ENTIDARE: Um Modelo Inteligente para Auxiliar Aprendizes na Elaboração de Modelos Conceituais de Software

Lucas Sommer [1]
Kleinner Farias [2]

Resumo: Modelagem conceitual é uma forma de ajudar pessoas a entender conceitos, é uma forma de descrever estruturas e comportamentos. A UML é o padrão atual de como representar estes conceitos em forma de diagramas. Portanto acaba que aprendizes precisam criar modelos conceituais sem qualquer suporte ferramental ou guia para a sua construção, incluindo na identificação de conceitos e seus relacionamentos. Este artigo propõe o Entidare, um modelo inteligente para auxiliar aprendizes na elaboração de modelos conceituais de *software*, para propor isso a abordagem foi pensada em uma arquitetura baseada em componentes, com o intuito de aplicar *machine learning* em questões específicas de criação de diagramas UML para levantar as entidades, as relações, e os atributos destas entidades. O Entidare foi avaliado através de um estudo de caso para demonstrar sua capacidade e através de uma avaliação de aceitabilidade com o questionário TAM (Modelo de Aceitação de Tecnologia). No total, 12 participantes responderam os questionários. Os resultados do estudo de caso demonstram que é viável o Entidare para auxiliar na elaboração de diagramas de classes da UML, obtendo valores apropriados. Também os participantes perceberam uma facilidade (94,43%) assim como uma utilidade (100%) do Entidare, e ainda uma intenção de uso da abordagem (100%). Estes resultados demonstram que a abordagem é promissora para o uso no contexto de auxiliar aprendizes.

Palavras-chave: Arquitetura Baseada em Componentes; Aprendizado de Máquina; Transformadores; Modelagem Conceitual; ChatGPT

Abstract: Conceptual modeling is a way of helping people understand concepts, it is a way of describing structures and behaviors. UML is the current standard for how to represent these concepts in the form of diagrams. Therefore, it turns out that learners need to create conceptual models without any tool support or guidance for their construction, including the identification of concepts and their relationships. This article proposes Entidare, an intelligent model to assist learners in the development of conceptual software models. To propose this, the approach was designed in a component-based architecture, with the aim of applying machine learning in specific issues of creating UML diagrams to identify entities, relationships, and attributes of these entities. Entidare was evaluated through a case study to demonstrate its capabilities and through an acceptability assessment with the TAM (Technology Acceptance Model) questionnaire. In total, 12 participants answered the questionnaires. The results of the case study demonstrate that Entidare is viable to assist in the creation of UML class diagrams, obtaining appropriate values. Participants also perceived an ease (94.43%) as well as a usefulness (100%) of Entidare, and even an intention to use the approach (100%). These results demonstrate that the approach is promising for use in the context of helping learners.

[1]Graduando de Ciência da Computação pela Unisinos. Email:lucasommer@edu.unisinos.br

[2]Possui doutorado em Informática pela Pontifícia Universidade Católica do Rio de Janeiro (2012), mestrado em Ciência da Computação pela Pontifícia Universidade Católica do Rio Grande do Sul (2008), graduação em Ciência da Computação pela Universidade Federal de Alagoas (2006) e em Tecnologia da Informação pelo Instituto Federal de Alagoas. Email: kleinnerfarias@unisinos.br

Keywords: Component-based Architecture, Machine Learning, Transformers; Conceptual Models, ChatGPT

1 INTRODUÇÃO

A linguagem de modelagem unificada, no inglês chamado de *unified modeling language* (UML), é o padrão na indústria atual para representar os *designs* de um sistema de *software* usando diagramas (BERGSTRÖM et al., 2022). A modelagem de diagramas UML é uma etapa fundamental para o processo de desenvolvimento de um *software* (REN et al., 2022). Um dos objetivos primários de diagramas de classe da UML é de providenciar uma visão geral da estrutura de um sistema (BERGSTRÖM et al., 2022). Diagramas são uma ajuda importante para entender os sistemas de um *software* e para compartilhar o entendimento do sistema entre os times de desenvolvimento (BERGSTRÖM et al., 2022). Acredita-se que linguagens visuais são superiores à linguagens textuais por conseguir suportar a percepção humana e os pensamentos, e isso é verdadeiro para UML (STöRRLE, 2012).

Muitos alunos encontram dificuldades no aprendizado de *design* de *software*, um dos motivos relacionados é que muitas vezes existem diversas formas de solucionar o problema ao contrario de haver apenas uma solução (RUKMONO; CHAUDRON, 2022). Outro fator para a dificuldade pode ser encontrado em razão de haver múltiplas perspectivas quando se está desenvolvendo diagramas (RUKMONO; CHAUDRON, 2022). Existem estudos no qual buscam avaliar a inteligência artificial para a modelagem de diagramas UML, como (CÁMARA et al., 2023). Porém (CÁMARA et al., 2023) chegou na conclusão que o ChatGPT só é capaz de lidar com modelos pequenos, e que não é capaz de entender conceitos básicos, como associação de classes e heranças múltiplas. Já (REN et al., 2022) faz uma série de experimentos com a utilização de um *chatbot* com a intuição de auxiliar humanos a desenvolver diagramas.

Como mencionado por (COMBEMALE; GRAY; RUMPE, 2023) a inovação de inteligências artificiais generativas não vêm de percepções novas e sim da recombinação de soluções e conhecimentos já existentes. Um estudo realizado por (REN et al., 2022) na utilização de um *chatbot* para a aplicação de *software engineering* com o intuito de reduzir a distância entre os requerimentos de engenharia e modelação. Já (CÁMARA et al., 2023) realizou experimentos utilizando o ChatGPT para a criação de diagramas UML, porém mesmo quando os *prompts* foram criados cuidadosamente e especificamente, o ChatGPT não gerou os resultados esperados. Por fim (CÁMARA et al., 2023) conclui que o ChatGPT ainda não é uma ferramenta confiável para produzir diagramas, portanto ainda existe muito espaço para melhorar e otimizar os resultados esperados com a inteligência artificial.

Este artigo, portanto, propõe Entidare, no qual é uma aplicação baseada em componentes para identificar entidades, relações e atributos. Aprendizes se beneficiam com o uso do Entidare quando há uma necessidade de auxílio para levantar as entidades, relações e atributos de um enunciado de elaboração de diagramas de classe UML. Para isso o Entidare utiliza *machine*

learning com um *dataset* especificamente treinado para identificar estas características de um enunciado, possibilitando aprendizes a focar nas cardinalidades e validar se todas as recomendações são válidas.

Este artigo apresenta uma abordagem para auxiliar aprendizes no levantamento de requisitos para a elaboração de diagramas de classe da UML. Para a proposta atual foi desenvolvido um protótipo da abordagem utilizando uma arquitetura baseada em componentes, onde o componente Java *back-end* faz a montagem da estrutura em objetos de entidades, relações e atributos, já o componente de *machine learning* realizado pela linguagem Python retorna para o Java as *tags* necessárias para descobrir as entidades, relações e atributos encontrados em um enunciado para elaboração de diagramas de classe da UML. O sistema foi avaliado através de uma pesquisa onde os participantes assistiram um vídeo de como a ferramenta funciona e após isso responderam um questionário com o Modelo de Aceitação de Tecnologia (TAM) (GRANIĆ; MARANGUNIĆ, 2019), assim como foi avaliado através de um estudo de caso entre o Entidare e o ChatGPT. Os resultados obtidos demonstram que existe uma necessidade de apoio de uma ferramenta para auxiliar aprendizes na elaboração de diagramas de classe da UML.

O estudo está dividido conforme a seguinte estrutura: a Seção 2 contêm o referencial teórico, com os principais conceitos para o entendimento do estudo proposto; a Seção 3 aborda os trabalhos relacionados, explorando o processo de seleção utilizado e também realizando um comparativo destes com o presente; a Seção 4 demonstra como a abordagem proposta foi pensada para o seu desenvolvimento; a Seção 5 apresenta as avaliações realizadas nesta abordagem; e, por fim, a Seção 6 traça algumas conclusões e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Esta seção aborda os conceitos teóricos usados durante a construção e desenvolvimento do estudo. A seção 2.1 demonstra a modelagem conceitual para a elaboração do Entidare. Já a seção 2.2 introduz o conceito de *machine learning*. A seção 2.3 aponta os conceitos básicos de *prompt engineering*.

