
SmartDocAI: Consulta e Análise de Documentos PDF pelo WhatsApp com Suporte de LLMs

Vinicius Pires Barreto



UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE COMPUTAÇÃO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

Monte Carmelo - MG
2024

Vinicius Pires Barreto

**SmartDocAI: Consulta e Análise de
Documentos PDF pelo WhatsApp com Suporte
de LLMs**

Trabalho de Conclusão de Curso apresentado
à Faculdade de Computação da Universidade
Federal de Uberlândia, Minas Gerais, como
requisito exigido parcial à obtenção do grau de
Bacharel em Sistemas de Informação.

Área de concentração: Sistemas de Informação

Orientador: Prof. Dr. Adriano Mendonça Rocha

Coorientador: Prof. Dr. Silvio Ereno Quincozes

Monte Carmelo - MG

2024

Este trabalho é dedicado aos meus pais, que dedicaram suas vidas ao meu sucesso e foram meu apoio em cada etapa da minha jornada acadêmica. Essa conquista é, acima de tudo, fruto do amor e da dedicação de vocês.

Agradecimentos

Agradeço primeiramente a Deus, pela força, sabedoria e saúde concedidas ao longo de toda esta jornada acadêmica. A Ele, que guiou meus passos e iluminou o caminho em todos os momentos, dedico minha gratidão e reconhecimento.

Aos meus pais, cujo apoio incondicional e incentivo constante foram fundamentais para que eu pudesse chegar até aqui. Obrigado por acreditarem em mim, por estarem ao meu lado em cada etapa, e por todos os sacrifícios feitos para que eu pudesse seguir em frente e alcançar este objetivo.

Aos professores, que, com dedicação e paciência, transmitiram não apenas conhecimentos técnicos, mas também valores e lições que levarei para a vida. Cada um de vocês contribuiu de maneira única e essencial para a minha formação.

Agradeço também a todos os amigos que fiz durante o curso, que foram de suma importância para minha formação também, me ajudando sempre em momentos difíceis e realizando o curso juntos.

Por fim, aos meus orientadores, cujos conselhos, dedicação e incentivo foram essenciais para o desenvolvimento deste trabalho. Agradeço pela confiança e por cada orientação.

“O sucesso é a soma de pequenos esforços repetidos dia após dia.”
(Robert Collier)

Resumo

A dificuldade de buscar informações precisas em documentos extensos tornou-se um problema recorrente em diversos contextos, como editais e regulamentos. Para abordar essa questão, este trabalho propõe o desenvolvimento da ferramenta *SmartDocAI*, que utiliza de Modelo de Linguagem de Grande Escala (LLM) para facilitar o acesso rápido a informações específicas em documentos textuais volumosos. A abordagem proposta visa a implementação de uma integração da Interface de Programação de Aplicações (API) do modelo GPT-3.5-Turbo juntamente com o aplicativo de mensagens *WhatsApp*. O usuário interage com a ferramenta por meio de uma conversa no *WhatsApp* e obtém informações presentes nos documentos por meio das respostas fornecidas pela *SmartDocAI*. Para avaliar a ferramenta proposta, foi implementado um sistema de avaliação automática, onde o usuário avalia cada resposta fornecida pela *SmartDocAI*, considerando os quesitos clareza e satisfação das respostas. Também foi realizado uma avaliação geral do sistema após cinco avaliações individuais realizadas pelo usuário, considerando os critérios economia de tempo e chance de uso futuro. Por fim, foi realizado uma avaliação pelos autores das respostas avaliadas de forma insatisfatórias pelos usuários. Na avaliação dos usuários, no quesito satisfação, 82,4% avaliaram a ferramenta positivamente, 82,6% dos usuários acharam as respostas da *SmartDocAI* claras. Nas avaliações gerais, 81,8% dos usuários avaliaram que a ferramenta economiza tempo e 90,9% disseram que voltariam a utilizar a *SmartDocAI* no futuro. Após avaliação dos autores nas respostas consideradas insatisfatórias pelos usuários, constatou-se que 57,1% delas foram perguntas fora do contexto do documento analisado pela ferramenta. Os resultados deste estudo mostram a viabilidade da *SmartDocAI* em interpretar e responder a consultas realizadas por usuários em documentos textuais extensos.

Palavras-chave: ChatBot, GPT, LLM, WhatsApp, Documentos Textuais Extensos.

Lista de ilustrações

Figura 1 – Fluxo que a aplicação deve seguir.	20
Figura 2 – Diagrama representando todo o fluxo de avaliações implementado. . . .	22
Figura 3 – Figura de exemplo contendo a pergunta do usuário para a ferramenta, a resposta gerada pelo modelo GPT e as avaliações individuais.	31
Figura 4 – Figura de exemplo contendo as duas perguntas de avaliações gerais disparadas pela <i>SmartDocAI</i>	31
Figura 5 – Figura de exemplo contendo a pergunta do usuário para a ferramenta, a resposta gerada pelo modelo Transformadores Pré-Treinados Gene- rativos (GPT) e as avaliações individuais que foram utilizadas para análise na Seção 5.1.2.	33
Figura 6 – Gráfico contendo todas as métricas referentes a primeira avaliação in- dividual.	34
Figura 7 – Gráfico contendo todas as métricas referentes a segunda avaliação in- dividual.	34
Figura 8 – Gráfico contendo todas as métricas referentes a primeira avaliação geral.	35
Figura 9 – Gráfico contendo todas as métricas referentes a segunda avaliação geral.	35

Lista de siglas

API Interface de Programação de Aplicações - *Application Programming Interface*

GPT Transformadores Pré-Treinados Generativos - *Generative Pre-trained Transformers*

HTTP Protocolo de Transferência de Hipertexto - *Hypertext Transfer Protocol*

IA Inteligência Artificial

LLM Modelo de Linguagem de Grande Escala - *Large Language Model*

PDF Formato Portátil de Documento - *Portable Document Format*

PLN Processamento de Linguagem Natural

RNN Rede Neural Recorrente - *Recurrent Neural Network*

Sumário

1	INTRODUÇÃO	10
1.1	Motivação	10
1.2	Problema	11
1.3	Hipótese	11
1.4	Objetivos	12
1.5	Organização da Monografia	12
2	FUNDAMENTAÇÃO TEÓRICA	13
2.1	Conceitos Básicos	13
2.1.1	ChatBot	13
2.1.2	Processamento de Linguagem Natural	13
2.1.3	Modelos de Linguagem de Grande Escala	14
2.1.4	Arquitetura <i>Transformer</i>	14
3	TRABALHOS RELACIONADOS	16
3.1	Discussão	17
4	ABORDAGEM PROPOSTA	19
4.1	Arquitetura do Sistema	19
4.1.1	Módulos	19
4.1.2	Funcionamento dos módulos em conjunto	22
4.1.3	Modelagem e Estrutura de Dados	23
4.2	Implementação da <i>SmartDocAI</i>	24
4.2.1	Implementação do módulo Extração de Dados de Arquivos	25
4.2.2	Implementação do módulo de integração com o WhatsApp	25
4.2.3	Implementação do módulo de Processamento das Mensagens	26
4.2.4	Implementação do módulo de Integração com o modelo <i>GPT-3.5-turbo</i>	27
4.3	Execução	28

5	MÉTODO E ANÁLISE DOS RESULTADOS	29
5.1	Método para a Avaliação	29
5.1.1	Avaliação dos usuários	29
5.1.2	Avaliação das respostas com base no conteúdo fornecido	32
5.2	Avaliação dos Resultados	33
5.2.1	Resultados obtidos pelos usuários da <i>SmartDocAI</i>	33
5.2.2	Resultados obtidos através da análise das respostas dadas pela <i>Smart-DocAI</i>	35
6	CONCLUSÃO	37
6.1	Principais Contribuições	37
6.2	Trabalhos Futuros	38
6.3	Contribuições em Produção Bibliográfica	38
	REFERÊNCIAS	39

Introdução

Este trabalho propõe a criação da ferramenta *SmartDocAI*, uma solução que utiliza modelos de Inteligência Artificial (IA) integrado com aplicativo de mensagens para simplificar o acesso a informações em documentos textuais extensos. Em muitos casos, a busca manual de informações específicas em documentos de múltiplas páginas demanda tempo e esforço significativos. A *SmartDocAI* é desenvolvida para otimizar esse processo, aplicando Modelo de Linguagem de Grande Escala(LLM) para interpretar e extrair respostas rápidas e precisas, especialmente em contextos onde o conteúdo é extenso e detalhado.

Com o crescimento expansivo das buscas por *chatbots* e ferramentas de IA(DAM et al., 2024), este trabalho se torna muito relevante, seguindo as tendências e aplicando os conceitos de *chatbots* e LLM para criar uma ferramenta que utiliza IA para interpretar quantidades abundantes de texto.

1.1 Motivação

Nos anos recentes, o uso crescente de *chatbots* integrado com IA vem se tornando cada vez mais evidente (LOKMAN; AMEEDDEEN, 2019), e muito utilizado para diversas aplicações, sendo útil para processar e interpretar abundância de informações textuais e entregar precisamente a necessidade do usuário.

Normalmente, sistemas que utilizam LLM para auxiliar e fornecer informações em um contexto específico, existe muita limitação com o conteúdo a ser procurado, já que o modelo pode não ter acesso a certas informações, por exemplo, no contexto de arquivos de regulamentos e editais, o modelo não tem um acesso prévio aquelas informações, sendo muitas vezes impossível de responder à solicitação do usuário de forma correta e precisa.

Com o uso de linguagem natural, palavras procuradas diretamente nos arquivos textuais extensos com a gramática errada ou um sinônimo podem ser corrigidas utilizando Processamento de Linguagem Natural (PLN) (NADKARNI; OHNO-MACHADO; CHAPMAN, 2011), que traz uma segurança e economia de tempo se comparado ao método de busca tradicional.

1.2 Problema

Como já citado, a IA juntamente com *chatbots* vem sendo utilizada em diversos contextos diferentes, mas há uma falta de utilizar este mesmo modelo para facilitar a busca de informações em documentos extensos, e também nota-se que disponibilizar esta ferramenta em um sistema de mensagens amplamente usada, acessível à grande parte da população é um grande desafio.

