros, mas à noite, ele desce 2 metros. Quantos dias o caracol leva para sair do poço?

Possível saída (sem CoT)> O caracol leva 10 dias para sair do poço.

Entrada (com CoT)> Um caracol está no fundo de um poço de 10 metros de profundidade. Durante o dia, ele sobe 3 metros, mas à noite, ele desce 2 metros. Quantos dias o caracol leva para sair do poço?

Pense passo-a-passo:

1. O caracol sobe 3 metros durante o dia e desce 2 metros à noite, então ele sobe $3 - 2 = 1$ metro por dia.
(...)

Possível saída (com CoT)> Certo, vamos avaliar por etapas:

1. O caracol sobe 3 metros durante o dia e desce 2 metros à noite, então ele sobe $3 - 2 = 1$ metro por dia.
2. Após 7 dias, o caracol terá subido $7 * 1 = 7$ metros.
3. No oitavo dia, o caracol sobe mais 3 metros, totalizando $7 + 3 = 10$ metros, o que significa que ele consegue sair do poço.

Resposta: o caracol leva 8 dias para sair do poço.

2.2.1.2 Fine-tuning (ajuste fino)

O *fine-tuning* ajusta os parâmetros de um LLM pré-treinado para um domínio específico, utilizando um conjunto de dados relevante. Essa adaptação aprimora a capacidade do modelo em compreender e gerar respostas adequadas ao contexto. Durante o processo, os parâmetros do modelo são atualizados, o que pode levar ao "esquecimento catastrófico", onde parte do conhecimento geral previamente adquirido é perdida.

Para mitigar limitações e melhorar a eficiência do *fine-tuning*, utilizam-se técnicas como o Parameter-Efficient Fine-Tuning (PEFT) (4). O PEFT reduz o número de parâmetros ajustáveis, mantendo ou melhorando o desempenho. Uma técnica comum é a LoRA (Low-Rank Adaptation), que adiciona matrizes de baixa ordem às camadas de atenção do modelo, permitindo ajustes com menos parâmetros treináveis e reduzindo requisitos de memória e computação (5).

O *fine-tuning* oferece vantagens como adaptação rápida para tarefas específicas, economia de recursos e aproveitamento de conhecimentos gerais em domínios especializados. No entanto, enfrenta desafios como a necessidade de conjuntos de dados significativos e de alta qualidade e o risco de *overfitting*, onde o modelo se torna excessivamente especializado, perdendo capacidade de generalização.

2.2.1.3 Geração Aumentada por Recuperação (RAG)

A Geração Aumentada por Recuperação (RAG) combina a recuperação de informações de um sistema de busca com a geração de texto de um LLM (6). Essa técnica permite que o assistente consulte uma base de dados e recupere trechos

relevantes para responder a perguntas, fornecer explicações e auxiliar em tarefas relacionadas. Algumas formas de implementar o RAG são:

- RAG básico (*naïve RAG*), que funciona em três etapas: (1) *recuperação*, onde o sistema seleciona trechos relevantes de uma base de conhecimento com base na consulta; (2) *contextualização*, onde os trechos recuperados são concatenados à consulta; e (3) *geração*, onde o modelo de linguagem gera uma resposta baseada na consulta enriquecida. Embora melhore a precisão e relevância das respostas, o RAG básico pode ser ineficiente para perguntas complexas sobre grandes bases de conhecimento, dependendo fortemente da eficácia do sistema de recuperação.
- RAG com *knowledge graphs*, que estende o RAG básico utilizando grafos para representar e recuperar informações. Envolve as etapas de (1) *preparação*, onde o conhecimento é representado como um grafo, com entidades como nós e relações como arestas; (2) *recuperação*, onde a consulta é utilizada para navegar e selecionar subgrafos relevantes; e (3) *contextualização e geração*, similares ao RAG básico, mas utilizando informações estruturadas do subgrafo. Este método captura relações complexas entre informações, permitindo raciocínio mais sofisticado e melhorando a coerência das respostas. No entanto, exige a construção e manutenção de um grafo, sendo mais intensivo computacionalmente e dependendo da qualidade da estrutura criada.

A escolha entre as abordagens de RAG depende da natureza da tarefa, da estrutura do conhecimento disponível e dos requisitos de desempenho.