2.1 Modelagem Conceitual

A figura 1 apresenta o diagrama de classes da UML que foi criado para representar as classes que serão utilizadas para resolver o problema abordado, a imagem do diagrama está demonstrado a seguir.

O diagrama conta com uma classe chamada questionário, onde nele estará o *input* da questão do usuário, essa questão é aplicada uma classificação, desta classificação se recebe uma lista de palavras e uma lista de códigos, esta lista de códigos são os valores definidos no *dataset* da etapa de *machine learning*. A classificação por si pode conter zero ou mais palavras, a palavra têm como atributos um nome e um código, a palavra têm duas especializações, a entidade

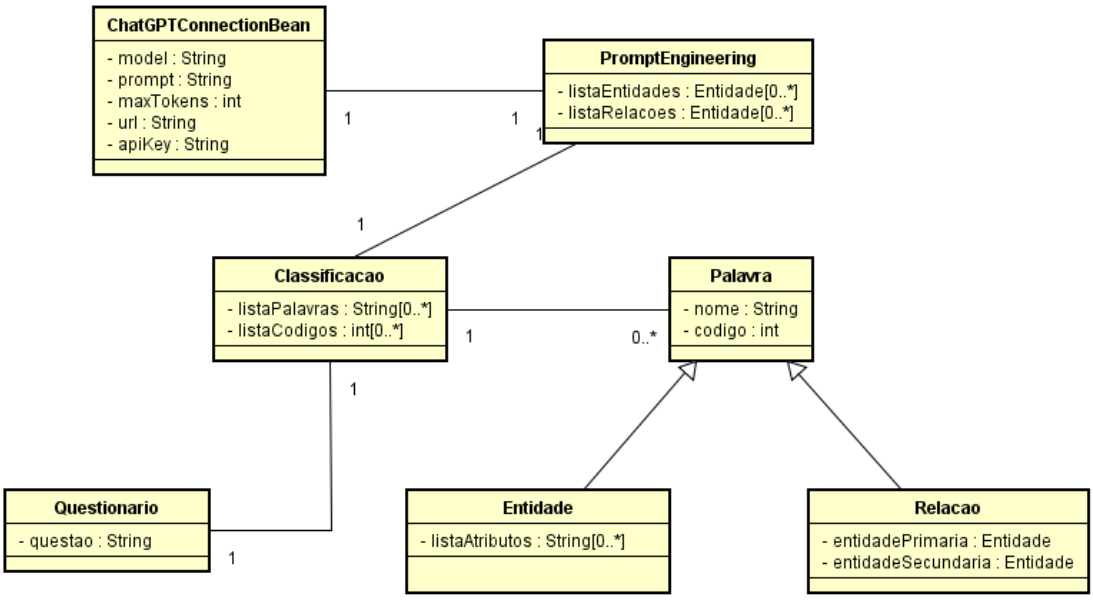

Figura 1 – Diagrama de classes de UML

no qual possuí uma lista de atributos, e relação no qual possuí uma entidade primária e uma entidade secundária. A classificação também é utilizada para a etapa de *prompt engineering* que também é um objeto, do qual possui uma lista de entidades e uma lista de relações. Já o *prompt engineering* é aplicado no objeto base de conexão com o ChatGPT para a criação do atributo de *prompt*, todos os atributos encontrados dentro do objeto do ChatGPT serão necessários para a realização da conexão REST.

2.2 Machine Learning

Machine learning é uma das ferramentas de auxílio no processo de inteligência artificial, como citado por (JANIESCH; ZSCHECH; HEINRICH, 2021) é o processo de melhorar a performance do programa de computador de acordo com experiência a respeito de algumas tarefas através de medidas de performances. *Machine learning* obtém isso aplicando algoritmos que iterativamente aprendem sobre o problema específico de dados de treinamento, no qual permite computadores ter uma percepção de padrões complexos sem explicitamente ser programado (JANIESCH; ZSCHECH; HEINRICH, 2021).

Machine learning pode ser dividido em três tipos de aprendizados, *supervised*, *unsupervised* e *semisupervised*. *Supervised* é definido como o valor da saída, normalmente chamada de *label* em *machine learning* é conhecida para cada observação (BI et al., 2019). Já o *unsupervised* é quando o algoritmo tenta identificar a relação natural e os agrupamentos encontrados dentro dos dados sem referências a nenhum resultado (BI et al., 2019). E por fim, o *semisupervised* combina as duas propostas, para *datasets* muito grandes, utiliza a tática de aplicar *label* para alguns dados limitados, com uma abundância de dados sem *label* com o objetivo de melhorar a

performance do modelo (BI et al., 2019).

A introdução de um modelo para redes neurais chamado *Transformer* modificou o cenário de processamento de linguagens naturais nos últimos anos (BRAşOVEANU; ANDONIE, 2020). A arquitetura inicial do *Transformer* começou com a ideia de *encoder-decoder* (LI; ZHANG; CHEN, 2021). Como explicado por (SINGH; MAHMOOD, 2021) sua arquitetura foi construída com o intuito de gerar paralelismo nos dados sequenciais em redes neurais recorrentes e *long short term memory*. O *decoder* prevê a próxima palavra através de uma função *softmax* que é executado múltiplas vezes até a geração do *token* de finalização da sentença (SINGH; MAHMOOD, 2021).

2.3 Prompt Engineering

Para compreender *prompt engineering* é necessário antes, compreender o que é uma LLM (*large language model*). LLMs são modelos de *deep learning* no qual foram treinados com *datasets* massivos para realizar tarefas específicas (CÁMARA et al., 2023). Um *Prompt* é um conjunto de instruções providenciado a um LLM que customiza ou melhora como a LLM realiza suas capacidades (WHITE et al., 2023). Portanto um *prompt* consegue influenciar as interações subsequentes com o *output* gerado, com o envio de regras específicas e orientações para a LLM seguir em suas conversas (WHITE et al., 2023).

Prompt engineering é os meios pelo qual LLMs são programados via *prompts* (WHITE et al., 2023). Algumas táticas para projetar *prompt engineering* vão facilitar na criação de um *prompt engineering* para a tarefa que busca realizar, uma destas táticas é o *prompt template engineering* que como definido por (LIU et al., 2023) é o processo de criar uma função de *prompting* que resulta na mais efetiva performance na tarefa. De acordo com (LIU et al., 2023) para definir um *prompt template engineering* é preciso considerar um *prompt shape* e decidir se a criação dos *prompts* vão ser dados de forma manual ou automática. Já o *prompt shape* é a variação que os *prompts* terão para a tarefa que deseja executar e portanto vão variar para cada tipo de *prompt engineering* que for querer aplicar (LIU et al., 2023).

3 TRABALHOS RELACIONADOS

A pesquisa pelos trabalhos relacionados foi realizada em repositórios digitais como *Google Scholar*. A *query* de busca utilizado para encontrar os trabalhos relacionados foi "("Diagram"OR "Model"OR "UML"OR "Conceptual Models") AND ("Recognize"OR "Identify"OR "Equivalent") AND ("Entity"OR "Classify") AND ("Machine Learning"OR "Transformers") AND ("Recommend"OR "Suggest"OR "Artificial Intelligence"OR "Intelligent model")". A metodologia aplicada para analisar os trabalhos relacionados já foi validada por estudos realizados previamente na literatura ((RUBERT; FARIAS, 2022); (CABANE; FARIAS, 2023); (VIEIRA; FARIAS, 2020b); (JÚNIOR; FARIAS, 2021); (GONÇALES et al., 2019); (URDAN-

GARIN; FARIAS; BARBOSA, 2021)).

3.1 Análise dos trabalhos relacionados

(DOMíNGUEZ-JIMéNEZ et al., 2020) (TR1). Desenvolveu uma luva para a aquisição dos dados GSR e PPG, os sensores foram então conectados a um microcontrolador. Após a seleção de características então foi aplicado algoritmos para a classificação como *linear discriminant analysis* e *support vector machines*, os *datasets* preparados e centralizados então foram divididos em 80% treino e 20% teste. Como conclusão é possível reconhecer emoções através de sinais PPG e GSR com alta acurácia, porém ainda existe a necessidade de buscar maneiras de identificar outras emoções assim como os estímulos no qual os gera.

(SHRIVASTAVA; KUMAR, 2021) (TR2). Propõe um modelo utilizando BERT para detectar sarcasmo em textos na língua inglesa. O *dataset* composto de 3834 *tweets* foi dividido em 3000 para treinamento e 834 para testes, onde foi utilizado como parâmetros de avaliação precisão, *F1-score* e acurácia. Como conclusão, observou-se que o modelo pode ser expandido para funcionar em múltiplas línguas, assim como expandir o modelo para detectar sarcasmo com dados visuais e de áudio.