A busca por informações específicas em documentos com várias páginas ou conteúdo detalhado, como editais e regulamentos, pode ser um processo demorado e desgastante, especialmente quando realizada manualmente. Essa dificuldade é ainda mais evidente quando o usuário não tem conhecimento técnico para localizar informações com precisão em textos volumosos ou não tem domínio das palavras-chave referentes ao documento para realizar a busca, consumindo bastante tempo e esforço do usuário que busca informações específicas nos documentos textuais.

De acordo com Statista (2024), no Brasil, 99% dos dispositivos móveis têm o aplicativo de mensagens *WhatsApp* instalado, tornando interessante realizar a implementação deste serviço na plataforma, já que, de acordo com Statista (2024), há mais de 147 milhões de usuários deste aplicativo no Brasil.

1.3 Hipótese

A hipótese central deste trabalho é que a integração de um *chatbot*, baseado em um LLM como o GPT-3.5-turbo, a uma plataforma amplamente acessível como o *WhatsApp*, proporciona um acesso mais eficiente e preciso a informações contidas em documentos institucionais extensos, como editais e regulamentos.

As questões de pesquisa para validar essa hipótese incluem:

- ❑ O *chatbot* baseado em LLM consegue fornecer respostas precisas e relevantes a consultas específicas e complexas sobre o conteúdo dos documentos?
- ❑ Os usuários têm uma percepção de uma economia de tempo ao usar o SmartDocAI, em comparação com métodos manuais de busca?
- ❑ A experiência geral dos usuários, em termos de satisfação, é positiva ao utilizar o SmartDocAI para obter informações rápidas e diretas?

A confirmação desta hipótese será realizada por meio de experimentos que avaliarão o desempenho do *chatbot* em termos de precisão, clareza e economia de tempo nas respostas, com base nas avaliações dos usuários quanto à eficiência e relevância da ferramenta.

1.4 Objetivos

O objetivo geral deste trabalho consiste na implementação de um *chatbot* no aplicativo *WhatsApp* integrado com um modelo de inteligência artificial GPT para auxiliar instituições a servir documentos extensos neste chatbot para pessoas interessadas, de forma que entregue informações precisas e rápidas de forma superior do que métodos tradicionais de busca de informações nestes documentos.

Os objetivos específicos são listados a seguir:

- ❑ Realizar a integração da ferramenta *SmartDocAI* com o aplicativo de mensagens *WhatsApp*.
- ❑ Utilizar o modelo GPT-3.5-turbo para processar e interpretar as perguntas dos usuários feitas através do *WhatsApp*.
- ❑ Avaliar todas as respostas obtidas pelo modelo GPT-3.5-turbo.
- ❑ Documentar e disponibilizar todo o processo e código realizado neste trabalho.

1.5 Organização da Monografia

A organização desta monografia é a seguinte: O Capítulo 2 aborda conceitos para o entendimento do trabalho e trabalhos semelhantes ao presente neste documento. O Capítulo 3 detalha a metodologia aplicada para a arquitetura do sistema, integrações com ferramentas externas, implementação da ferramenta como um todo e também a execução da ferramenta criada. No Capítulo 4 é abordado a metodologia para realizar a avaliação da ferramenta aplicada em um cenário real e também a exploração dos resultados obtidos através destas avaliações. No Capítulo 5 é apresentado a conclusão do trabalho realizado, destacando as principais contribuições, limitações e também trabalhos futuros que podem ser feitos a partir deste trabalho.

Fundamentação Teórica

2.1 Conceitos Básicos

Neste capítulo, serão abordados alguns conceitos básicos cruciais para o entendimento do trabalho desenvolvido.

2.1.1 ChatBot

De acordo com Cahn (2017), *ChatBot* é um sistema de diálogo entre o humano e o computador que utiliza linguagem natural para manter a comunicação e pode ser classificado com diversos tipos, incluindo baseado em regras e modelo generativo.

Nos *chatbots* baseados em regras, o sistema escolhe a resposta baseado em predefinições de diversas regras e no reconhecimento das perguntas pelo usuário, sendo incapaz de criar novas respostas. O conhecimento usado nos *chatbots* são todos criados pelo desenvolvedor, por este método ser baseado em regras, se houver regras que não são cobertas pelo programador, como erros de gramática ou uma pergunta feita de forma diferente, este tipo de *chatbot* pode acabar não atendendo corretamente a solicitação do usuário (ADAMOPOULOU; MOUSSIADES, 2020). Diferente do generativo, a grande diferença é que é necessário prever todas as possíveis possibilidades que o usuário pode inserir.

No tipo de *chatbot* generativo (ADAMOPOULOU; MOUSSIADES, 2020), o modelo gera as respostas baseado na pergunta atual do usuário e das outras mensagens anteriores, ele utiliza *machine learning* e *deep learning* para lidar com erros ortográficos, situações não programadas e gerar respostas inéditas.

2.1.2 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN) se refere a um subcampo da IA (KHURANA et al., 2023) que possui como foco o entendimento da linguagem humana por computadores. O principal objetivo do PLN é dar a capacidade das máquinas de compreender,

interpretar e gerar texto como os humanos fazem. Para isto, é necessário desenvolver algoritmos e modelos que consigam processar gigantes volumes de dados de texto, podendo extrair significados e padrões de forma eficiente. O PLN precisa desenvolver diversas tarefas complexas, dentre elas estão a análise gramatical e sintática, extração de informações, reconhecimento de voz, classificação de textos, tradução automática, análises de sentimentos e também a geração de linguagem natural como saída do sistema. Essas tarefas podem parecer fáceis, porém, para máquinas que não têm capacidade de compreensão de sentimentos nem mesmo de significado de cada coisa, pode ser extremamente complexo, dado a complexidade da mente humana. O PLN é essencial para a criação de sistemas inteligentes que interagem com humanos por meio de textos.

2.1.3 Modelos de Linguagem de Grande Escala

Os Modelos de Linguagem de Grande Escala (Large Language Models - LLMs) fazem parte dos modelos de inteligência artificial feitos para processar, compreender e gerar linguagem natural em larga escala. São baseados em arquiteturas avançadas de redes neurais e são treinados utilizando uma grande quantidade de dados texturais com muitos parâmetros, que permite realizar diversas tarefas relacionadas à linguagem. Os modelos recebem o nome de Grande Escala por causa da quantidade de dados utilizados no seu treinamento e também ao número de parâmetros do modelo, que gera essa capacidade de aprendizado sem precedentes (IBM, 2024). Como exemplo, podemos citar o modelo *GPT-3.5*, que foi treinado com uma quantidade muito grande de dados textuais e possui mais de 175 bilhões de parâmetros.

2.1.4 Arquitetura *Transformer*

A arquitetura *Transformer* (VASWANI et al., 2023) é um modelo neural que foi feito para considerar dependências de longo prazo em sequência de dados, por exemplo, uma informação fornecida muito antes em uma potencial conversa pode ser um fator determinando para um raciocínio nos dados recebidos no presente momento. Esse modelo utiliza mecanismos de atenção para processar e analisar contextos, podendo assim compreender relações entre as palavras e frases. Essa capacidade de analisar o contexto de forma paralela diferencia o modelo *Transformer* de outras abordagens, como Rede Neural Recorrente (RNN), que apresentam dificuldades ao lidar com essas dependências ao longo prazo.

2.1.4.1 Modelos de Linguagem GPT

Os Modelos de Linguagem GPT (Generative Pre-Trained Transformer) vêm de uma grande evolução no campo de PLN e uma direta aplicação da arquitetura *Transformer*.

Estes modelos foram feitos para resolver diversas tarefas que são relacionadas à linguagem, como exemplo pode-se citar tradução, resumos de textos, respostas a perguntas, geração de conteúdos, entre outras. O termo *Generative Pre-Trained Transformer* tem por objetivo citar que os modelos são pré-treinados com uma grande quantidade de dados, fornecendo o modelo todo o conhecimento necessário para gerar texto de forma autônoma e coerente (SERVICES, 2024).

Trabalhos Relacionados

Existem diversas propostas recentes que utilizam de inteligência artificial para integrar com sistemas de mensagens que são usados no dia a dia, explorando estas propostas é possível encontrar o exemplo em Putri Ariatna Alia, Johan Suryo Prayogo, Rony Kriswibowo, Agung Teguh Setyadi (2024), onde foi implementado uma aplicação utilizando as linguagens de programação *JavaScript* e *C++*, para realizar a integração entre o aplicativo do *WhatsApp* e a tecnologia *Generative Pre-Trained Transformer* (GPT) como forma de facilitar a busca por informações ao usuário, utilizando uma ferramenta em que o usuário já utiliza no dia a dia, para buscar informações voltadas para a saúde. Neste modelo, a implementação realizada trabalha apenas com os dados já disponíveis no modelo, não havendo a possibilidade de inserir novos conteúdos para utilizar como contexto.

Gisnaya Faridatul Avisyah, Ivandi Julatha Putra, Sidiq Syamsul Hidayat (2023) utiliza o algoritmo GPT-3 (*Generative Pre-Trained Transformer*) integrado ao Telegram via Python para criar um sistema de mensagens automatizado mais eficiente, capaz de lidar com múltiplas línguas e fornecer respostas ampliadas, focado em informações sobre a Universidade Politeknik Negeri Semarang. O método que o autor utiliza neste trabalho é puramente utilizando os dados disponíveis nos modelos GPT, sem dar possibilidade de inserção de novas informações sobre a universidade, limitando-se apenas ao conhecimento já disponível.

Em Jonatan e Igor (2023) é proposto uma arquitetura de sistema de mensagens automatizadas parecido com o implementado neste trabalho, tendo sua interface do *WhatsApp* para o usuário, uma API para conectar com a integração realizada neste trabalho e uma inteligência artificial baseada em processamento de linguagem natural para gerar as respostas ao usuário. Apesar da arquitetura parecida, no presente trabalho, foi implementada a camada de integração entre o *WhatsApp* e a IA, o *JavaScript*, pela facilidade de integração com o *WhatsApp* utilizando bibliotecas do *JavaScript* em poucas linhas, diferente do *Python* proposto pelo Jonatan e Igor (2023). Optou-se também por utilizar o modelo GPT-3.5-Turbo como alternativa ao mecanismo de IA mostrado em Jonatan e Igor (2023), pois, além de ter uma fácil conexão com a aplicação através de uma API, é

treinada com uma quantidade muito grande de dados, também não é necessário realizar treinamentos, pois o modelo já é treinado e disponibilizado para uso.