3.0 - METODOLOGIA

3.1 Fine-tuning do modelo de linguagem

O objetivo inicial foi especializar modelos de linguagem para fornecer respostas precisas e detalhadas sobre a configuração do SAGE e protocolos amplamente utilizados na Eletrobras, como MMS (IEC 61850), DNP3 e ICCP.

3.1.1 Preparação do conjunto de dados para *fine-tuning*

Para realizar o *fine-tuning*, foi criado um conjunto de dados representativo. Foram coletadas informações detalhadas sobre a configuração do SAGE e dos protocolos a partir de manuais técnicos, normas oficiais e materiais de referência, visando cobrir aspectos técnicos essenciais e situações práticas enfrentadas por especialistas. Utilizando o modelo Gemini 1.5 Pro, foram geradas 10 perguntas e respostas para cada tópico identificado, incluindo estrutura das tabelas do SAGE, utilização de atributos específicos, relacionamentos entre entidades, exemplos práticos de configuração, resolução de problemas e procedimentos para manipulação de pontos e configurações. O processo resultou em mais de 800 pares de perguntas e respostas, abrangendo diversos assuntos relevantes para a configuração do SAGE e dos protocolos associados.

3.1.2 Avaliação e refinamento do conjunto de dados

Uma amostra das perguntas e respostas criadas foi submetida à avaliação de especialistas em SAGE da Chesf e de outras empresas do setor elétrico. Os especialistas avaliaram as respostas considerando critérios como precisão técnica, clareza, relevância e utilidade prática, atribuindo notas de 1 a 10 e fornecendo comentários. Ao total foram coletados 13 formulários de avaliação, totalizando 170 pares de perguntas e respostas avaliadas. O resultado revelou uma média de 8,6, indicando boa qualidade geral. Algumas respostas apresentaram notas inferiores, evidenciando inconsistências ou falta de clareza (**Tabela 1****Erro! Fonte de referência não encontrada.**). As perguntas e respostas com avaliações mais baixas foram revisadas ou removidas, garantindo que o conjunto de dados utilizado para o *fine-tuning* fosse consistente e de boa qualidade.

Pergunta	Resposta	Avaliação
Como configurar os parâmetros de failover (FAILP, FAILR, NFAIL, SFAIL) na tabela CXU?	* FAILP: Tempo em que o enlace principal permanece primário antes de um failover programado. * FAILR: Tempo em que o enlace reserva permanece primário antes de um failover programado. * NFAIL: Número máximo de tentativas de failover. * SFAIL: Tempo máximo para aguardar o sincronismo dos enlaces antes de realizar o failover.	1
Como a tabela CXU se relaciona com o modelo de dados do protocolo DNP3?	A tabela CXU define os parâmetros de comunicação para a conexão com a UTR ou centro de controle, que utiliza o protocolo DNP3 para a troca de dados.	5

Tabela 1 – Exemplo de avaliação de especialistas

3.1.3 Treinamento do modelo

Com o conjunto de dados preparado, o *fine-tuning* foi configurado com diferentes hiperparâmetros. Foram escolhidos os modelos GPT-4o-mini (OpenAI) e Gemini 1.5 Flash (Google) para os experimentos, devido à sua disponibilidade, baixo

custo e bom desempenho em tarefas gerais. Também foi utilizado o modelo de código aberto Llama 3.1 8B (Meta) em uma máquina local. Diversas combinações de hiperparâmetros foram testadas para identificar configurações eficazes:

- Dimensão dos adaptadores de camadas: ajustes no tamanho influenciando a capacidade de aprendizado.
- Tamanho do *batch*: variação do número de exemplos processados simultaneamente, buscando equilíbrio entre estabilidade e eficiência computacional.
- Número de épocas: diferentes quantidades de passagens completas pelo conjunto de dados para avaliar o impacto na convergência.
- Taxa de aprendizado: ajustes para evitar problemas como *overfitting* ou *underfitting*.

O *fine-tuning* foi conduzido para cada conjunto de hiperparâmetros, visando minimizar a função de perda e melhorar a capacidade dos modelos em gerar respostas corretas sobre o SAGE e os protocolos. Após cada treinamento, os modelos foram avaliados em um conjunto de validação para medir desempenho e capacidade de generalização. Como será visto na próxima seção, esta abordagem não apresentou os resultados esperados.