(ZHAO et al., 2021) (TR3). Introduziu a ideia de afinar um modelo BERT para realizar o reconhecimento de entidades em materiais literários. Com esse propósito em mente foi treinado dois tipos de BERT, o BERT base e o SciBert. O BERT base foi treinado com 11308 livros, enquanto o SciBert foi treinado com artigos acadêmicos. Foi aplicado para análise as medidas de precisão, *recall* e *F1-score*. Observou-se que os resultados são promissores para tarefas de mineração de texto, pelo fato de o modelo BERT conseguir uma performance boa mesmo sem o conhecimento contextual, porém pelo fato de ter sido treinado por um vocabulário de livros aleatórios, teve uma parcela de *sub-tokens* no qual não foi capaz de reconhecer.

(JI; WEI; XU, 2020) (TR4). Realizou um estudo no qual foi proposto uma arquitetura de normalização de entidades sobre 3 tipos de modelos, BERT, BioBert e ClinicalBert no qual foram feito experimentos para avaliar o comportamento de cada um. Foi utilizado 3 *datasets* para o experimento, o ShARe/CLEF, NCBI e TAC2017ADR. Foi aplicado como métrica de avaliação para a normalização de diferentes entidades a acurácia, definida pela porcentagem de entidades que foram corretamente normalizadas. Como conclusão, é possível observar que afinar um modelo pré-treinado BERT para a normalização de entidades bio-medicinais é um modo efetivo de efetuar tal atividade.

(BUKHARI et al., 2022) (TR5). Utilizou uma *rede neural artificial* para examinar e prever a performance dos alunos. Com o objetivo principal de identificar alunos que estão com nota baixa, no intuito de que os educadores consigam auxiliar esses alunos em como melhorar suas performances. A performance da rede neural artificial foi classificada pela acurácia, a precisão, o *recall* e o *F1-score*, o *dataset* foi composto pelas informações como resultados acadêmicos anteriores, histórico familiar e informações das atitudes do aluno, assim como informações

dos semestres anteriores do aluno. Observou-se que, as variáveis escolhidas são de extrema importância para conseguir prever a performance de alunos, porém uma das limitações é que necessita ter as notas dos semestres anteriores para prever se o aluno está falhando, o que pode significar ser tarde demais para os educadores conseguirem oferecer apoio a esse aluno.

3.2 Análise comparativa dos trabalhos relacionados

Critério de Comparação. Foram definidos sete Critérios de Comparação para efetuar a comparação de similaridades e diferenças entre o trabalho atual e os artigos selecionados. Sua comparação é fundamental no auxílio da identificação de oportunidades de pesquisa utilizando critérios objetivos. Os critérios são descritos a seguir:

- **Objetivo**: Compreende o que o estudo busca atingir de resultado, é o marco inicial do que o estudo averígua.
- **Algoritmos**: Compõe as diferentes formas de como foi produzido os estudos, as diferentes tecnologias lógicas utilizadas, no quesito de programação.
- **Domínio**: Representa em qual campo de atuação social o estudo se adéqua.
- **Contexto**: Informa em qual contexto social se encontra o estudo proposto, se faz parte da área acadêmica ou da indústria.
- **Contribuição Principal**: Compreende o que o estudo no final de todo o processo conseguiu realizar e inovar no contexto inserido.
- **Métodos de Pesquisa**: Pode ser definido pelas múltiplas maneiras de conduzir uma pesquisa, a maneira de como foi adquirido as informações e também de como demonstrar estas.
- **Artefato de Software**: Representa o que o estudo conseguiu criar no final do estudo, caso tenha sido criado um software como parte do objetivo ou resultado do estudo.

Oportunidades de pesquisa. A tabela 1 apresenta a comparação dos estudos selecionados, demonstrando o objetivo e critério de todos. Após isso, as oportunidades observadas a partir das comparações: (1) apenas o trabalho atual busca utilizar *transformers* para o reconhecimento de entidades em questões para modelos conceituais e diagramas de classe da UML; (2) apenas um outro trabalho propõe um modelo inteligente, mas para funcionalidades diferentes. Portanto, a seguinte oportunidade de pesquisa foi encontrada: A criação de um modelo inteligente para auxiliar aprendizes no levantamento e criação de modelos conceituais. Tal conceito que será explorado nas próximas seções.

Tabela 1 – Análise de comparação dos trabalhos relacionados selecionados

ID	Objetivo	Algoritmos	Domínio	Contexto	Contribuição	Método	Artefato
Trabalho Atual	Reconhecimento de entidades para auxiliar aprendizes em criar modelos conceituais	Transformadores	Educação	Academia	Modelo Inteligente para auxiliar no reconhecimento de modelos conceituais	Estudo de caso	Modelo Inteligente
TR1	Reconhecimento de emoções por *Machine Learning*	*Random Forest* Recursivo e GBM	Biomedicina	Indústria	Viável detectar emoções através da GSR	*Survey*	uma luva com um Bluno Nano, DFRobot
TR2	Reconhecimento de sarcasmo em textos nas redes sociais	Transformadores	Social	Indústria	Propôs um modelo Google BERT para detectar sarcasmo	Estudo de caso	BERT
TR3	Afinando um BERT para o reconhecimento de entidades	Transformadores	Literatura	Academia	Focou em extrair entidades nomeadas da literatura	Estudo de caso	BERT
TR4	Afinando um BERT para a normalização de entidades	Transformadores	Biomedicina	Indústria	Demonstrou o potencial de metodologias *deep-learning*	Experimento Controlado	BERT
TR5	Prevendo a performance de alunos através de uma ANN	Rede Neural Artificial	Educação	Academia	Propondo ANN como o algoritmo ideal para atingir a maior acurácia e performance	Experimento Controlado	Modelo Inteligente

4 ABORDAGEM PROPOSTA

Esta seção descreve o Entidare, a abordagem proposta para resolver o problema. A Seção 4.1 apresenta uma visão geral da abordagem proposta e de seu funcionamento. A Seção 4.2 descreve uma visão da arquitetura baseada em componentes pensada para a proposta. A Seção 4.3 detalha as decisões do projeto, do qual são fundamentais para o seu funcionamento e existência. Por fim, a Seção 4.4 traz as linguagens e tecnologias que serão utilizadas para o desenvolvimento e criação do sistema proposto nesta abordagem. A estrutura utilizada para esta seção foi baseado em um estudo previamente realizado na literatura ((VIEIRA; FARIAS, 2020a)).