Em Pinto (2023), é utilizado a integração do aplicativo de mensagens *Telegram* juntamente com a API do GPT para auxiliar estudantes vítimas de bullying, onde as vítimas podem entrar em contato direto em um *chatbot* específico do *Telegram* e conversar diretamente com a ferramenta integrada ao GPT, onde caso a mensagem seja uma denúncia de bullying, o sistema responde o usuário com o suporte necessário para a situação específica, e salva os detalhes do ocorrido passado pelo usuário. Nesta abordagem, foi utilizado a plataforma *Google Scripts* como forma de implementação das integrações. Pode ser interessante a implementação de uma forma de avaliação automática neste trabalho para coletar a percepção do usuário ao utilizar a ferramenta.

Em Florindi, Fedele e Dimitri (2024), foi criado um *chatbot* capaz de adicionar elementos emocionais durante a conversa, que pode melhorar muito a qualidade das respostas obtidas pela ferramenta. Neste trabalho, semelhante ao *SmartDocAI*, há uma inserção de informações através de um arquivo Formato Portátil de Documento (PDF), que será utilizado como base de dados para os *prompts* utilizados, adicionando também um modelo de inserção de sentimentos para gerar a resposta para o usuário. No *SmartDocAI*, foi utilizada uma arquitetura semelhante, porém voltada para arquivos extensos, que podem economizar tempo. Também foi utilizado o modelo GPT, que não é necessário ser treinado antes do uso, já que foi utilizada uma integração direta utilizando a API da *OpenAI*.

Em Gams et al. (2024), foi realizado a integração de um sistema de *chatbot* numa plataforma médica já existente para auxiliar os usuários com instruções médicas, onde é inserido um arquivo PDF com informações da plataforma como forma de enriquecer o conhecimento gerado pelo modelo GPT, semelhante a forma que é utilizado na *SmartDocAI*, onde o conteúdo do PDF é utilizado como contexto para as respostas juntamente com a pergunta do usuário. Este método utiliza uma plataforma própria para prover o *chatbot*, diferente da abordagem neste trabalho, que é utilizado o *WhatsApp*, também não foi implementado métodos de avaliação automatizado nesta implementação.

3.1 Discussão

Com esta revisão, conclui-se haver trabalhos com propostas semelhantes ao *SmartDocAI*, porém com propósitos diferentes da *SmartDocAI*, que visa auxiliar no processo de consulta em arquivos com quantidades abundantes de texto. Na *SmartDocAI*, foi criado um módulo que permite configurar o conteúdo fornecido na ferramenta de forma rápida, que não é possível observar nos demais trabalhos revisados.

Também pode-se destacar o módulo de avaliação automática para o usuário como um ponto diferencial da ferramenta em comparação com as demais, já que a ferramenta

dispara as mensagens de avaliação de forma automática e coleta as notas fornecidas pelo usuário.

O uso do *WhatsApp* integrado com LLM juntamente com a possibilidade de inserção de documentos PDF para fornecer as informações precisamente para os usuários é inovador por ser possível que diversos tipos de documentos diferentes seja utilizado na ferramenta, ela consegue processar qualquer documento fornecido em formato textual e fornecer informações precisas presentes nestes arquivos.

Abordagem Proposta

Neste capítulo, será detalhada a abordagem metodológica adotada e a proposta desenvolvida.

A proposta foi criada com o intuito de fazer um modelo interativo de consulta automatizada, utilizando tecnologias modernas e modelos de inteligência artificial para processar as respostas.

4.1 Arquitetura do Sistema

Nesta seção será discutido sobre as decisões de sistema escolhido para implementar a proposta apresentada.

A implementação do sistema deve seguir o fluxo mostrado na Figura 1, onde o fluxo começa com o usuário enviando uma mensagem para o *WhatsApp* conectado no número da *SmartDocAI*, então a ferramenta detecta a mensagem e a adiciona no histórico de mensagens da LLM GPT-3.5-turbo. Após a LLM processar esta nova mensagem no histórico, é retornada a resposta para a *SmartDocAI*, então a ferramenta envia a mensagem de volta para o usuário através do *WhatsApp*, fechando o ciclo de operação.

4.1.1 Módulos

Neste sistema, foram escolhidos diversos módulos para suprir as necessidades de extrair o texto do arquivo PDF, fazer a conexão com o *WhatsApp*, processar as mensagens, enviar para o modelo *GPT-3.5-turbo* e também realizar a avaliação das respostas de forma automática.

4.1.1.1 Extração de Dados de Arquivos

Para o conteúdo do PDF ficar disponível para a *SmartDocAI* processar, foi necessário criar um módulo de extração de texto do PDF, onde ele foi escrito em *JavaScript*.

Este módulo possui as seguintes funções:

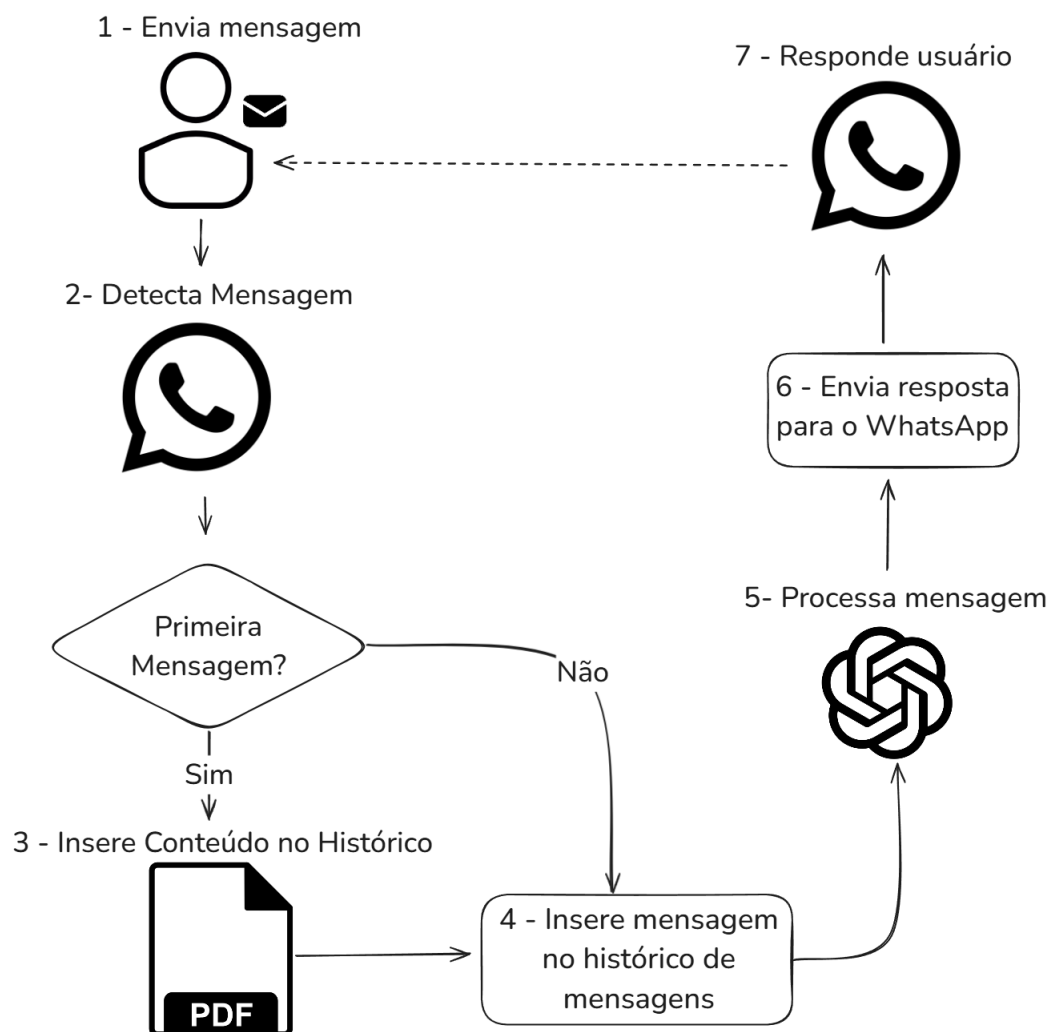


Figura 1 – Fluxo que a aplicação deve seguir.

- ❑ Extração do conteúdo de arquivos de texto como o formato PDF: O código é capaz de ler um arquivo PDF, localizado na raiz do módulo, e utilizar a função *extractText* da biblioteca *ironsoftware* para obter o texto completo do arquivo selecionado, esta biblioteca foi selecionada pois é uma ferramenta de manipulação de arquivos PDF muito leve e prática para inserção nos projetos.
- ❑ Junção do texto com um *prompt* inicial: Neste módulo, há a possibilidade de passar as instruções iniciais que serão enviadas posteriormente para o modelo *GPT-3.5-turbo*, juntamente com o conteúdo do arquivo extraído na função discutida acima.
- ❑ API: Este módulo também contém uma API que serve como interface para manipulação do *prompt*. Através de um método POST, é possível atualizar o *prompt*, sendo a frase de instrução passada para a LLM, enquanto uma rota adicional permite obter tanto o *prompt* quanto o texto do PDF. Dessa forma, é estabelecida a comunicação eficiente entre a *SmartDocAI* e o módulo.

4.1.1.2 Integração com o *WhatsApp*

Outro módulo escolhido para este trabalho foi uma integração realizada com o sistema de mensagens *WhatsApp*, onde cada mensagem recebida por uma conta no *WhatsApp*, é detectado pelo sistema utilizando a biblioteca do *JavaScript* chamada *venom-bot*.

No módulo da integração com o *WhatsApp* estão presentes as funcionalidades:

- ❑ Criar a conexão com o *WhatsApp*: Esta biblioteca permite que, ao iniciar a ferramenta, o sistema gera um *QRCode* para ser escaneado com o *WhatsApp*, conectando assim a conta com a ferramenta.
- ❑ Captar mensagens: Todas as mensagens recebidas com esta conta conectada, é detectada pela biblioteca *venom-bot*, então acontece um filtro para processar apenas as mensagens recebidas por mensagens individuais, desconsiderando grupos, atualizações de status ou qualquer outro evento que não seja uma mensagem de um usuário único.
- ❑ Responder as perguntas feitas para o usuário: Esta integração permite também passar as mensagens processadas e responder diretamente no *WhatsApp* do usuário com a resposta gerada pela LLM, também é utilizada para requisitar as avaliações ao usuário.