3.2 Utilização de janelas de contexto longas

A compreensão e processamento de informações por um modelo de linguagem dependem, além de sua arquitetura e tamanho, da extensão da janela de contexto que pode considerar ao gerar respostas. Janelas de contexto longas permitem que o modelo acesse mais informações em uma única interação, o que é particularmente útil em domínios complexos e técnicos como a configuração do sistema supervisório SAGE.

3.2.1 Uso de técnicas de *prompting* em contextos longos

Tradicionalmente, modelos como o GPT-4o possuem janelas de contexto limitadas a alguns milhares de *tokens* (até 128.000 *tokens*). Avanços recentes permitiram a criação de modelos com janelas de contexto significativamente maiores, como a família Gemini 1.5 do Google, que suporta até 2 milhões de *tokens*. Essa capacidade permite que o modelo acesse grandes volumes de informações detalhadas, como manuais técnicos, procedimentos operacionais e exemplos práticos, facilitando o entendimento de relacionamentos e dependências em sistemas complexos. Além disso, ao ter acesso a mais informações relevantes, o modelo tende a produzir respostas mais precisas e menos propensas a erros ou alucinações.

Após a avaliação positiva das perguntas e respostas iniciais obtidas via técnicas de *prompting*, e considerando os resultados dos testes com ajuste fino, o desenvolvimento do assistente focou em uma abordagem que aproveitasse ao máximo as técnicas de *few-shot prompting*, fornecendo ao modelo exemplos de perguntas e respostas para orientar sua geração de texto; *chain-of-thought*, estimulando o modelo a elaborar passos intermediários de raciocínio antes de fornecer a resposta final; e contexto longo, utilizando modelos como a família Gemini 1.5 para incorporar uma quantidade significativa de informações no *prompt*.

3.2.2 Criação de relações entre tabelas

Para incorporar as relações entre as tabelas do SAGE à base de conhecimento, foi criada uma descrição textual compacta dos diversos subsistemas e das relações entre as entidades. Embora o Gemini 1.5 seja um modelo multimodal e aceite tipos de dados diversos, como imagens de diagramas, a abordagem textual foi preferida para minimizar a latência e o uso de *tokens*, reduzindo custos. Utilizando o modelo Claude 3.5 Sonnet, da Anthropic, foram gerados códigos *mermaid* a partir de imagens de diagramas existentes. *Mermaid* é uma linguagem de marcação que permite criar diagramas de forma simples e textual (ver Figura 2). Os códigos gerados foram compilados em um único arquivo de referência, permitindo que o assistente tivesse acesso a uma representação textual e estruturada das relações entre tabelas, facilitando a compreensão e a explicação desses relacionamentos em suas respostas.

3.2.3 Refinamento do *prompt* de sistema e preparação dos dados

O desenvolvimento do assistente envolveu o refinamento do *prompt* de sistema e a preparação dos dados que comporiam a base de conhecimento. A base de conhecimento consistiu em:

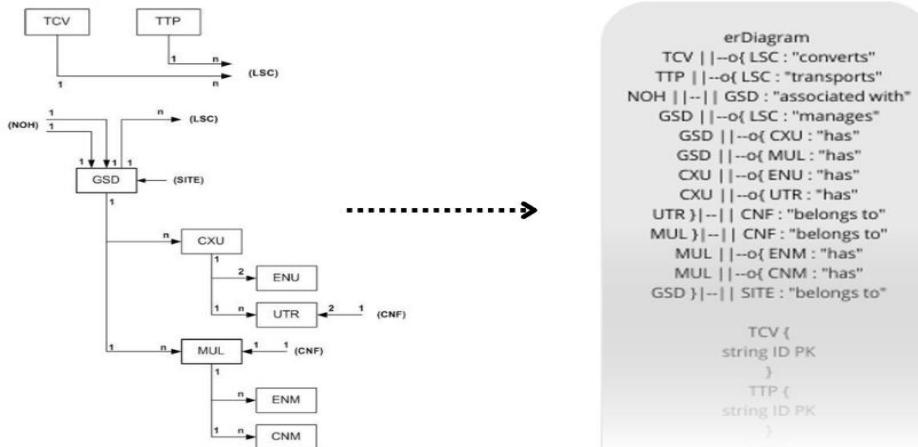


Figura 2 - Diagrama do Modelo de Comunicação do SAGE e sua versão em *mermaid*.