4.1 Visão Geral da Abordagem Proposta

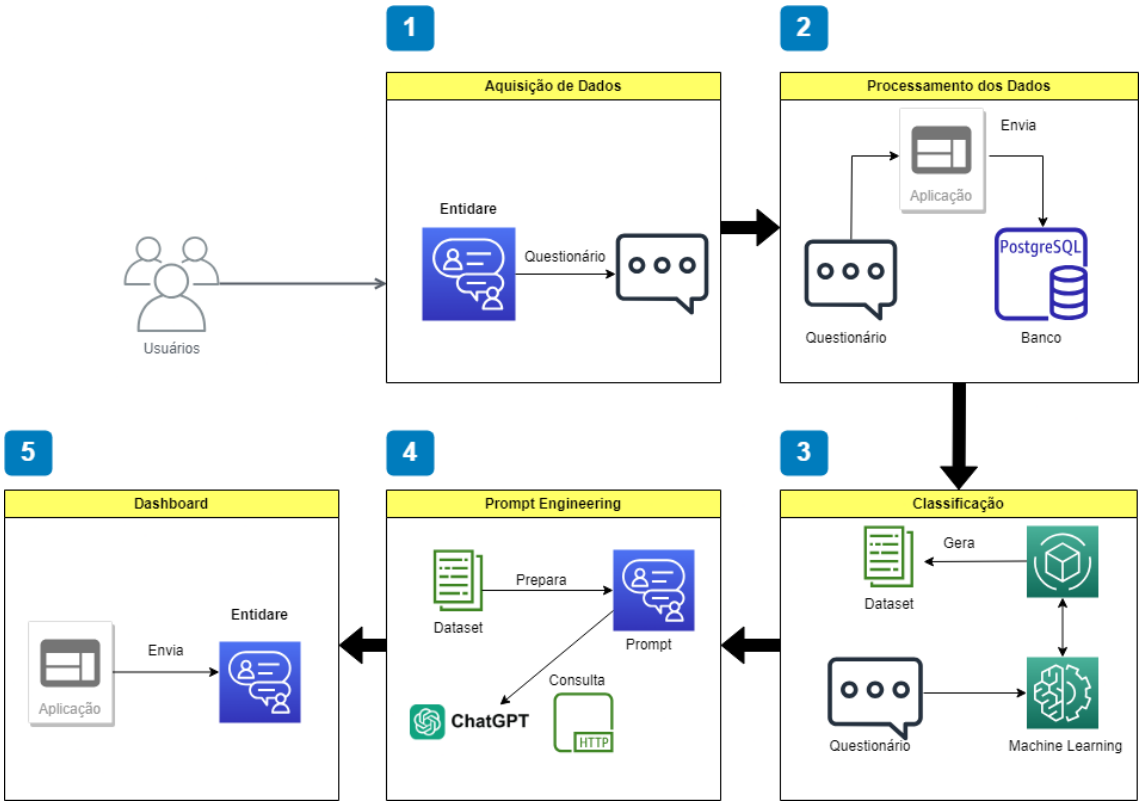

Figura 2 – Uma visão geral da abordagem proposta pelo modelo

A Figura 2 apresenta os passos da abordagem desejada em um formato de visão geral do sistema e sua funcionalidade, o modelo foi dividido em 5 etapas, no qual foram definidas como:

- **Passo 1: Aquisição de Dados.** Esta etapa é composta pelas atividades de comunicação entre o usuário e o sistema, no caso é definido como a etapa onde o usuário faz suas dúvidas ao sistema, etapa de *input* no qual suas dúvidas são os dados sendo adquiridos para o modelo conseguir prosseguir com o processo de auxílio, em um contexto mais

técnico é um *chatbox* no qual o usuário se comunica para obter suas respostas, e ao mesmo tempo é por onde o sistema irá obter os dados que necessita.

- **Passo 2: Processamento dos Dados.** A etapa que procede a **Aquisição de Dados** têm por sua característica principal pegar esta questão ou dúvida realizada pelo usuário e então salvar no banco de dados. Após salvar o *input* no banco de dados, verifica se este questionamento já não foi realizado anterior na base existente, esta verificação não é importante nesta etapa, mas será na próxima etapa, já que esta etapa fica apenas encarregada de salvar no banco de dados as informações recebidas na aquisição e verificar se é a primeira vez que essa questão foi realizada para o modelo.

- **Passo 3: Classificação.** A característica principal desta etapa se encontra em utilizar a parte de *machine learning* utilizando *transformers* que está vinculada no modelo, para gerar então as entidades, as relações entre essas entidades e os atributos destas entidade, que são concebidos através dos dados adquiridos na primeira etapa do processo, ou seja, da questão colocada como *input* pelo usuário. Após gerar o resultado da classificação, e validar se conseguiu gerar um resultado positivo, envia-o para a próxima etapa, conhecida como *prompt engineering*.

- **Passo 4: *Prompt Engineering*.** Esta etapa é caracterizada pela execução do processo de *prompt engineering*, que no caso, é formularizar as entidades, relações e atributos recebidos da etapa de classificação, para a geração de *prompts* com o objetivo de facilitar o retorno esperado que a inteligência artificial vai conseguir em sua resposta sobre estes *prompts*. Estes *prompts* são necessários, pois frequentemente, as inteligências artificiais necessitam de um jeito específico de indagação para obter resultados favoráveis naquilo em que se busca uma resposta.

- **Passo 5: *Dashboard*.** A função desta etapa é de ligar o resultado do processo de classificação gerado pela funcionalidade de *machine learning* e do processo de *prompt engineering* para formalizar a pergunta sendo feita para a inteligência artificial, e em seguida, demonstrar esse resultado na tela para o usuário, portanto essa etapa é também o processo de *output* do modelo para o usuário.

4.2 Arquitetura Baseada em Componentes

A Figura 3 apresenta a arquitetura de componentes produzido para representar o sistema, do qual é formado por sete componentes, que dentro destes componentes estão as etapas discutidas na visão geral. Cada etapa é discutida em mais detalhes a seguir.

Observability Dashboard: O fluxo do modelo inicia e termina neste mesmo componente, ele serve como o *input* dos usuários para as questões que desejam, e ao mesmo tempo o *output*

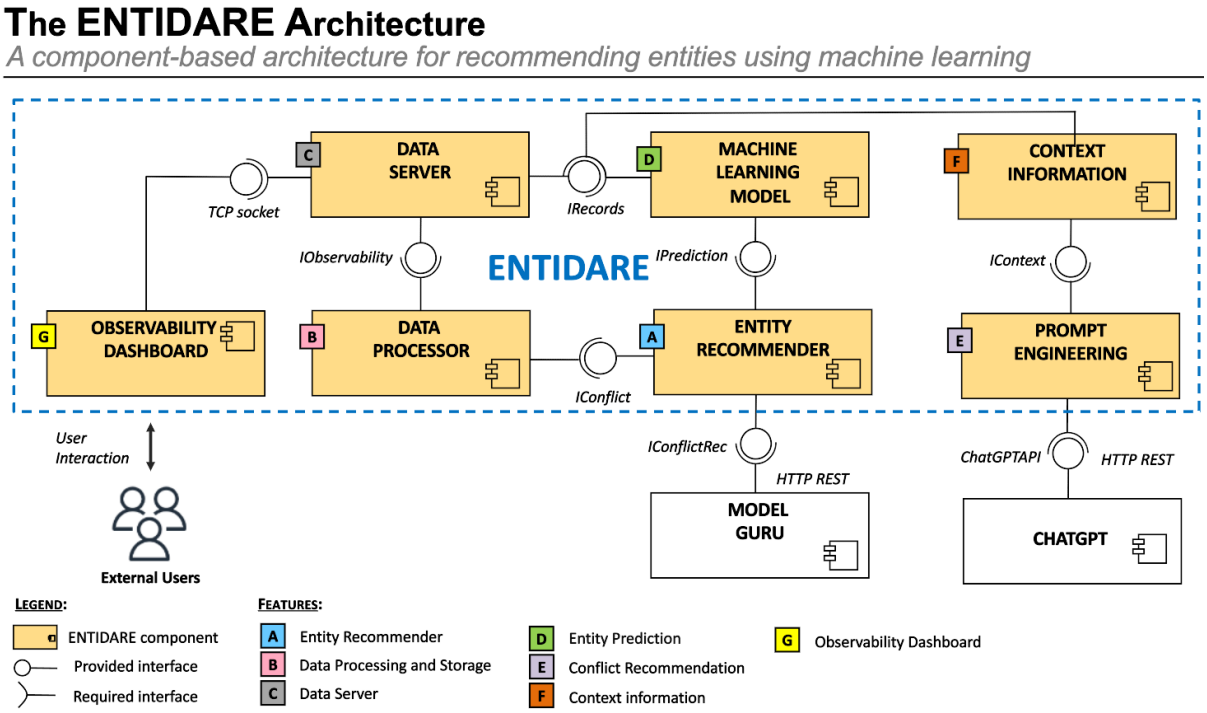

Figura 3 – Arquitetura baseada em componentes

gerado pelo sistema para as questões dos usuários, por fim realiza a comunicação REST com o *back-end* Java para retornar os resultados obtidos.

Data Processor: A funcionalidade deste componente está no processo de analisar o *input* dos usuários, verificar a validade da entrada recebida anteriormente, e enviar então para o banco de dados a questão, é a etapa que precede a análise e recomendação de entidades, visto que têm por objetivo garantir que os dados são passíveis de fazer uma análise para recomendar entidades.

Machine Learning Model: Esse componente é o processo como um todo de gerar o modelo treinado com o *dataset* de questões utilizadas para identificar entidades, relacionamentos e atributos destas entidades. Esta etapa é fundamental para o funcionamento do componente de *entity recommender* pelo motivo de ser o processo que provê toda a lógica necessária para a identificação das informações, assim como é o componente onde se encontra a maior parte das funcionalidades providas de utilizar a linguagem Python.

Entity Recommender: O fluxo que se encontra nesse componente, inicia na etapa do *machine learning model*, pois para a recomendação de entidades, é fundamental a existência do modelo treinado, do qual para existir precisa de um *dataset* de treinamento e validação com dados no estilo de questionários possíveis para a recomendação de entidades para diagramas de classe da UML. Utiliza tanto a questão feita pelo usuário e o modelo gerado pela etapa de *machine learning model* para identificar tanto as entidades, como os atributos e relações entre estas entidades.