4.1.1.3 Processamento das mensagens

Para cada mensagem recebida pelo *WhatsApp*, o sistema realiza um processamento específico. Primeiro, verifica se a mensagem é um número. Se for, considera-se que a mensagem representa uma avaliação, detalhada na Seção 4.1.1.5. Caso contrário, a mensagem é enviada diretamente para o modelo *GPT-3.5-turbo*, com o conteúdo do arquivo PDF armazenado no histórico de conversa. Após receber a resposta do modelo, o sistema encaminha ao usuário, incluindo uma solicitação para o usuário fazer uma avaliação.

4.1.1.4 Integração com o modelo *GPT-3.5-turbo*

Foi necessário realizar a integração com o modelo utilizando a API que a *OpenAI* fornece. Neste módulo, foi preparada uma estrutura para enviar a mensagem do usuário e seu histórico, contendo o conteúdo do PDF e a instrução inicial que pode ser revisada na Seção 4.1.1.1. Então é enviada a mensagem e aguardado o modelo retornar a resposta. Após receber a resposta do modelo, é retornada a resposta para o processamento das mensagens.

O modelo *GPT-3.5-turbo* foi escolhido para testar a eficiência do modelo, já que é o modelo mais barato comparado a modelos como *GPT-4* e também o *GPT-o1*.

4.1.1.5 Implementação de Avaliação automatizada para os usuários

Para ser possível o usuário avaliar a ferramenta, foi projetado um módulo de avaliações, onde após cada resposta da *SmartDocAI*, o sistema automaticamente envia uma mensagem requisitando as avaliações individuais para cada resposta, e após cinco respostas, a *SmartDocAI* requisita uma avaliação geral para o usuário, que pode ser visto na Seção 5.1. Foi modelado um diagrama contendo o fluxo das mensagens de avaliação e como são detectadas na *SmartDocAI*, o sistema é capaz de saber e armazenar qual avaliação ele deve disparar, e pode ser visto na Figura 2.

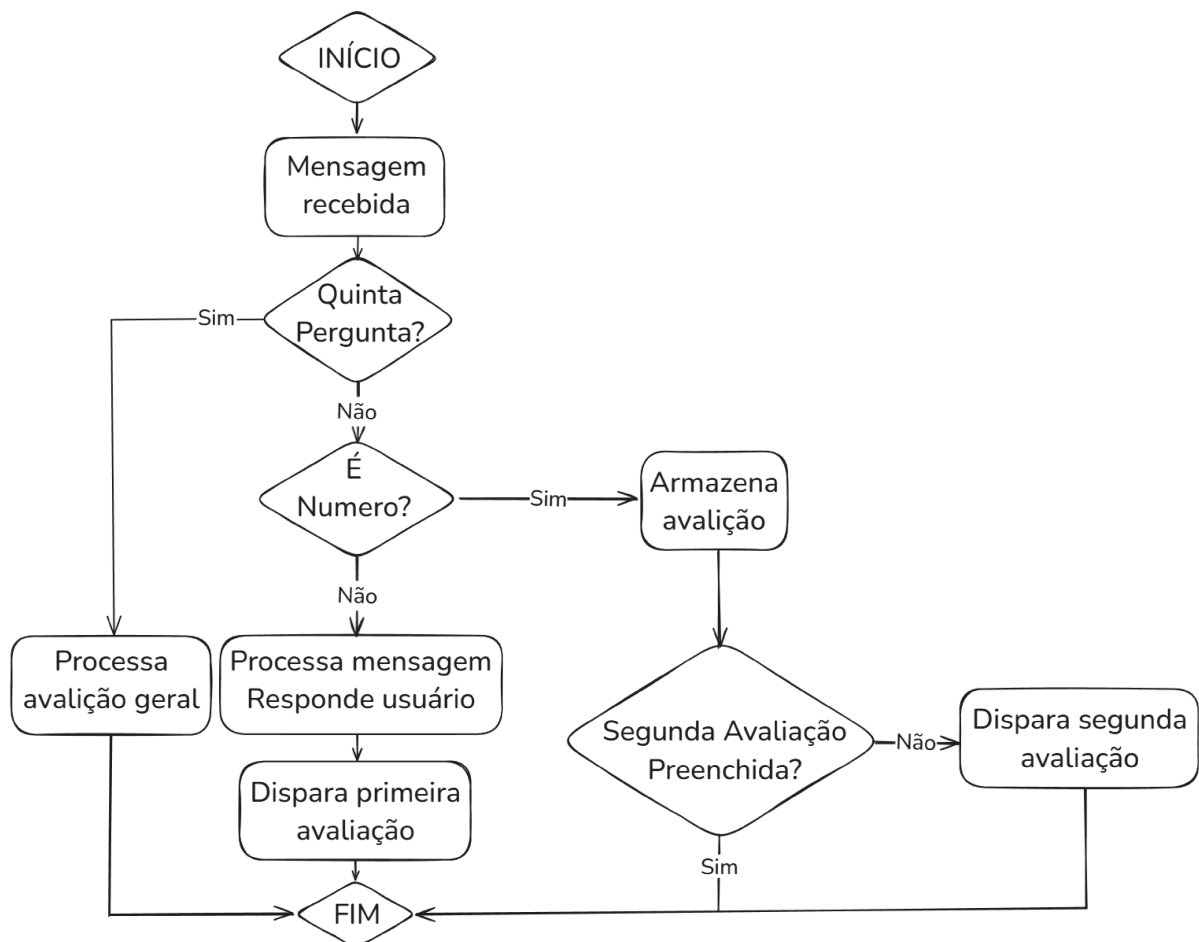


Figura 2 – Diagrama representando todo o fluxo de avaliações implementado.

4.1.2 Funcionamento dos módulos em conjunto

O funcionamento geral da ferramenta começa com a inicialização da *SmartDocAI*, no momento em que o módulo de extração processa o conteúdo do PDF e disponibiliza os dados para o módulo da API. Um *prompt* inicial também é gerado e adicionado ao histórico do modelo *GPT-3.5-turbo*. Quando uma mensagem é recebida pelo *WhatsApp*, o módulo de integração com o *WhatsApp* encaminha a mensagem ao módulo de processamento de

mensagens, que combina o conteúdo do PDF com o histórico existente e o *prompt* enviado pelo usuário. A resposta gerada pelo *GPT-3.5-turbo* é então enviada de volta ao módulo de integração, que a entrega diretamente ao usuário no *WhatsApp*.

4.1.3 Modelagem e Estrutura de Dados

Nesta seção, será discutida como os dados foram armazenados e também a estrutura de implementação do sistema.

4.1.3.1 Armazenamento dos dados

Na implementação da *SmartDocAI*, serão armazenadas algumas informações, dentre elas estão:

- ❑ PDF para extração: Foi necessário carregar o PDF para realizar a extração do texto, este PDF foi disponibilizado na pasta do módulo da Subseção 4.1.1.1, com o nome de arquivo *input.pdf*, então o sistema identifica este nome e realiza a extração.
- ❑ Conteúdo do PDF após extração e *prompt* do usuário: Semelhante ao PDF em si, o arquivo texto contendo o *prompt* e o conteúdo de texto foi armazenado em um arquivo com o nome *prompt.txt*, na pasta do módulo da Subseção 4.1.1.1.
- ❑ Também é necessário armazenar as informações de avaliações dos usuários, estas informações são armazenadas em um arquivo JavaScript Object Notation (JSON) contendo um identificador aleatório para cada número e as informações de avaliação, como as avaliações individuais de cada pergunta e também as avaliações gerais da *SmartDocAI*.

Vale ressaltar que não há armazenamento de nenhum dado dos usuários que utilizam a ferramenta, como número, nome, até mesmo conversas. A *SmartDocAI* não salva estado de nenhuma informação além das informações citadas acima.

4.1.3.2 Estrutura de projeto

Foram definidos alguns pontos em relação à estrutura de projeto, como paradigma de programação utilizada para a realização dos módulos e o tipo do padrão no projeto da *SmartDocAI*.

Para a construção do módulo de extração de dados, citado na Subseção 4.1.1.1, foi utilizado o paradigma de programação orientado a objetos, juntamente com uma arquitetura em camadas bem comum no desenvolvimento de aplicações *BackEnd*, contendo as seguintes camadas:

- ❑ Camada de serviço: Esta camada é criada para conter toda a regra de negócio de cada função da aplicação, contém as funções de obter o *prompt* do usuário juntamente com o conteúdo do arquivo, e também definir o *prompt* do usuário juntamente com o texto extraído do arquivo.
- ❑ Camada de controle: Como este módulo utiliza uma API como interface de comunicação, é necessário ter uma camada de controlador, responsável por receber as requisições, repassar as informações para a camada de serviço juntamente com os dados recebidos pelo usuário e devolver o retorno da função de serviço com o código de status Protocolo de Transferência de Hipertexto (HTTP) adequado ao realizar a chamada da API.
- ❑ Camada de roteamento: Esta camada é necessária para definir os caminhos da API e definir quais controladores estarão presentes em cada rota e também qual o método HTTP que será chamado para realizar determinada requisição. Desta forma, cada controlador fica registrado em uma rota específica com um método HTTP específico, facilitando a chamada e a manutenção do código.

No módulo de integração com o *WhatsApp*, introduzido na Subseção 4.1.1.2, foi utilizado também o paradigma de programação orientado a objetos, pois a biblioteca *venom-bot* por padrão precisa ter uma instância do objeto da classe *WhatsApp*, que é responsável por fazer a conexão e realizar as futuras operações, como envio de mensagens e detectar novas mensagens.

O módulo de processamento de mensagens, apresentado na Subseção 4.1.1.3, é a peça chave do programa, onde foi construída toda a lógica de comunicação entre todos os módulos. Este módulo é responsável por realizar a validação da mensagem quando detectada uma mensagem, por exemplo, se a mensagem não é de origem de uma atualização de status, ou se a mensagem não tem origem de um grupo, e posteriormente, repassar o conteúdo da mensagem para os outros módulos para serem feitas as tratativas, ser processada pelo modelo *GPT-3.5-turbo*, e também repassar a resposta do modelo ao usuário.