- Informações gerais sobre o sistema: contextualização sobre o SAGE, sua função e utilização no sistema elétrico.
- Configuração da base fonte: informações técnicas sobre a estrutura da base de dados e como configurá-la.
- Protocolos mais usados: especificações e configurações dos protocolos IEC 61850, DNP3 e ICCP.
- Relações entre tabelas: resultado do processo descrito em 3.2.2.
- Troubleshooting e experiência de especialistas: casos e soluções para problemas comuns e dicas práticas.
- Exemplos de configurações: casos práticos que ilustram configurações típicas ou complexas.
- Comportamentos e respostas desejadas: exemplos de como o assistente deve responder em diferentes situações, alinhando-o às expectativas dos usuários, utilizando técnicas de *few-shot prompting* e CoT.

3.2.4 Implementação de cache para redução de tokens

No código de testes do assistente, foram implementadas opções de cache para armazenar resultados de chamadas frequentes. Isso reduziu o fluxo de *tokens* em cada chamada à API, diminuindo custos e melhorando a eficiência. O cache permitiu que o assistente reutilizasse informações da base de conhecimento e do *prompt* de sistema, acelerando o tempo de resposta e otimizando o uso dos recursos disponíveis.

3.2.5 Processo iterativo de testes e ajustes

Com a base de conhecimento preparada, o desenvolvimento seguiu um ciclo iterativo:

1. Testes de desempenho: implementação do assistente utilizando o *prompt* refinado e inserindo a base de conhecimento na janela de contexto longa, além de utilizar o cache.
2. Avaliação das respostas: as respostas geradas foram avaliadas quanto à precisão, utilidade e alinhamento com o comportamento esperado.
3. Identificação de melhorias: foram identificados aspectos como ambiguidades, erros técnicos ou falta de profundidade nas respostas.
4. Ajustes no *prompt* e nos dados: o *prompt* de sistema foi refinado e informações foram adicionadas ou ajustadas conforme necessário.
5. Repetição do ciclo: o processo foi repetido até que os resultados atingissem um nível satisfatório.

No refinamento do *prompting*, priorizou-se manter o assistente focado no SAGE e em sua configuração, implementando também proteções contra *prompts* maliciosos. Isso foi alcançado por meio de instruções claras no *prompt* de sistema e exemplos de como lidar com solicitações inadequadas, garantindo que o assistente resistisse a tentativas de extração de informações impróprias ou geração de conteúdo inadequado.

3.3 Abordagens alternativas

Uma possível abordagem alternativa seria o uso de RAG com knowledge graphs, que estende a técnica RAG tradicional incorporando estruturas de grafos na recuperação de informações. No contexto do SAGE, essa abordagem poderia capturar eficazmente as intrincadas relações entre tabelas, protocolos e configurações. No entanto, desafios significativos foram identificados: complexidade da base de conhecimento; necessidade de múltiplas chamadas ao modelo, com custos computacionais elevados; e complexidade de implementação e manutenção da solução (7). Considerando os aspectos de latência, custos financeiros, complexidade de implementação e qualidade das respostas, optou-se por manter o uso das técnicas de *prompting* com janelas de contexto longas como abordagem preferencial. Esta decisão baseia-se na eficiência operacional, simplicidade, estabilidade e controle de custos oferecidos pela abordagem de *prompting design*.

4.0 - RESULTADOS E DISCUSSÃO

Esta seção apresenta os resultados do desenvolvimento do assistente inteligente para a configuração do SAGE, analisando sua eficácia, os desafios enfrentados e as implicações das técnicas utilizadas. Discute-se a qualidade das respostas, a eficiência operacional, a capacidade de generalização e compara-se as abordagens de *fine-tuning* e *prompting* com janelas de contexto longas.

4.1 Desempenho do assistente

A implementação de técnicas avançadas de *prompting*, aliada ao uso de janelas de contexto longas do modelo Gemini 1.5, foi fundamental para aprimorar a capacidade do assistente em responder a perguntas sobre a configuração do SAGE.

4.1.1 Qualidade das respostas

As respostas anteriormente mal avaliadas pelos especialistas foram novamente submetidas ao assistente, juntamente com novas interações mais complexas. Os resultados demonstraram uma melhora substancial na qualidade das interações, evidenciando a capacidade do assistente em esclarecer dúvidas e fornecer orientações detalhadas, facilitando o trabalho dos especialistas. O exemplo a seguir mostra a resposta melhorada a uma pergunta cuja avaliação original do especialista foi 5 (ver 3.1.2):

Usuário: Como a tabela CXU se relaciona com o modelo de dados do protocolo DNP3?