Context Information: Este componente representa a busca das informações importantes salvadas no banco de dados, no qual são necessárias no processo de recomendação de entidades,

neste caso, as informações salvas são as entidades identificadas, as relações identificadas e os atributos identificados. Após essa busca então prepara essas informações para enviar para o processo do componente de *prompt engineering*.

Prompt Engineering: Componente crucial para o processo de comunicação com a inteligência artificial, é composto por toda a funcionalidade de preparar as entidades, as relações entre estas entidades, e os atributos das entidades, em *prompts* preparados e otimizados para obter a melhor resposta possível do ChatGPT. É o componente que entra em ação logo após a recomendação de entidades, utiliza da etapa de *context information* para obter essas recomendações, no qual são fundamentais no processo final de auxiliar os usuários em como elaborar o diagrama de classe da UML do problema do *input* inicial.

Data Server: É composto pelas funcionalidades atribuídas ao banco de dados do sistema, que têm por objetivos principais, salvar os dados de *input* do usuário, assim como salvar as entidades, relações e atributos recomendados sobre esse *input*, e salvar uma relação entre a questão e os valores recomendados, e por fim, disponibilizar essas informações de entidades para a etapa de consulta ao ChatGPT que é funcionalidade do componente de *prompt engineering*.

4.3 Decisões de Projeto

Baseado nos estudos realizados sobre os Trabalhos Relacionados 3.1, é possível identificar características comuns entre os trabalhos estudados, em seguida é possível definir algumas decisões sobre a abordagem que determinam os pontos mais importantes, estes pontos são listados a seguir:

1. **A Aplicabilidade do ChatGPT em modelagem de *software*.** A aplicação de inteligência artificial e *chatbots* para modelagem de software é uma questão recente sendo analisada, exemplificado com (REN et al., 2022) que utilizou o SOCIO *chatbot* para realizar diversos experimentos com a aplicatividade para modelagem UML, outro exemplo seria (CÁMARA et al., 2023) no qual realizou experimentos com o ChatGPT para realizar a tarefa de modelagem de diagramas. Já a abordagem atual busca utilizar o ChatGPT através de *prompt engineering* para recomendar as entidades, relações e atributos recomendados para realizar um estudo de caso com o processo de *machine learning* da abordagem proposta.

2. **A Utilidade de *Prompt Engineering*.** A abordagem faz uso de *prompt engineering* através de dados recebidos de uma etapa de *machine learning*, com o intuito de consultar o ChatGPT através de comunicação REST para retornar por fim uma recomendação de como criar um diagrama de classe da UML. Pelo fato do projeto possuir comunicação com uma inteligência artificial, o processo de *prompt engineering* é essencial para conseguir otimizar as perguntas e conseguir respostas eficazes do ChatGPT.

3. **A Praticabilidade de *Machine Learning*.** A abordagem faz uso de *machine learning*

com o intuito de recomendar entidades para a criação de diagramas. O sistema aplica o algoritmo conhecido como *transformers* para este processo, para o seu funcionamento é fornecido um *dataset*, no qual é composto por questões práticas de desenvolvimento de diagramas de classe da UML, com este *dataset* então é realizado a criação do modelo treinado, toda essa etapa, é feito de maneira prática no projeto, pelo uso da biblioteca Tensorflow.

4. **A Versatilidade de aplicar *Prompt Engineering* e/ou *Machine Learning*.** A abordagem atual conta com a tomada de decisão que o sistema efetuará entre utilizar *machine learning* e *prompt engineering* quando ambas as etapas forem favoráveis para o processo, caso o contrário, a abordagem pode optar por utilizar apenas o *prompt engineering*, essa tomada de decisão, essa versatilidade é um fator importante da abordagem. Assim como a decisão de utilizar ambas as opções, no caso o *machine learning* para tentar otimizar e melhorar o *prompt engineering* que será realizado no ChatGPT, um experimento de junção das duas funcionalidades.

5. **A Flexibilidade de uma arquitetura baseada em componentes.** A abordagem foi iniciada na base de uma arquitetura baseada em componentes, no qual permite separar as funcionalidades do sistema em elementos separados, no qual disponibilizam ou necessitam de uma interface para integrar com outros componentes, isso permite uma flexibilidade na implementação e na separação das funcionalidades e dados de cada elemento que são uma das vantagens de uma arquitetura baseada em componentes. A abordagem aproveita essa arquitetura também na questão de comunicação entre os componentes, principalmente na integração entre a recomendação de entidades e a comunicação REST com a etapa de *machine learning*.

4.4 Aspectos de Implementação

O projeto utiliza de diversas linguagens de programação para criar o modelo proposto, no qual serão descritos em qual parte da abordagem serão utilizadas e o motivo de ter escolhido tal linguagem de programação a seguir.

HTML: Conhecido como *Hyper Text Markup Language* foi a linguagem escolhida para implementar a interface de usuário, funcionalidade essa conhecida como *dashboard* da abordagem. A escolha do HTML foi devido ao fato de ser possível e fácil a criação de uma interface de usuário com o suporte do CSS 3 para a visualização dos dados que o usuário precisa receber. Em mais detalhes também foi utilizado uma biblioteca de *front-end* chamada de Bootstrap para a criação da interface de forma responsiva e *mobile-friendly* em paginas web.

Javascript: Foi a linguagem utilizada para fazer a junção entre o que é demonstrado pelo HTML e o que precisa chegar no usuário do *back-end*. Um dos motivos de usar javascript foi devido ao fato de ser a linguagem predominante quando o assunto é paginas web, tendo em

vista o fato de possuir vários *frameworks* para facilitar o seu uso, sendo VUE o que foi escolhido dentre estes para ser aplicado na abordagem.

Java: Foi a linguagem selecionada para desenvolver as camadas de comunicação REST, de comunicação com o ChatGPT, a conexão com o banco de dados, o processamento dos dados, a etapa de carregar as informações de contexto, de realizar o processo de *prompt engineering*, de criar *sockets* de comunicação entre processos e de vincular o *dashboard* na parte do *front-end* com o *back-end*. O banco de dados relacional escolhido para a abordagem foi o PostgreSQL. A decisão de utilizar Java pode ser definida por dois motivos, primeiro pela facilidade que a linguagem proporciona para realizar processos de comunicação REST e o segundo de vincular os seus dados com um banco de dados.

Python: Seu uso é devido ao fato de ser nela onde serão utilizadas as práticas para a criação de *sockets* de comunicação entre processos e da execução da operação de *machine learning*, usando desta forma o algoritmo de *transformers* para aplicar classificação textual, como mencionado por (ALAM; KHAN; ALAM, 2020) que modelos baseados na utilização de *transformers* são adequados para a atividade de classificação textual. A escolha da linguagem Python é dada pelo (SANCHEZ; ROMERO; MORALES, 2020) Python ter uma grande importância no processo de *Machine Learning* comparado com outras linguagens pelo suporte que se encontra em seus *frameworks* de *Deep learning*. O *framework* que será utilizado para o desempenho de *machine learning* conta com uma biblioteca chamada de Tensorflow, que de acordo com (SANCHEZ; ROMERO; MORALES, 2020), é uma biblioteca *open source* para *machine learning* que permite implantar computação tanto em CPU e GPU. Junto com esse *framework* também se encontra e sera utilizado o Keras que é uma biblioteca para redes neurais e que no projeto fará a parte de gerar e salvar o modelo treinado.

A figura 4 demonstra a tela de *dashboard* que foi implementada na abordagem, com o intuito de demonstrar os resultados assim como possibilitar o usuário de usar a aplicação e visualizar a resposta que a implementação da abordagem proposta proporciona. Já a figura 5 apresenta os botões implementados na tela de *dashboard*, estes botões servem tanto para limpeza de campos como para fazer consulta para o *back-end*.