O módulo de integração com o modelo *GPT-3.5-turbo*, citado na Subseção 4.1.1.4, é uma classe cujo objetivo é inicializar a integração com a chave de API quando for instanciada e possui os métodos para gerenciar o histórico de mensagens de cada usuário e seu método principal, que recebe como parâmetro o conteúdo da mensagem contendo a pergunta do usuário, o identificador do usuário e faz a chamada para a API do modelo *GPT-3.5-turbo*, recebendo a resposta e retornando o conteúdo para a chamada do método.

4.2 Implementação da *SmartDocAI*

Podemos destacar as principais operações que foram implementadas para tornar possível a realização deste projeto. Nesta subseção é detalhada a implementação de cada

módulo presente na Subseção 4.1.1. A implementação completa da *SmartDocAI* está disponível no GitHub¹.

Async e o *await* são recursos do *JavaScript* que simplificam o trabalho com operações assíncronas, que diferem do código síncrono por não bloquear a execução enquanto aguardam uma tarefa ser concluída. A palavra-chave *async* é usada para declarar funções assíncronas, que podem executar operações de forma paralela com o código principal. Já o *await* é utilizado dentro dessas funções para aguardar o resultado de alguma operação assíncrona antes de continuar o código.

4.2.1 Implementação do módulo Extração de Dados de Arquivos

Foi importada a classe *PDFDocument* da biblioteca *ironsoftware*, e utilizado a função *open* para carregar o PDF localizado na raiz do módulo, após carregado, foi utilizado a função *extractText*, que retorna o conteúdo texto do arquivo para a variável *text*, repassada na função de escrita de arquivo para o arquivo *prompt.txt*, juntamente com o prompt inicial do usuário, essas duas informações serão carregadas no histórico do usuário para ser utilizado pelo modelo *GPT-3.5-turbo*. Essa implementação pode ser vista na função presente na Listagem 4.1.

```
public static async setPrompt(prompt: string) {  
  const pdf = await PdfDocument.open("./src/input.pdf");  
  
  const text = await pdf.extractText();  
  
  fs.writeFileSync("./src/prompt.txt", prompt + "\n\n" + text);  
  
  return { message: "success" };  
}
```

Listagem 4.1 – Código Principal do Módulo de Extração de Dados do PDF.

4.2.2 Implementação do módulo de integração com o WhatsApp

Para se conectar no *WhatsApp*, foi instanciada uma classe do *WhatsApp* utilizando a biblioteca *venom-bot* da linguagem *JavaScript*, então, após realizar a conexão com o celular, é possível utilizar os métodos dessa classe que permite fazer várias ações no *WhatsApp*. Os métodos utilizados dessa classe foi o *onMessage*, que permite o código realizar um determinado fluxo sempre que uma mensagem nova for detectada no *WhatsApp*, permitindo ser tratada e processada pela *SmartDocAI*, e também o método *sendText*, que foi utilizado sempre que foi necessário enviar uma resposta ao cliente que estava entrando em contato com o número ligado na ferramenta. Essa implementação pode ser vista de forma simplificada na Listagem 4.2, onde há a detecção da mensagem, então ela é verificada

¹ Código disponível em: <<https://github.com/viniciuspairesb/tcc>>

caso não seja de grupo, repassado o conteúdo da mensagem para o módulo do modelo *GPT-3.5-turbo*, e enviada a mensagem com a resposta para o cliente.

```
venomClient.onMessage(async message => {  
  if (message.isGroupMsg) return;  
  
  const responseGpt = await chatGpt.completion(message.body, message.from);  
  
  if (!responseGpt) return;  
  
  await venomClient.sendText(message.from, responseGpt);  
});
```

Listagem 4.2 – Código Simplificado do Módulo de Integração com o WhatsApp.

4.2.3 Implementação do módulo de Processamento das Mensagens

O processamento de cada mensagem individualmente se dá início na detecção da mensagem mencionado na Subseção 4.2.2, onde cada mensagem tem a sua verificação se não é uma mensagem do tipo *broadcast*, se não é uma mensagem vazia, se não é de grupo, pois caso alguma dessas cláusulas forem verdadeiras, o código não é executado neste tipo de mensagens. Após essas primeiras verificações, é feita uma checagem se a mensagem é um número, pois se for, ela deve ser tratada como avaliação, e não como uma pergunta normal. Em caso de não ser uma avaliação ou um tipo de mensagem incorreto, a mensagem é tratada como uma pergunta para a ferramenta, então ela é passada para o modelo *GPT-3.5-turbo*, e retornada a resposta ao usuário posteriormente, juntamente com uma requisição de avaliação sobre esta resposta que foi repassada. A implementação simplificada deste módulo pode ser vista na Listagem 4.3.

```
venomClient.onMessage(async message => {  
  if (!message.body || message.from == "status@broadcast") return;  
  
  if (message.isGroupMsg) return;  
  
  let numberQuantityMessage = 1;  
  
  if (message.body.length <= 10 && checkIfHasNumber(message.body)) {  
    handleEvaluation(message.body);  
    return;  
  }  
  
  const responseGpt = await chatGpt.completion(message.body, message.from);  
  
  if (!responseGpt) return;  
  
  await venomClient.sendText(message.from, responseGpt);  
  
  venomClient.sendText(  
    message.from,
```

```

    "Em uma escala de 0 a 5, avalie o nível de satisfacao da resposta obtida."
  );
});

```

Listagem 4.3 – Código Simplificado do Módulo de Processamento de Mensagens.

4.2.4 Implementação do módulo de Integração com o modelo *GPT-3.5-turbo*

Para realizar a integração da *SmartDocAI* com a API do *GPT-3.5-turbo*, foi utilizado a biblioteca oficial da *openai*, onde foi modelado uma classe chamada *ChatGPT*, que ao ser instanciada, define a chave de API na classe, e possui os métodos para obter uma resposta do modelo, que foi chamado de *completion*, neste método é possível observar que o modelo passado com o parâmetro *model* é o *GPT-3.5-turbo*, também há um método para verificar se aquela mensagem é a primeira do usuário, pois se for, é necessário carregar o PDF no histórico do usuário juntamente com a instrução inicial mencionada na Subseção 4.1.1.1. Também há um método chamado *setMessage* para definir o *prompt* que será inserido no histórico inicial de cada usuário, que é enviado para a LLM, contendo o PDF e a instrução inicial dada através da requisição. A implementação simplificada da classe pode ser vista na Listagem 4.4.

```

export default class ChatGPT {
  openAI: OpenAIApi;
  messages: { [messageFrom: string]: ChatCompletionRequestMessage[] } = {};
  constructor() { const openAiApi = getOpenAiApi(); this.openAI = openAiApi; }
  async completion(messageContent: string, messageFrom: string) {
    const message: ChatCompletionRequestMessage = {
      role: "user",
      content: messageContent
    };
    const completion = await this.openAI.createChatCompletion({
      model: "gpt-3.5-turbo-0125",
      temperature: 0,
      max_tokens: 512,
      message
    });
    const gptResponse = completion.data.choices[0].message;
    return gptResponse.content;
  }
  setMessage(message: ChatCompletionRequestMessage, messageFrom: string) {
    this.messages[messageFrom].push(message);
    return this.messages[messageFrom];
  }
  async checkFirstMessage(messageFrom: string) {
    const messages = this.messages[messageFrom];
    if (messages) return;
    const initPrompt = await getPrompt();
    for (const text of initPrompt) {
      const message: ChatCompletionRequestMessage = {
        role: "system",

```

```
        content: text
    };
    this.setMessage(message, messageFrom);
  }
}
```

Listagem 4.4 – Código Simplificado do Módulo de Integração com o GPT-3.5-turbo.

4.3 Execução

Para executar o sistema, foi necessário utilizar um número no *WhatsApp* e realizar a conexão através do *QrCode* gerado para a conexão com o *WhatsApp* que aparece no terminal da ferramenta, após realizar essa conexão, foi necessário apenas enviar uma mensagem no número que estava sendo disponibilizado na ferramenta.

A *SmartDocAI* permite que, mesmo sem expor portas de onde a aplicação está rodando, ou mesmo sem colocar a ferramenta para funcionar em um sistema de *cloud* como *aws*, *azure*, a *SmartDocAI* funcione para todo o mundo, pois a partir do momento que uma mensagem chega no *WhatsApp* que está conectado no servidor, podendo ter a origem de qualquer lugar do mundo, aquela mensagem já entra no fluxo da ferramenta no servidor em que está rodando, já que a ferramenta está conectada diretamente no *WhatsApp* versão web da conta que escaneou o *QrCode*. Então, desta forma, é possível utilizar a *SmartDocAI* em qualquer dispositivo que tenha o ambiente de execução *Node.js* instalado, não sendo necessário se preocupar com roteamento de portas, etc.

A *SmartDocAI* não precisa de uma interface visual, já que é utilizada diretamente na plataforma *WhatsApp*, não sendo necessário nenhuma interface para o seu perfeito funcionamento.

A *SmartDocAI* foi criada, testada e utilizada em um computador pessoal básico, sem abertura de acesso externo via internet através do computador pessoal e funcionou perfeitamente, já que o software não necessita de um computador com grandes capacidades para ser executado, apenas sendo necessário suportar um navegador web com base no *chromium* já é suficiente para a execução perfeita da *SmartDocAI*.

Método e Análise dos Resultados

Neste capítulo, será abordada a metodologia empregada para avaliar a eficiência do modelo *GPT-3.5-Turbo* na interação com documentos extensos, como arquivos em PDF e textos longos. O objetivo principal é descrever o método utilizado para verificar se a abordagem proposta atende aos objetivos gerais e específicos definidos, utilizando testes práticos com usuários reais, que acessaram um edital universitário através da *SmartDocAI*. Em seguida, serão apresentados e discutidos os resultados obtidos, incluindo as avaliações feitas pelos usuários, permitindo uma análise detalhada sobre a eficácia e a relevância da *SmartDocAI* na facilitação do acesso a informações específicas em documentos longos.

5.1 Método para a Avaliação

Nesta seção, serão apresentados os métodos de avaliação necessários para entender a eficiência da *SmartDocAI*.

No total, 19 pessoas participaram das avaliações da *SmartDocAI*, todas vinculadas à Universidade Federal do Pampa e com acesso ao edital utilizado no processo de avaliação. O objetivo principal das avaliações realizadas foi mensurar a qualidade e a precisão das respostas obtidas com o uso da *SmartDocAI*, através de avaliações realizadas pelos usuários e pelos autores. Outro objetivo foi avaliar as respostas da *SmartDocAI* com base no conteúdo original fornecido, verificando se as respostas foram assertivas baseadas no PDF inserido na ferramenta.