Assistente: A tabela CXU no SAGE, que representa a Conexão com Equipamentos de Aquisição e Controle (UTRs/CLPs), tem uma relação direta com o modelo de dados do protocolo DNP3. Vamos analisar como essa relação se manifesta:

1. Camadas de Comunicação:

O DNP3 trabalha com as camadas de aplicação, enlace e física. A CXU se relaciona principalmente com as camadas de enlace e física, definindo parâmetros que influenciam diretamente a comunicação entre o SAGE e as UTRs.

2. Parâmetros da CXU e DNP3:

- **AQANL:** Define o tempo em centésimos de segundo para aquisição periódica de dados analógicos (OBJ30), caso a CXU esteja relacionada a uma CNF configurada para DNP Level 0. Essa informação é crucial para determinar a frequência de varredura de dados analógicos no DNP3.
- **AQPOL:** Tempo em centésimos de segundo para verificação periódica do estado da UTR (OBJ50). Esse parâmetro influencia a frequência de polling words no DNP3, garantindo a manutenção da conexão com a UTR.
- **AQTOT:** Tempo em centésimos de segundo para integridade periódica de totalizadores (OBJ20). A CXU configura o intervalo de tempo para atualização de totalizadores no DNP3, garantindo a atualização do valor acumulado.
- **INTGR:** Tempo em centésimos de segundo para a leitura periódica de classe zero, caso a CXU esteja relacionada com uma Ligação de Aquisição. Isso define a frequência de coleta de eventos para transmissão em classes 1, 2 ou 3 no DNP3.
- **FAILP e FAILR:** Tempo em que o enlace principal e o enlace reserva permanecem primários antes do failover programado. Esses parâmetros influenciam a lógica de failover no DNP3, definindo quando o sistema deve alternar entre os enlaces.
- **NFAIL e SFAIL:** Definem o número máximo de tentativas de failover e o tempo máximo para aguardar o sincronismo das máquinas de estado dos dois enlaces. Esses parâmetros influenciam a estratégia de failover no DNP3, definindo quantas tentativas o sistema deve fazer para restabelecer a comunicação em caso de falha.

Em resumo, a tabela CXU no SAGE é uma ferramenta crucial para configurar os parâmetros da camada física e enlace do DNP3, garantindo o bom funcionamento da comunicação com as UTRs. A CXU, em conjunto com a CNF, define as características da comunicação com as UTRs, incluindo tempo de aquisição, polling, integridade, failover e outros parâmetros específicos do protocolo DNP3. Se tiver mais alguma dúvida sobre a configuração do SAGE com DNP3, me avise! 😊

4.1.2 Capacidade de generalização

O assistente mostrou habilidade em responder a perguntas não previstas nos exemplos fornecidos, evidenciando boa capacidade de generalização devido ao refinamento do *prompt* de sistema e à ampla base de conhecimento incorporada. Isso demonstra que o assistente pode aplicar conhecimentos a novas situações, auxiliando em diversas tarefas.

Usuário: preciso de ajuda para configurar o opmsk de um IED.

Assistente: Com prazer, posso te ajudar a configurar o OPMSK do seu IED! 😊

Para te dar uma mãozinha, preciso de algumas informações. Me fale sobre a sua configuração:

1. O IED suporta a criação de datasets dinâmicos pelo SAGE? Se sim, podemos desligar o bit responsável por essa funcionalidade no OPMSK.
2. Você está utilizando GOOSE para receber dados do IED? Se não, podemos desligar o bit relacionado ao GOOSE.
3. Como estão configurados os Report Control Blocks do seu IED? Eles estão configurados individualmente em cada Logical Node, ou estão centralizados no LLN0? Essa informação é importante para configurar o bit que define qual RCB o SAGE deve utilizar.
4. Você prefere que o SAGE receba os dados do IED apenas quando houver mudança no valor do ponto, ou também quando houver mudança na qualidade do dado? Essa preferência define o trigger dos reports e, consequentemente, o bit correspondente no OPMSK.