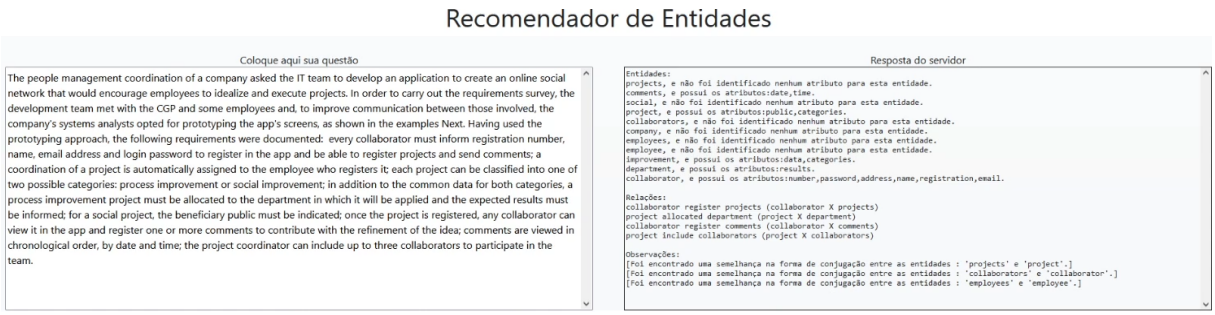

Figura 4 – Tela de dashboard da abordagem proposta

A tela de *dashboard* para visualizar os resultados, conta com um *textarea* para preencher a questão ou pergunta do usuário na esquerda, já o *textarea* localizado na direita é o local no qual

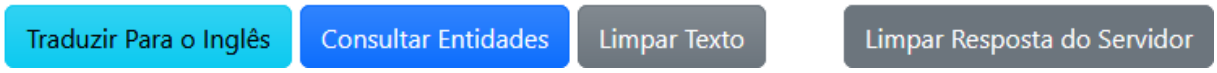

Figura 5 – Botões da tela de dashboard da abordagem proposta

o servidor irá responder de acordo com a questão do usuário. A tela conta também com alguns botões para o seu funcionamento, no qual serão descritos a seguir.

Traduzir Para o Inglês. Esse botão serve para o usuário traduzir sua questão se ela estiver em português para o inglês, já que a abordagem proposta funciona apenas no inglês. Por fim esse botão faz uma consulta para a API do *Google* passando como parâmetros a língua original, a língua na qual busca traduzir e o próprio texto sendo traduzido.

Consultar Entidades. Já esse botão realiza o processo mais importante da abordagem, é o processo de enviar para o *backend* Java a questão recebida no *textarea* à esquerda, ou seja, do *input* do usuário e realizar o processo de montagem da estrutura de objetos de entidades, atributos e relações. Para isso o Java também faz uma consulta REST para o servidor Python, do qual, realiza o processo de *machine learning* e esse resultado o Java utiliza aplicando orientação a objetos e montando uma estrutura de objetos Java *bean* para entidades e relações, com o objetivo de enviar um JSON para o *front-end*, onde será formatado para facilitar a visualização do usuário e, com isso, mostrar ao usuário a resposta no *textarea* a direita. E, por fim, os botões de limpar texto e limpar resposta do servidor, servem para o usuário limpar a própria questão de *input* e de limpar o *output* do servidor, respectivamente.

5 AVALIAÇÃO

Esta seção descreve como o modelo inteligente proposto foi avaliado. Para isso, a Seção 5.1 reporta um estudo de caso realizado para avaliar se a abordagem favorece usuários na elaboração de diagramas de classe da UML. A Seção 5.2 apresenta o procedimento adotado para avaliar a aceitabilidade do modelo proposto.

5.1 Estudo de Caso

Esta seção apresenta uma avaliação quantitativa do modelo inteligente proposto através da realização de um estudo caso. Para isso, uma análise comparativa com o ChatGPT foi realizada através da execução de 10 cenários realísticos de avaliação (Seção 5.1.1). A efetividade da abordagem proposta foi quantificada através das métricas de precisão e *recall* em contextos práticos (Seção 5.1.2). Por fim, os resultados coletados são analisados (Seção 5.1.3).

5.1.1 Cenários de Avaliação

A tabela 2 apresenta uma descrição resumida sobre a tarefa que foi aplicada para os contextos encontrarem as respostas corretas. O cenário de avaliação foi realizado para 10 tarefas experimentais para avaliar os sistemas representados na métrica de contexto, foram aplicadas em 10 tarefas para descobrir a precisão e *recall*, valores representados na métrica de variáveis, e cada tarefa possui o seu valor representado para os atributos, entidades e relações tanto da abordagem proposta quanto do próprio ChatGPT. Destas 10 tarefas, as primeiras 5 foram selecionadas de questionários de provas do ENADE de diferentes anos, e as últimas 5 foram selecionadas de questões de âmbito escolar sobre a criação de diagramas de classe da UML.

Tabela 2 – Descrição resumida das tarefas aplicadas

Nome	Descrição Resumida
Tarefa 1	Todo colaborador deve informar número de matrícula, nome, endereço de e-mail e senha de login para se cadastrar no app e poder registrar projetos e enviar comentários; a coordenação de um projeto é atribuída automaticamente ao colaborador que o registrar; Cada projeto pode ser classificado em uma de duas possíveis categorias: Melhoria de processo ou Social; um projeto de Melhoria de Processo deve ser alocado ao departamento; para um projeto Social, deve-se indicar o público beneficiário; qualquer colaborador pode registrar um ou mais comentários.
Tarefa 2	Essa empresa tem três tipos de colaborador: o comissionado, o horista e o assalariado. Todos os colaboradores registram o número de horas trabalhadas no mês. os comissionados registram o valor do percentual de comissão e o valor total acumulado no mês; os horistas registram o valor recebido por hora; e os assalariados registram o valor do salário. Cada colaborador pertence a um departamento e cada departamento possui pelo menos um colaborador.
Tarefa 3	Uma revista científica possui título, ISSN e periodicidade; Essa revista publica diversas edições com os seguintes atributos: número da edição, volume da edição e data de edição. Um artigo possui título e nome do autor. Um artigo é um conteúdo exclusivo de uma edição.
Tarefa 4	Uma montadora de automóveis produz carros de luxo e esportivos. Um carro é formado de várias partes e cada parte pode ser fabricada por diferentes fornecedores. Um gerente ou um operador possui permissão para cadastrar partes do carro, e consultar a sua disponibilidade para a fabricação dos carros.
Tarefa 5	No tiro esportivo, cada competição é composta por eventos, que podem ser a fase classificatório, a fase final ou a cerimônia de entrega de medalhas. Para cada evento, é preciso saber a data de realização, o horário de início, o horário estimado de término e o local. As competições possuem o equipamento utilizado, que pode ser pistola, carabina ou espingarda. Para cada atleta, é preciso registrar o nome, o país e o comitê olímpico nacional.
Tarefa 6	Uma Escola tem várias turmas. Uma turma tem vários professores , sendo que um professor pode ministrar aulas em mais de uma turma. Uma turma tem sempre aulas na mesma sala, mas uma sala pode estar associada a várias turmas com horários diferentes.
Tarefa 7	Um berçário deseja informatizar suas operações. Quando um bebê nasce, algumas informações são armazenadas sobre ele, tais como: nome, data do nascimento, peso do nascimento, altura, a mãe deste bebê e o médico que fez seu parto. Para as mães guarda informações como: nome, endereço, telefone e data de nascimento. Para os médicos, é importante saber: CRM, nome, telefone celular e especialidade.
Tarefa 8	Uma biblioteca deseja manter informações sobre seus livros. quer armazenar para os livros as seguintes características: ISBN, título, ano editora e autores deste livro. Para os autores, deseja manter: nome e nacionalidade. Cada livro da biblioteca pertence a uma categoria. A biblioteca quer manter um cadastro das categorias existentes, com informações como: código da categoria e descrição.
Tarefa 9	Uma firma vende produtos de limpeza, e deseja melhor o controle dos produtos que vende. Cada produto é caracterizado por um código, nome do produto, categoria, e seu preço. A categoria é uma classificação criada pela própria firma. Cada cliente é identificado por um código, nome, endereço, telefone, status (bom, médio, ruim), e o seu limite de crédito. Cada pedido possui um número e a data de elaboração do pedido. Cada pedido pode envolver de um a vários produtos.
Tarefa 10	Uma floricultura deseja informatizar suas operações. deseja manter um cadastro de todos os seus clientes, mantendo informações como: RG, nome, telefone e endereço. Deseja cadastrar informações sobre os produtos que vende, tais como: nome do produto, tipo, preço e quantidade em estoque. Quando uma compra é realizada é armazenada informações como: cliente, data, valor total e produtos.