A coleta de dados foi feita durante interações de usuários reais com a *SmartDocAI*, seguida de um questionário contendo as informações necessárias para gerar os gráficos apresentados na seção de resultados.

5.1.1 Avaliação dos usuários

Como medidas de avaliação de cada resposta que a ferramenta fornece para o usuário, foram utilizadas duas principais métricas, sendo elas a satisfação do usuário em relação

às respostas recebidas da ferramenta e as clarezas das respostas que o usuário recebeu, sendo aplicadas em diferentes momentos:

- ❑ Satisfação do usuário: Foi avaliado em uma escala de 0 a 5, com uma pergunta da ferramenta para o usuário logo após receber a resposta da *SmartDocAI*.
- ❑ Clareza das respostas: Foi avaliado em uma escala de 0 a 5, sendo disparada esta pergunta após a *SmartDocAI* receber a avaliação do item acima.

Estas duas medidas foram aplicadas ao usuário ao final de cada resposta obtida através da *SmartDocAI* em diferentes cenários de uso e perguntas, visando avaliar cada resposta individual obtida pela ferramenta.

De modo parecido com as avaliações de cada resposta individual, também foi realizado uma avaliação geral da *SmartDocAI*, com o intuito de obter a percepção do usuário em relação à economia de tempo que a *SmartDocAI* oferece se comparado a leitura manual dos documentos fornecidos, e também a probabilidade do uso futuro por parte dos usuários, sendo aplicadas em diferentes momentos:

- ❑ Economia de tempo: Foi avaliado em uma escala de 0 a 5, com uma pergunta da ferramenta para o usuário logo após o usuário realizar 5 perguntas para a *SmartDocAI*.
- ❑ Probabilidade de uso futuro: Foi avaliado em uma escala de 0 a 5, logo após a *SmartDocAI* receber a resposta da avaliação de economia de tempo depois de 5 perguntas feitas pelo usuário para a *SmartDocAI*.

Estas duas medidas têm por finalidade analisar a probabilidade dos usuários em utilizar a ferramenta no futuro, tendo em vista que ela pode economizar o tempo do usuário em comparação com abrir documentos e procurar determinada informação específica.

No geral, um total de 19 pessoas interagiram com a *SmartDocAI* e responderam pelo menos uma avaliação.

Nas avaliações obtidas através dos usuários, foram coletadas primeiro as avaliações referentes a cada pergunta individual, chamando assim de Avaliações Individuais.

Nas avaliações Individuais, foram coletadas duas perguntas após cada resposta dada pela *SmartDocAI*, onde a avaliação foi “Em uma escala de 0 a 5, avalie o quanto a resposta foi esclarecedora.”, conforme pode ser visualizado no exemplo na Figura 3.

Nesta primeira avaliação individual, o objetivo foi obter o nível de satisfação do usuário com a resposta específica dada pela ferramenta, onde a maioria dos usuários considerou que, na escala de 0 a 5, o nível de satisfação foi 5, conforme pode ser notado na Seção 5.2.

Outra questão levantada para o usuário avaliar foi a clareza das respostas, então foi direcionada ao usuário a mensagem “Em uma escala de 0 a 5, avalie o quanto a resposta foi esclarecedora.”, conforme a Figura 3.

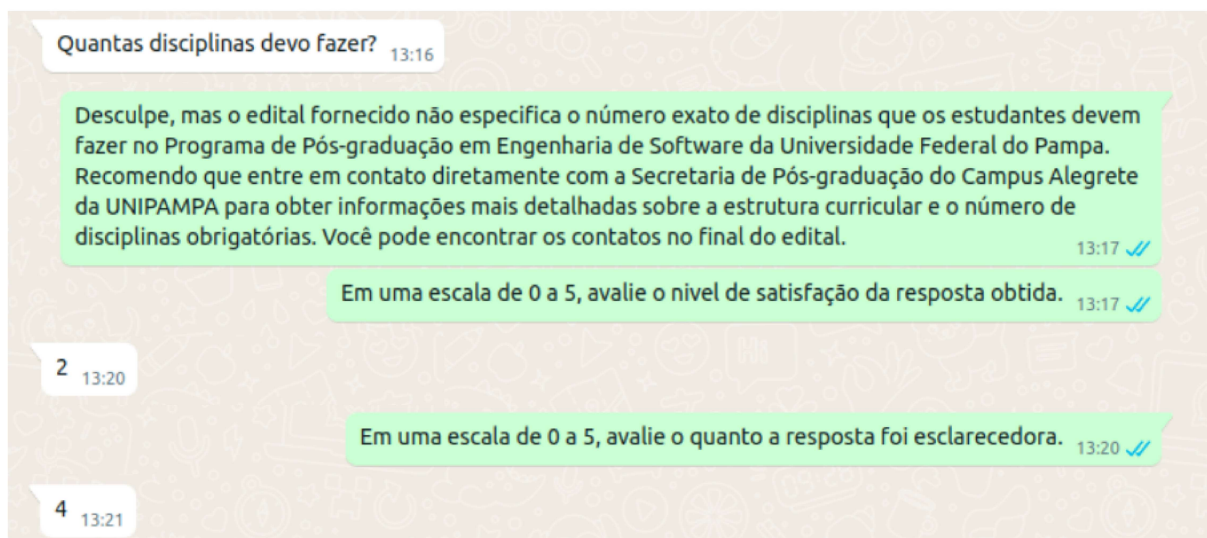


Figura 3 – Figura de exemplo contendo a pergunta do usuário para a ferramenta, a resposta gerada pelo modelo GPT e as avaliações individuais.

O objetivo da segunda avaliação individual foi avaliar o quanto a informação foi entendida pelos usuários. A coleta foi feita no mesmo modelo, em uma escala de 0 a 5, onde a maioria também avaliou como 5.

Nas avaliações Gerais, foram coletadas também duas perguntas, a primeira com o foco em avaliar a economia de tempo em comparação com métodos tradicionais de busca de informações em documentos, e a segunda com o foco em avaliar a intenção de uso futuro pelos usuários, que mostra que a *SmartDocAI* faz sentido quando se compara com outros métodos.

A primeira avaliação geral requisitada foi “De 0 a 5, classifique se você considera que essa ferramenta economiza tempo comparando a abrir o arquivo do edital e procurar a informação necessária?”, conforme visível na Figura 4.

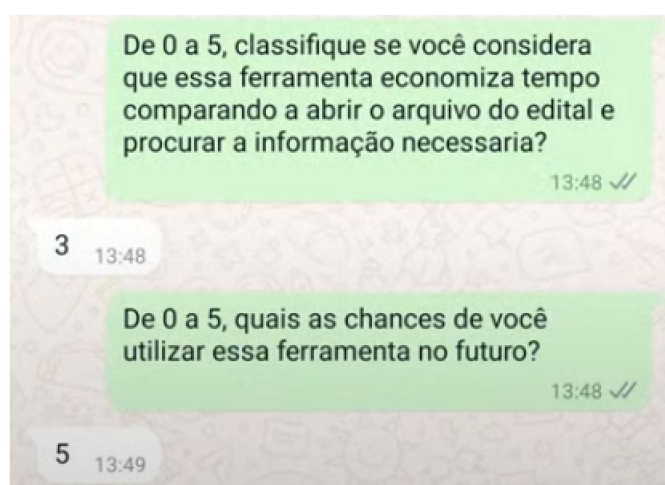


Figura 4 – Figura de exemplo contendo as duas perguntas de avaliações gerais disparadas pela *SmartDocAI*.

Nesta primeira avaliação geral, o usuário foi instruído a dar sua nota de economia de tempo com base no possível tempo que a *SmartDocAI* poderia economizar caso fosse utilizada como alternativa a meios tradicionais, onde mais de 60% dos usuários atribuíram nota máxima, conforme pode ser visto na Seção 5.2.

A segunda avaliação geral requisitada foi “De 0 a 5, quais as chances de você utilizar essa ferramenta no futuro?”, conforme visível na Figura 4. Na segunda avaliação geral, o intuito foi coletar a percepção do usuário em relação à chance de um uso futuro, caso a *SmartDocAI* seja disponibilizada em larga escala. Neste ponto, mais de 70% dos usuários avaliaram com nota máxima em relação às chances de utilizar a *SmartDocAI* em um futuro, conforme pode ser visto na Seção 5.2.

5.1.2 Avaliação das respostas com base no conteúdo fornecido

A avaliação foi realizada utilizando um arquivo PDF de um edital de Ingresso de Discentes no Programa de Pós-Graduação em Engenharia de Software da Universidade Federal do Pampa, contendo 7 páginas de conteúdo e 6508 palavras, disponível no repositório do *GitHub* ¹.

Todas as respostas obtidas pelo modelo foram analisadas pelos autores com base no conteúdo fornecido neste PDF, bem como a precisão do modelo em fornecer o conteúdo exato requisitado pelo usuário, também foi analisado se o modelo apresentou conteúdo inexistente ou não encontrou algum conteúdo fornecido que estava presente no PDF.

Além das avaliações realizadas por parte do usuário, também foi realizada a análise das respostas obtidas através do modelo, onde o foco foi analisar o motivo de respostas que os usuários avaliaram como ruins.

Para isto, foi feita uma seleção com todas as respostas que obtiveram avaliações individuais menores que 3 em algumas das duas avaliações. Após a seleção, foi feita uma análise das perguntas realizadas pelo usuário à *SmartDocAI*, as respostas obtidas pela ferramenta e as notas atribuídas nas avaliações.

No exemplo da resposta presente na Figura 5, o usuário realizou a seguinte pergunta para a ferramenta: “quais são os horários do mestrado?”, então, após realizar a busca no conteúdo fornecido e não encontrar registros sobre os horários, a *SmartDocAI* retorna ao usuário que, os horários específicos de disciplinas no mestrado não foram fornecidos no PDF/edital, então o usuário avaliou a satisfação da resposta como 3.

Neste exemplo, é possível notar que o usuário não ficou devidamente satisfeito com a resposta justamente por não obter a informação que ele gostaria, porém, o conteúdo fornecido para a *SmartDocAI* não continha esta informação, então a resposta foi fornecida de forma correta para o usuário, porém por não existir a informação nos materiais fornecidos, a *SmartDocAI* optou por dizer que não conseguiu encontrar.