5. Você deseja forçar a checagem de intertravamento independente das configurações de CGF?
 6. O IED é redundante? Se sim, o bit de redundância no OPMSK deverá ser ligado.
 7. Por último, você deseja que o IED envie os identificadores das variáveis nos Reports?
- Com base nessas informações, podemos definir quais bits do OPMSK devem ser ligados ou desligados para atender a sua necessidade! 🤝

Aguardo pelas suas respostas para que possamos configurar o OPMSK! 😊

Usuário: o IED não suporta datasets dinâmicos. não estou usando goose. os reports não estão no LLN0. quero que os dados sejam reportados na mudança de valor ou qualidade. o IED não é redundante. habilite os ids nos reports e force o intertravamento. também ligue o bit 15.

Assistente: Maravilha, você me deu todas as informações que eu precisava! 🎉

Com base no que você me disse, a configuração do OPMSK para o seu IED ficaria assim:

- Bit 4 (datasets dinâmicos): Ligado, pois o IED não suporta datasets dinâmicos. O SAGE terá que usar os datasets pré-configurados no IED.
- Bit 5 (Report Control Blocks no LLN0): Desligado, já que os reports não estão configurados no LLN0.
- Bit 6 (ativação de GOOSE): Desligado, pois você não está utilizando GOOSE.
- Bit 8 (uso de InterLockCheck): Ligado, para forçar a checagem de intertravamento independente do KCONV.
- Bit 9 (trigger dos reports): Desligado, para que o SAGE receba dados quando houver mudança no valor ou qualidade do ponto.
- Bit 12 (redundância): Desligado, pois o IED não é redundante.
- Bit 15 (time-tags inválidos): Ligado, o SAGE deve ignorar os atributos de invalidade em time-tags.
- Bit 20 (opção de DataReference no Report Control Block): Ligado, o IED deve enviar a identificação dos pontos nos Reports.
- Demais bits: Desligados por padrão.

Representando em binário, a máscara ficaria: 0001 0000 1000 0001 0001 0000

Este valor binário corresponde ao valor hexadecimal 108110. Basta substituir <OPMSK> no seu arquivo CNF.dat por 108110.

Após a alteração, salve o arquivo e execute **AtualizaBD fria fonte** para atualizar a base de dados do SAGE. Pronto! 🎉 Com essa configuração, o SAGE deve se comunicar com o seu IED utilizando as opções que você definiu. Se tiver mais alguma dúvida, pode me perguntar. 😊

4.2 Aspectos adicionais da solução

O pré-processamento das informações reduziu em 35% o consumo de *tokens*, mantendo a qualidade das respostas. Ao focar em informações essenciais e eliminar redundâncias, o assistente tornou-se mais eficiente. A utilização de um LLM para gerar códigos *mermaid* a partir de diagramas facilitou a representação textual das relações entre tabelas, aprimorando a compreensão estrutural pelo assistente. Já a utilização de caching resultou em uma redução de 20% no tempo de resposta, e o menor volume de *tokens* nas chamadas à API diminuiu significativamente os custos operacionais.

4.3 Alinhamento temático e segurança

O refinamento do *prompt* de sistema e a inclusão de exemplos garantiram que o assistente mantivesse foco no SAGE, atendendo às necessidades dos usuários. Foram implementadas salvaguardas básicas para evitar a divulgação de informações sensíveis e fornecer respostas adequadas a solicitações inapropriadas, fortalecendo a segurança e a confiabilidade do sistema.

4.4 Comparação com outras abordagens

Os experimentos de *fine-tuning* não atingiram as expectativas, com modelos ajustados fornecendo respostas imprecisas e incoerentes. A Tabela 2 mostra testes realizados com os modelos Gemini 1.5 Flash do Google e GPT-4o-mini da OpenAI.

Modelo	Configuração de <i>fine-tuning</i>	Exemplo de Interação	
		Usuário	Assistente
gemini-1.5-flash	Dados de treinamento: +1700 exemplos Epochs: 1 Batch size: 20 Learning rate: 0,0002 Duração do processo: 46m56s	O que é um ponto PDF?	Um ponto PDF (Ponto de Referência de Distribuição) é um ponto lógico que representa um ponto físico de distribuição de energia elétrica. Ele é utilizado para calcular o valor total de energia distribuída em um determinado nível de tensão.
gpt-4o-mini	Dados de treinamento: +1700 exemplos Epochs: 4 Batch size: 1 Learning rate: 0,1 Duração do processo: 4,5h	O que é um ponto PDF no contexto do SAGE?	No contexto do SCASE (Sistema de Apoio à Gestão de Emergências), um ponto PDF (Ponto Digital Físico) representa um dado digital que é recebido de um equipamento de campo, como um sensor ou um atuador. Os pontos PDF são utilizados para agragar e organizar dados digitais, que podem ser utilizados para gerar alarmes, controlar equipamentos ou serem apresentados em telas de supervisão.