5.1.2 Variáveis Controladas e Contexto

Cada variável representa um tipo de valor sendo calculado, seria a métrica para representar os valores do contexto específico, cada variável possuí uma fórmula diferente de adquirir seus resultados, portanto é de extrema importância destacá-los. As variáveis sendo calculadas neste estudo de caso são as de precisão e de *recall*. A precisão ou também chamado de confiança, indica a proporção de casos positivos previstos que corretamente são positivos reais (POWERS, 2020). Já o *recall* ou também conhecido como sensitividade, representa a proporção de verdadeiros positivos que as abordagens corretamente previram (POWERS, 2020).

Já o contexto representa os sistemas utilizados para realizar o experimento, que no caso, é caracterizado pelo Entidare e o ChatGPT com o intuito de comparar a abordagem proposta em relação ao que o ChatGPT é capaz de fazer com as mesmas tarefas. Ambos os sistemas foram aplicados sobre as mesmas tarefas, com os mesmos enunciados e em relação destes enunciados elaborar diagramas de classe da UML, definindo as relações, atributos e entidades para cada uma destas tarefas. Já as métricas de Atributos, Entidades e Relações representam os resultados em percentagem, no qual vão de 0 à 1, onde 0,5 representa 50% e as três métricas são de extrema importância para o contexto de acurácia e *recall* obtidos pela abordagem proposta em comparação com o ChatGPT.

5.1.3 Análise dos Resultados

Análise comparativa de precisão e *recall*: Na Tabela 3 se encontra os resultados obtidos nas 10 tarefas. Na primeira tarefa, o destaque se encontra na precisão obtida pela abordagem proposta em quesito as relações, do qual obteve 100% do valor de precisão, também se destaca os 75% obtidos pela abordagem no tema entidades e, por fim, os 50% obtidos pelo ChatGPT no que se refere aos atributos. Já os valores referentes ao *recall* foram semelhantes, apenas se destacando os 100% obtidos pela abordagem sobre as entidades. Já a terceira tarefa acabou por ser uma no qual o ChatGPT conseguiu se sair superior tanto na precisão quanto no *recall*, contendo apenas um empate, na precisão de relações, com um valor de 50% entre os dois contextos sendo aplicados.

Na tabela 3 a quinta tarefa teve um resultado espetacular para a abordagem proposta, obteve 100% de precisão tanto em atributos quanto em entidade, mas apenas 50% em relações. Já o ChatGPT acabou não obtendo um resultado favorável, nos valores de precisão conseguindo apenas 54% para atributos, 27% para entidades e 29% para relações, isso devido ao fato de acabar declarando entidades e atributos demasiados, fato que causou a precisão decair a percentagem, o que se percebe nos resultados de *recall*, onde não se utiliza os valores declarados errados para seu cálculo, onde o ChatGPT conseguiu valores favoráveis. É possível observar que no quesito de *recall* o ChatGPT conseguiu resultados superiores no que tange os atributos, alcançando em 8 tarefas 100% de *recall*, já os resultados de *recall* sobre entidades e relações obtidas pelo En-

tidare, foram equivalentes ao ChatGPT, e onde a abordagem proposta conseguiu se sobressair perante o ChatGPT foi nos valores de precisão obtidos na definição de relações, e adquirindo valores semelhantes, mas ainda assim, superiores ao ChatGPT em atributos e entidades. E, por fim, com os resultados dos cálculos obtidos é possível estabelecer que para estas questões em específico, em média, o Entidare conseguiu se sair melhor do que o ChatGPT, destacando-se os resultados de precisão alcançados no levantamento de relações por parte do Entidare.

Na Tabela 3 foi utilizado para o Entidare os resultados obtidos aplicando a metodologia de agrupar as entidades do qual as palavras são sinônimas ou conjugações diferentes da mesma palavra em uma entidade apenas, ex: as entidades "Clientes"e "Cliente", são atados em uma só entidade chamada "Clientes". Foi aplicado esta metodologia após perceber que os resultados para a precisão de entidades ainda estava muito baixa ao não fazer esta junção. Para aplicar esta lógica foi utilizada uma biblioteca do Java chamada GroupDocs no qual busca dicionários para verificar sinônimos e conjugações diferentes de uma mesma palavra. ([3]).

Tabela 3 – Análise comparativa de precisão e *recall*

Tarefas	**Variáveis**	**Contexto**	**Atributos**	**Entidades**	**Relações**
Tarefa 1	Precisão	ChatGPT	0,5	0,27	0,36
		Entidare	0,36	0,75	1
	Recall	ChatGPT	0,38	0,5	0,8
		Entidare	0,38	1	0,8
Tarefa 2	Precisão	ChatGPT	0,83	0,83	0,67
		Entidare	0,17	0,71	0,33
	Recall	ChatGPT	1	1	0,5
		Entidare	0,6	1	0,5
Tarefa 3	Precisão	ChatGPT	1	1	0,5
		Entidare	0,88	0,6	0,5
	Recall	ChatGPT	1	1	1
		Entidare	0,88	1	1
Tarefa 4	Precisão	ChatGPT	0	0,5	0,33
		Entidare	1	0,75	0,67
	Recall	ChatGPT	1	1	1
		Entidare	1	1	1
Tarefa 5	Precisão	ChatGPT	0,54	0,27	0,29
		Entidare	1	1	0,5
	Recall	ChatGPT	0,5	1	1
		Entidare	0,93	1	1
Tarefa 6	Precisão	ChatGPT	0	0,75	0,6
		Entidare	1	1	1
	Recall	ChatGPT	1	0,6	0,6
		Entidare	1	0,6	0,4
Tarefa 7	Precisão	ChatGPT	0,75	1	0
		Entidare	1	1	1
	Recall	ChatGPT	1	0,75	0
		Entidare	0,92	0,75	0,25
Tarefa 8	Precisão	ChatGPT	0,78	0,75	0,33
		Entidare	0,78	1	1
	Recall	ChatGPT	1	0,75	0,5
		Entidare	1	0,5	0,5
Tarefa 9	Precisão	ChatGPT	0,87	0,8	0,14
		Entidare	0,92	1	0,67
	Recall	ChatGPT	1	0,8	0,2
		Entidare	0,92	0,6	0,4
Tarefa 10	Precisão	ChatGPT	0,77	0,75	0,2
		Entidare	1	1	0,33
	Recall	ChatGPT	1	0,75	0,25
		Entidare	1	0,75	0,25

[3]Biblioteca utilizada : https://products.groupdocs.com/search/

Análise estatística descritiva e teste de hipótese: A Tabela 4 demonstra os valores alcançados, utilizando os resultados obtidos nas 10 tarefas realizadas na Tabela 3 para obter a precisão e *recall* de atributos, entidades e relações para a elaboração de diagramas de classe da UML.

Tabela 4 – Estatística descritiva e teste de hipótese

Var	Tipo	Contexto	Desvio Padrão	Mín	25th	Mediana	75th	Máx	Média	Diff	Wilcoxon *p-value*	*Paired t-test* *p-value*
Precisão	Atributos	ChatGPT	0,33	0	0,37	0,76	0,84	1	0,60	+25,48%	0,24	0,23
		Entidare	0,28	0,17	0,68	0,96	1	1	0,81			
	Entidades	ChatGPT	0,25	0,27	0,44	0,75	0,88	1	0,69	+21,39%	0,09	0,09
		Entidare	0,15	0,60	0,74	1	1	1	0,88			
	Relações	ChatGPT	0,19	0	0,19	0,33	0,53	0,67	0,34	+51,06%	0,03	0,02
		Entidare	0,27	0,33	0,46	0,67	1	1	0,70			
Recall	Atributos	ChatGPT	0,22	0,38	0,88	1	1	1	0,89	-2,89%	0,59	0,70
		Entidare	0,20	0,38	0,81	0,93	1	1	0,86			
	Entidades	ChatGPT	0,17	0,50	0,71	0,78	1	1	0,82	+0,609%	1	0,94
		Entidare	0,19	0,50	0,60	0,88	1	1	0,82			
	Relações	ChatGPT	0,34	0	0,24	0,55	1	1	0,59	+4,10%	0,97	0,54
		Entidare	0,29	0,25	0,36	0,50	1	1	0,61			

Legendas: Var = Variáveis, Mín = Mínima, Máx = Máxima, Diff = Diferença

Na Tabela 4, observa-se que a abordagem proposta obteve médias melhores que o ChatGPT, em destaque as médias de precisão de relação e de entidades da Entidare, no qual conseguiram valores de 0,70 e 0,88 respectivamente. Já o desvio padrão ambos os contextos aplicados alcançaram valores relativamente baixos, mostrando que ambos conseguiram dados uniformes, o maior alcançado foi 0,34 pelo ChatGPT, tratando-se de valor de *recall* de relações. Após isso, falando sobre a mediana destaca-se o valor 1 obtido pelo Entidare, quando se tratava de precisão sobre entidades, e o valor 1 alcançado pelo ChatGPT quando se tratava de *recall* sobre atributos. E para finalizar merece ser mencionado que a menor mediana obtida foi pelo Entidare, quando o assunto era *recall* de relações, um valor de 0,50.