¹ Código disponível em: <<https://github.com/viniciuspiresb/tcc>>

Avaliando este cenário, é possível dizer que a ferramenta operou de forma correta com base nas informações que ela possuía em sua base de dados, porque após analisar profundamente o PDF fornecido, foi constatado que realmente não havia aquela informação e que não foi um erro da *SmartDocAI* essa resposta.

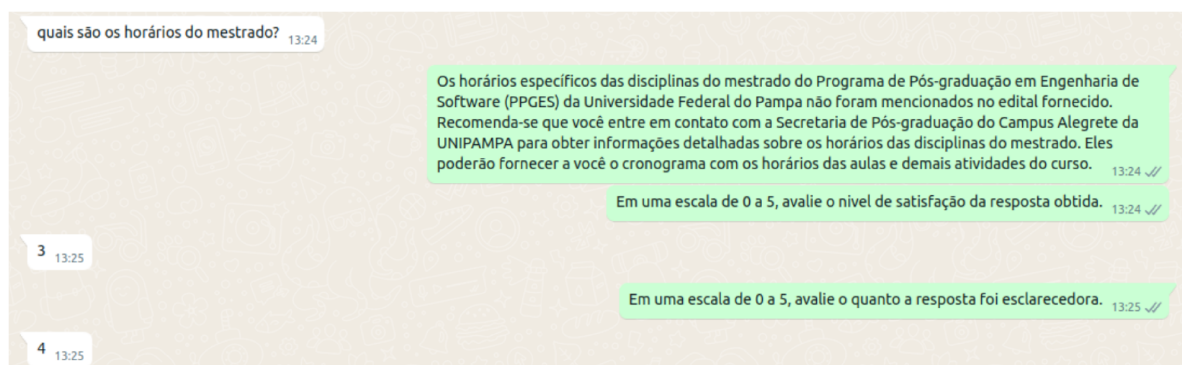


Figura 5 – Figura de exemplo contendo a pergunta do usuário para a ferramenta, a resposta gerada pelo modelo GPT e as avaliações individuais que foram utilizadas para análise na Seção 5.1.2.

5.2 Avaliação dos Resultados

Nesta seção, será apresentada a análise detalhada dos resultados obtidos através dos métodos de avaliação apresentados na seção 5. Os resultados serão divididos entre a análise dos usuários da ferramenta e a análise das respostas dadas pela *SmartDocAI*.

5.2.1 Resultados obtidos pelos usuários da *SmartDocAI*

Neste tópico, serão tratados os resultados obtidos pelas avaliações individuais feitas com os usuários a cada resposta obtida através da ferramenta. Através dos gráficos apresentados nesta seção, pode-se notar que os números presentes dentro de cada fatia no gráfico representa o número de pessoas que atribuíram uma determinada nota na pesquisa representada. O número que aponta para cada fatia é a nota que aquela fatia representa, por exemplo, de 0 a 5. A porcentagem presente abaixo da nota representada na fatia representa quantos por cento de todos os usuários que participaram daquela avaliação escolheram determinado número de 0 a 5.

5.2.1.1 Análise dos resultados das Avaliações Individuais

Após executar a metodologia da Subseção 4.1.1.5 com todos os usuários envolvidos, foram obtidos 68 avaliações para a primeira pergunta e 69 avaliações para a segunda pergunta, este número elevado comparado com o número de pessoas que avaliaram a

ferramenta se deve ao fato de que cada interação com a *SmartDocAI* pode gerar duas avaliações, e este processo se repete para cada pergunta feita para a ferramenta.

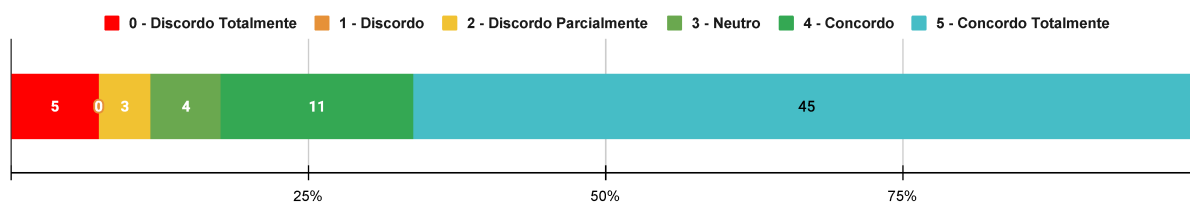


Figura 6 – Gráfico contendo todas as métricas referentes a primeira avaliação individual.

Na Figura 6, é possível notar que a maioria dos usuários avaliou a ferramenta com nota máxima no quesito do nível de satisfação obtido com a resposta da *SmartDocAI*, onde 45 do total avaliaram com nota 5, que representa um total de 66.2% das avaliações. Também no mesmo gráfico, é possível notar que 11 pessoas avaliaram as perguntas com nota 4, que representa 16.2%, somando as notas 5 e 4, obtemos o valor que representa todas as avaliações que foram acima do médio, que é um total de 82.4% das notas atribuídas a esta primeira avaliação das respostas.

Como houve avaliações ruins, como as notas 3, 2 e 0, representando 17.7% das avaliações, foi realizado uma análise mais rigorosa com estas notas para entender o motivo de serem avaliadas desta forma, que pode ser encontrado na Seção 5.2.2.

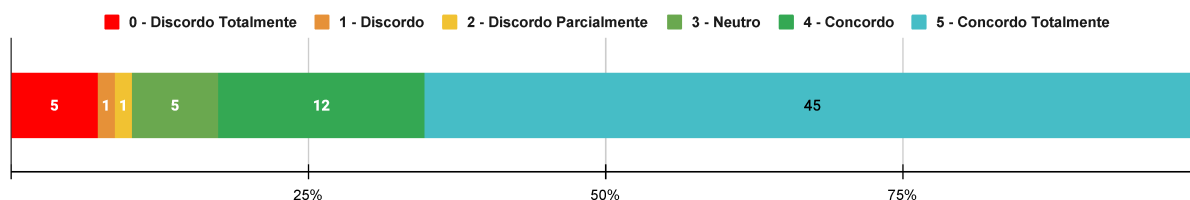


Figura 7 – Gráfico contendo todas as métricas referentes a segunda avaliação individual.

O gráfico representado na Figura 7 apresenta a avaliação dos usuários quanto à clareza das respostas obtidas, semelhante ao primeiro gráfico desta seção. Houve 45 notas 5 para as respostas obtidas, que representam um total de 65.2% do total das avaliações. Semelhantemente, para as notas 4 houve 12 avaliações, representando 17.4% do total. Na soma entre as notas 4 e 5, obtivemos um total de 82.6%.

Semelhante ao gráfico apresentado na primeira avaliação individual, houve avaliações ruins, com notas entre 0 a 3, representando 17,2% do total. Da mesma forma, foram analisadas individualmente o motivo dessas notas abaixo da média na Seção 5.2.2.

5.2.1.2 Análise dos resultados das Avaliações Gerais

Após os usuários realizarem 5 interações com a ferramenta, foram coletadas as avaliações gerais da *SmartDocAI* na percepção do usuário. Foram obtidas 11 avaliações para

a percepção do usuário em relação à economia de tempo comparada a utilizar métodos tradicionais e 11 avaliações para a chance de um uso futuro da ferramenta. Nas avaliações gerais, foi obtido um número menor de avaliações, representando 57.89% dos usuários que avaliaram, pois a pesquisa apareceu para cada usuário apenas uma vez depois de realizar 5 perguntas a *SmartDocAI*.

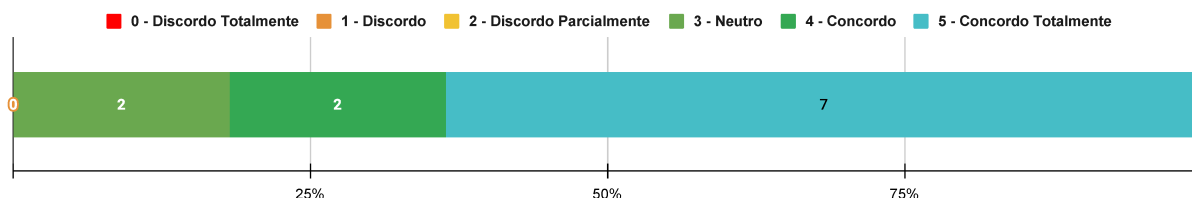


Figura 8 – Gráfico contendo todas as métricas referentes a primeira avaliação geral.

Com a visualização do gráfico da Figura 8, pode ser notado que 7 das 11 pessoas avaliaram o quesito de economia de tempo com nota 5, representando 63.6% do total das avaliações, também houve 2 avaliações com nota 4, representando 18.2% do gráfico e 2 avaliações nota 3, representando 18.2% do total.

Nesta avaliação geral, é possível notar que não houve nenhuma nota 0, 1 ou 2, que representam notas ruins no quesito de economia de tempo.

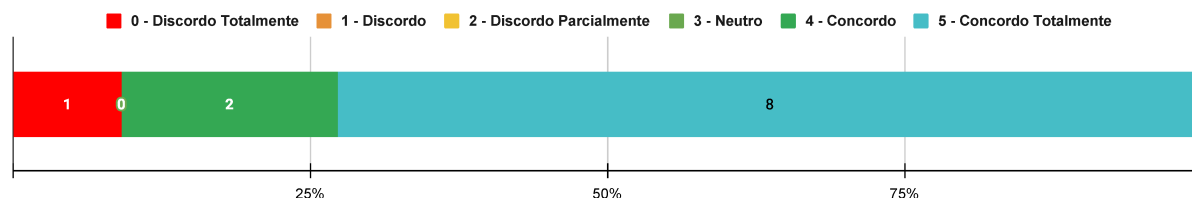


Figura 9 – Gráfico contendo todas as métricas referentes a segunda avaliação geral.

Também foi avaliada a percepção dos usuários em relação à chance de uso futuro da ferramenta, como pode ser notado no gráfico, houve 8 notas 5, que representa um total de 72.7% do total das avaliações, foram recebidas duas notas 4, e uma nota 0. No quesito uso futuro, de 11 usuários que avaliaram a *SmartDocAI*, apenas 1 deles não utilizaria a *SmartDocAI* no futuro, os outros 10 avaliaram entre nota 4 e 5.