Tabela 2 - *Fine-tuning* dos modelos Gemini 1.5 Flash e GPT-4o-mini, e testes do resultado com uma pergunta simples

Possíveis causas para este desempenho incluem:

- Volume de dados insuficiente: os 1700 pares de perguntas e respostas podem ser inadequados para treinar modelos de grande escala em domínios complexos.
- *Overfitting*: modelos excessivamente ajustados aos dados de treinamento, perdendo a capacidade de generalização.
- Complexidade do domínio: a especificidade técnica do SAGE e dos protocolos pode exigir conjuntos de dados maiores e mais diversificados.
- Limitações dos modelos: o acesso limitado aos modelos GPT-4o-mini e Gemini 1.5, com *fine-tuning* realizado via interfaces simplificadas, pode ter influenciado os resultados. Testes com o modelo de código aberto Llama 3.1 da Meta não demonstraram melhorias.

Em contrapartida, as técnicas utilizando *prompts* detalhados e janelas de contexto longas produziram resultados significativamente melhores. As respostas foram mais precisas e coerentes, beneficiando-se do conhecimento pré-treinado do modelo e da capacidade de incorporar informações contextuais, permitindo ajustes rápidos sem a necessidade de novos treinamentos.

Avaliou-se também o uso do GraphRAG (8) (9) como alternativa. Embora ofereça vantagens na representação de relações complexas, desafios como a complexidade da base de conhecimento e o aumento de custos e tempo de resposta devido a múltiplas chamadas ao modelo limitaram sua viabilidade. Dado que o assistente atende principalmente especialistas com perguntas complexas, optou-se por manter as janelas de contexto longas, evitando complexidade adicional sem ganhos significativos de eficiência.

4.5 Limitações identificadas

Apesar dos resultados positivos, algumas limitações foram observadas:

- Dependência de recursos computacionais: o método depende das janelas de contexto de 2 milhões de *tokens* dos modelos Gemini, exigindo infraestrutura robusta via API do Google. Até agosto de 2024, não havia modelos similares disponíveis.
- Base de conhecimento: o desempenho do assistente está vinculado à qualidade da base de conhecimento, que precisa ser atualizada e adaptada conforme necessário.
- Alucinações: embora o uso de um longo contexto reduza a possibilidade de erros, o modelo ainda pode produzir alucinações, especialmente em situações que exigem maior capacidade de generalização.

4.6 Repositório disponível

Materiais suplementares, incluindo resultado da avaliação dos especialistas, conjunto de dados gerado, notebooks Python utilizados para *fine-tuning* e exemplos estendidos de respostas do assistente, estão disponíveis em um repositório público do GitHub (10). Esses recursos fornecem detalhes adicionais e permitem a reprodução de alguns experimentos realizados neste estudo.

5.0 - CONCLUSÃO

O desenvolvimento do assistente inteligente para a configuração do SAGE, utilizando modelos de linguagem de grande escala e técnicas avançadas de processamento de linguagem natural, mostrou-se promissor para enfrentar os desafios na configuração da base de dados e assistência ao usuário. O assistente oferece orientações claras e precisas, auxiliando especialistas em tarefas complexas e na depuração de erros, reduzindo a necessidade de consultar manuais extensos. Além disso, contribui para prevenir erros comuns na configuração, melhorando a confiabilidade do sistema e diminuindo o tempo gasto em correções. Como ferramenta educacional, facilita o aprendizado de novos especialistas em SAGE por meio de explicações e orientações, aumentando a eficiência e produtividade das equipes de engenharia e operação graças à agilidade nas respostas e ao acesso imediato a informações relevantes.