5.2.2 Resultados obtidos através da análise das respostas dadas pela *SmartDocAI*

Como algumas avaliações apresentaram resultados insatisfatórios, como notas entre 0 e 3, foi feita uma análise com cada um dos casos, para entender os motivos pelos quais a ferramenta foi avaliada com notas razoáveis.

Neste processo, foi passado por cada mensagem que recebeu uma avaliação razoável, e discutido de o motivo do usuário ter avaliado aquela resposta com a respectiva nota.

Ao total, houve 14 respostas com a avaliação baixa nos dois quesitos de avaliação individual, sendo que uma resposta pode ter ambas as avaliações baixas. Todas elas foram analisadas individualmente e os resultados coletados serão discutidos a seguir.

A grande maioria destas respostas recebeu essas notas devido ao fato de que a pergunta do usuário não estava presente no atual documento fornecido para a ferramenta, então 8 respostas foram avaliadas com notas de 0 a 3 por este motivo, representando 57.14% do total. Nestes casos, a resposta da *SmartDocAI* foi considerada satisfatória pelos autores, pois o problema encontrado não foi na ferramenta em si, mas sim na falta de material a ser lido para responder a requisição do usuário, de qualquer forma, a *SmartDocAI* instruiu o usuário que não havia aquela informação em sua base de dados.

Também houve alguns casos que a *SmartDocAI* respondeu o usuário de forma correta, porém recebeu uma avaliação até 3, foi analisado todas as informações passadas neste caso, porém estavam corretas, não se tem conhecimento do motivo da avaliação ruim pelo usuário, que foi o caso de 4 respostas avaliadas, representando 28.57% das 14 mensagens.

Por fim, temos duas respostas que foram avaliadas com notas de 0 a 3, e que realmente são consideradas insatisfatórias, que representam 14.29% das 14 avaliadas com notas de 0 a 3. No primeiro caso, o usuário questiona quantas disciplinas o aluno bolsista precisa cursar por semestre, então a ferramenta dá a resposta que ele precisa cursar todas as disciplinas obrigatórias do programa. A resposta desta pergunta não estava presente no edital fornecido a *SmartDocAI*, poderia ser mais interessante indicar para o usuário que a ferramenta não possuía a informação na sua base de dados do que, de certa forma, fugir um pouco da pergunta realizada. No segundo caso, o usuário perguntou à ferramenta os requisitos para entrar no programa, então a *SmartDocAI* respondeu com o item 3 do edital, que lista os requisitos, mas poderia citar itens de regras presentes no item 7 do edital para complementar sua resposta.

Conclusão

Este capítulo conclui o trabalho, avaliando o cumprimento dos objetivos gerais e específicos ao longo do desenvolvimento. A pesquisa comprovou que o *chatbot* integrado ao modelo GPT-3.5 atendeu à proposta inicial, mostrando-se eficaz na consulta de informações em documentos de múltiplas páginas que custa um certo tempo para ser realizado a leitura. As principais contribuições incluem a validação da hipótese e sugestões para futuros aprimoramentos, como o uso de modelos mais avançados e uma extração de dados aprimorada.

Também pode-se citar as avaliações positivas que os usuários avaliaram a *SmartDocAI*, obtendo ótimas notas, e a grande maioria das pessoas afirmando que a ferramenta poupa tempo e utilizaria em um futuro.

6.1 Principais Contribuições

A implementação da *SmartDocAI*, um *chatbot* integrado ao modelo LLM GPT-3.5, possibilitou o acesso rápido e eficiente a informações em documentos extensos, como PDFs e arquivos de texto. Os experimentos com usuários reais demonstraram que a ferramenta superou os métodos tradicionais de consulta, proporcionando uma experiência satisfatória, refletida nas avaliações positivas sobre a clareza e precisão das respostas.

Os resultados indicaram que a maioria dos usuários considerou as respostas claras e expressou alta satisfação com a economia de tempo em relação às abordagens convencionais. Essas evidências reforçam a eficácia da integração entre inteligência artificial e *chatbots* para facilitar o acesso à informação, consolidando a *SmartDocAI* como uma solução valiosa para instituições que lidam com grandes volumes de documentos. Além disso, sugestões para melhorias e expansões da ferramenta indicam seu potencial para futuras implementações.

6.2 Trabalhos Futuros

Após executar as metodologias e realizar uma análise profunda dos resultados obtidos, notou-se um bom resultado utilizando o modelo *GPT-3.5*, porém, diversos novos modelos são lançados com o passar do tempo, além do mais, alguns usuários que avaliaram a ferramenta realizaram perguntas sobre conteúdos que não estavam presentes nos editais fornecidos como base de dados para a ferramenta.

Trabalhos futuros poderiam aplicar novamente as metodologias utilizando novos modelos GPT e analisar se há uma melhor compreensão do conteúdo carregado. Também pode ser analisado se a qualidade das respostas obtidas apresenta melhora caso sejam carregados diversos arquivos na ferramenta, fornecendo assim um melhor contexto sobre o conteúdo fornecido. Também pode ser interessante trabalhar em um *prompt* mais abrangente para o modelo GPT, buscando informações sem depender do conteúdo fornecido na ferramenta.

Também pode ser interessante inserir no módulo de extração de conteúdo do arquivo inserido na ferramenta, algum modelo que extrai informações das imagens presentes no PDF de conteúdo, assim, fornecendo mais contexto para o modelo GPT trabalhar e entender todo o conteúdo presente no arquivo, incluindo as imagens.

6.3 Contribuições em Produção Bibliográfica

Com este trabalho desenvolvido, foi possível apresentar um resumo preliminar no evento XVIII Workshop de Teses e Dissertações em Ciência da Computação, realizado na Universidade Federal de Uberlândia pela Faculdade de Computação (FACOM).

Referências

ADAMOPOULOU, E.; MOUSSIADES, L. An overview of chatbot technology. In: MAGLOGIANNIS, I.; ILIADIS, L.; PIMENIDIS, E. (Ed.). **Artificial Intelligence Applications and Innovations**. Cham: Springer International Publishing, 2020. p. 373–383. ISBN 978-3-030-49186-4. Citado na página 13.

CAHN, J. Chatbot: Architecture, design, & development. **University of Pennsylvania School of Engineering and Applied Science Department of Computer and Information Science**, 2017. Citado na página 13.

DAM, S. K. et al. **A Complete Survey on LLM-based AI Chatbots**. 2024. Disponível em: <<https://arxiv.org/abs/2406.16937>>. Citado na página 10.

FLORINDI, F.; FEDELE, P.; DIMITRI, G. M. A novel solution for the development of a sentimental analysis chatbot integrating chatgpt. **Personal and Ubiquitous Computing**, Springer Science and Business Media LLC, jul. 2024. ISSN 1617-4917. Disponível em: <<http://dx.doi.org/10.1007/s00779-024-01824-6>>. Citado na página 17.

GAMS, M. et al. Developing a medical chatbot: Integrating medical knowledge into gpt for healthcare applications. In: _____. **Intelligent Environments 2024: Combined Proceedings of Workshops and Demos; Videos Session**. IOS Press, 2024. ISBN 9781643685212. Disponível em: <<http://dx.doi.org/10.3233/AISE240018>>. Citado na página 17.

Gisnaya Faridatul Avisyah, Ivandi Julatha Putra, Sidiq Syamsul Hidayat. **View of Open Artificial Intelligence Analysis using ChatGPT Integrated with Telegram Bot**. 2023. Disponível em: <<https://eltikom.poliban.ac.id/index.php/eltikom/article/view/724/110>>. Citado na página 16.

IBM. **Large Language Models - IBM**. 2024. <<https://www.ibm.com/topics/large-language-models>>. Accessed: 2024-10-20. Disponível em: <<https://www.ibm.com/topics/large-language-models>>. Citado na página 14.

JONATAN, V.; IGOR, A.-A. **Creation Of A ChatBot Based On Natural Language Proccesing For Whatsapp**. 2023. Disponível em: <<https://arxiv.org/abs/2310.10675>>. Citado na página 16.

- KHURANA, D. et al. Natural language processing: state of the art, current trends and challenges. **Multimedia Tools and Applications**, v. 82, n. 3, p. 3713–3744, Jan 2023. ISSN 1573-7721. Disponível em: <<https://doi.org/10.1007/s11042-022-13428-4>>. Citado na página 13.
- LOKMAN, A. S.; AMEEDDEEN, M. A. Modern chatbot systems: A technical review. In: ARAI, K.; BHATIA, R.; KAPOOR, S. (Ed.). **Proceedings of the Future Technologies Conference (FTC) 2018**. Cham: Springer International Publishing, 2019. p. 1012–1023. ISBN 978-3-030-02683-7. Citado na página 10.
- NADKARNI, P. M.; OHNO-MACHADO, L.; CHAPMAN, W. W. Natural language processing: an introduction. **Journal of the American Medical Informatics Association**, v. 18, n. 5, p. 544–551, 09 2011. ISSN 1067-5027. Disponível em: <<https://doi.org/10.1136/amiajnl-2011-000464>>. Citado na página 10.
- PINTO, R. M. **Artificial Intelligence in the Fight Against Bullying: Integration of ChatGPT in an Emotional Support Chatbot**. 2023. <<https://ceur-ws.org/Vol-3691/paper30.pdf>>. [Accessed 23-08-2024]. Citado na página 17.
- Putri Ariatna Alia, Johan Suryo Prayogo, Rony Kriswibowo, Agung Teguh Setyadi. **Implementation Open Artificial Intelligence ChattGPT Integrated With Whatsapp Bot**. 2024. Disponível em: <<https://repository.uam.ac.id/id/eprint/1057/1/Jurnal%20Implementation%20Open%20Artificial%20Intelligence%20ChattGPT%20Integrated%20Bot.pdf>>. Citado na página 16.
- SERVICES, A. W. **What is GPT? - Amazon Web Services (AWS)**. 2024. <<https://aws.amazon.com/what-is/gpt/>>. Accessed: 2024-10-20. Disponível em: <<https://aws.amazon.com/what-is/gpt/>>. Citado na página 15.
- Statista. **WhatsApp in Brazil - Statistics & Facts**. 2024. <<https://www.statista.com/topics/7731/whatsapp-in-brazil/>>. Accessed: 2024-10-20. Citado na página 11.
- VASWANI, A. et al. **Attention Is All You Need**. 2023. Disponível em: <<https://arxiv.org/abs/1706.03762>>. Citado na página 14.