O projeto também evidenciou desafios e proporcionou lições importantes. As tentativas de ajuste fino foram ineficazes devido ao volume insuficiente de dados e à complexidade do domínio, indicando a necessidade de estratégias alternativas. Desenvolver técnicas de destilação e otimização de conteúdo foi essencial para fornecer informações ao modelo sem comprometer a eficiência, podendo ser aplicado em projetos similares. A implementação de salvaguardas contra *prompts* maliciosos ressaltou a importância de considerar aspectos de segurança e ética no desenvolvimento de assistentes inteligentes.

5.1 Perspectivas futuras

Os resultados apontam diversas oportunidades de expansão e aprimoramento. A integração com modelos de IA mais avançados pode aumentar a eficácia do assistente. Além da configuração da base SCADA e alguns protocolos, o assistente pode ser aprimorado para suportar funções EMS, outros protocolos e tarefas como instalação e configuração de telas. A

metodologia pode ser replicada em outros sistemas supervisórios ou áreas que demandem suporte técnico especializado, ampliando o impacto da solução. Embora as tentativas iniciais não tenham sido bem-sucedidas, explorar o uso de *fine-tuning* para melhorar a segurança e o desempenho do modelo é uma possibilidade. Além disso, incorporar técnicas avançadas de RAG pode reduzir custos em perguntas simples e melhorar a eficiência do assistente.

5.2 Considerações finais

Conclui-se que modelos de linguagem avançados são eficazes na criação de assistentes inteligentes em domínios técnicos complexos (11). O assistente para o SAGE demonstrou potencial para melhorar a eficiência operacional e apoiar o treinamento de profissionais. Apesar dos desafios, existem oportunidades significativas para aprimoramento e expansão, indicando um futuro promissor para a inteligência artificial na automação de sistemas supervisórios. Investir em pesquisa e desenvolvimento em IA aplicada, considerando aspectos técnicos, éticos e humanos, contribuirá para que sistemas complexos sejam gerenciados com maior facilidade, segurança e eficácia, beneficiando empresas e a sociedade como um todo.

6.0 - REFERÊNCIAS BIBLIOGRÁFICAS

- (1) Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. Attention is All You Need. *Advances in Neural Information Processing Systems*. 2017.
- (2) Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*. 2020.
- (3) Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., ... & Le, Q. V. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv preprint*. 2022, Vol. arXiv:2201.11903.
- (4) Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., de Laroussilhe, Q., Gesmundo, A., ... & Gelly, S. Parameter-Efficient Transfer Learning for NLP. *Proceedings of the 36th International Conference on Machine Learning*. 2019.
- (5) Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, L., ... & Chen, W. LoRA: Low-Rank Adaptation of Large Language Models. *arXiv preprint*. arXiv:2106.09685, 2021.
- (6) Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Riedel, S. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems*. 33, 2020.
- (7) Ji, S., Pan, S., Cambria, E., Marttinen, P., & Yu, P. S. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Transactions on Neural Networks and Learning Systems*. 33, 2021, Vol. 2.
- (8) Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Jonathan Larson. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *arXiv:2404.16130*. 2024.
- (9) Microsoft. A modular graph-based Retrieval-Augmented Generation (RAG) system. *Github*. [Online] 2024. <https://github.com/microsoft/graphrag>.
- (10) Araújo, Pablo Mascarenhas de. Assistente Inteligente para Configuração do SAGE (Material Suplementar). *Github*. [Online] <https://github.com/pascarujo/assistente-sage-xvii-stpc>.
- (11) Bommasani, R., Dai, A. M., Altman, R., Babuschkin, I., et al. On the Opportunities and Risks of Foundation Models. *arXiv preprint*. arXiv:2108.07258, 2021.

7.0 - DADOS BIOGRÁFICOS



PABLO MASCARENHAS DE ARAÚJO é formado em Eng. Elétrica com ênfase em Eletrônica pela Universidade Federal de Pernambuco em 2002, Especialista em Proteção de Sistemas Elétricos pela UFRJ em 2005, com MBA em Gerenciamento de Projeto pela Fundação Getúlio Vargas em 2023 e Executive Certification on Artificial Intelligence: Business Strategies and Applications pela UC Berkeley em 2024. Trabalha na Chesf desde 2002 no planejamento e implantação de projetos de SPCS, teleproteção e segurança cibernética em tecnologias operacionais. A partir de 2022 atuou como gerente de projeto para a formular e implantar o Plano de Digitalização das Subestações da Chesf (PDIG). Desde o segundo semestre de 2023 passou a integrar a equipe de Gestão de Valor de Ativos da Transmissão da Eletrobras.