Para os resultados de percentagem da diferença, é possível notar um ganho razoável da Entidare em comparação com o ChatGPT, quando se trata de precisão para entidades, um percentual de 51,06% superior ao resultado do ChatGPT. Porém a abordagem proposta perde para o ChatGPT no quesito *recall* de atributos em um valor de 2,89%. Já os outros resultados de diferença, a Entidare foi superior ao ChatGPT, mas apenas a percentagem discutida anteriormente foi de um valor razoavelmente significativo para destaque.

Por fim, sobre os resultados dos testes de hipótese, e considerando o teste de hipótese nula com os resultados obtidos, percebe-se que apenas quando o assunto se trata de precisão sobre relações a abordagem proposta conseguiu recusar a hipótese nula com um valor de p igual a 0,03 para o teste de Wilcoxon, e um valor de p igual a 0,02 para o teste t pareado. Definindo que para ter um significado estatístico p precisa ter um valor menor que 0,05, o caso mencionado anteriormente é o único que acaba por atingir essa meta.

Resultado do Estudo de Caso: Com os resultados dos cálculos alcançados é possível afirmar que a Entidare conseguiu resultados com um diferencial positivo nas questões de levantar a precisão de atributos, entidades e relações, principalmente para as médias (0,81, 0,88 e 0,70 respectivamente) e as diferenças (25,48%, 21,39% e 51,06% respectivamente) alcançadas, em comparação com o ChatGPT. Com estes resultados, é possível definir que, para estas questões, a abordagem proposta favorece e possivelmente facilitaria aprendizes na elaboração de diagramas de classe da UML.

5.2 Avaliação de Aceitabilidade

Esta seção apresenta uma avaliação qualitativa do modelo inteligente proposto através da aplicação do modelo de aceitação de tecnologia (TAM) (GRANIĆ; MARANGUNIĆ, 2019) com profissionais da indústria. Para isso, um protocolo de pesquisa foi definido através da elaboração de um processo experimental (Seção 5.3.1), um questionário (Seção 5.2.2), e apresentação dos resultados obtidos (Seção 5.2.3).

5.2.1 Processo Experimental

A figura 6 representa o processo experimental no qual foi dividido em duas fases com uma lista de funções para cada uma, a primeira fase foi identificada como a identificação dos usuários da aplicação, chamada de encontrando o público alvo. Já a segunda fase foi chamada de análise dos resultados, essas fases serão explicadas a seguir. A metodologia aplicada para o processo experimental foi baseada e validada por estudos realizados previamente na literatura((FARIAS et al., 2015); (FARIAS, 2016); (VIEIRA; FARIAS, 2020a)).

- **Fase 1: Encontrando o público alvo.** Esta fase é definida como a fase de apresentação ao usuário sobre a abordagem proposta, é feito um treinamento com um vídeo explicativo sobre como o sistema funciona e como realizar os processos de identificação de entidades, relações e atributos da abordagem, assim como o video ajuda a explicar a utilidade e se os usuários conseguiram compreender o formato e a técnica utilizada pela abordagem proposta.

- **Fase 2: Aplicando e analisando os resultados.** Esta fase representa dois processos separados. O primeiro é importante para a reunir informações sobre o perfil dos usuários através de uma série de perguntas, o resultado é a saída conjunta dos dados respondidos por todos os usuários participantes. Já o segundo processo representa a aplicação do questionário TAM, onde os usuários também respondem uma série de perguntas sobre a percepção de utilidade, a facilidade de manuseio e a intenção de comportamento da abordagem proposta. A saída é os dados respondidos pelos participantes sobre a aceitação e a usabilidade percebida pelos usuários. Os usuários respondem a todos os questionários

e executam as duas fases para evitar qualquer tipo de incoerência durante todo o processo experimental sendo realizado.

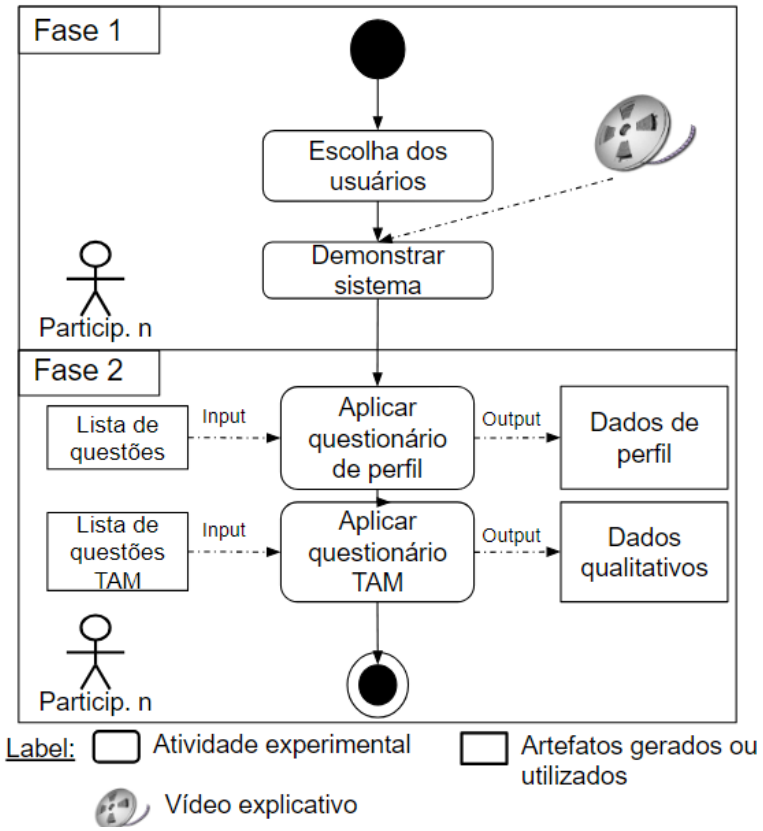

Figura 6 – Processo experimental aplicado

5.2.2 Questionário TAM

Foi realizado este questionário de forma online, em duas partes [4] e [5] (criados no *Google Forms*). O questionário foi elaborado e dividido em duas partes, cada uma com uma funcionalidade distinta para a avaliação. A primeira parte do questionário busca definir um perfil dos participantes relacionado ao conhecimento gerais de *software*. Já a segunda parte do questionário tem como objetivo avaliar a aplicação em um contexto de dados qualitativos. As partes deste questionário mencionado são detalhados a seguir.

- **Parte 1: Perfil dos participantes.** O objetivo principal desta parte do questionário é de estabelecer um perfil de usuários referentes as suas opiniões e características pessoais, e com isso analisar usuários que realmente são potenciais para utilizarem a abordagem proposta. As perguntas aplicadas portanto foram para coletar dados como nível de escolaridade, sexo, e nome caso o usuário deseje informar. Foi levantado também a experiência

[4] Etapa de perfil dos participantes: https://forms.gle/QHPHva2HBuMpgekR9
[5] Etapa do questionário TAM: https://forms.gle/SCapH54JTaE5yr868

que o usuário têm com o inglês, assim como experiência de manutenção e desenvolvimento de *softwares* tanto como usuário como de desenvolvedor. A construção de perfis de usuários foi realizada por entender-se que é de extrema importância avaliar se a abordagem sendo proposta foi avaliada por futuros usuários da abordagem.

- **Parte 2: Questionário TAM.** Já a segunda parte do questionário busca avaliar a aceitação da abordagem proposta e a usabilidade em dados qualitativos, através do Modelo de Aceitação de Tecnologia (TAM). O questionário foi separado em três categorias para avaliar a intenção do usuário no quesito de aceitabilidade da abordagem proposta e ao todo o questionário possuí 6 afirmações. As 3 primeiras afirmações são referentes a categoria de facilidade de uso da abordagem proposta, já dá quarta a quinta questão foi apurado o nível de percepção da utilidade da abordagem e por fim, a última afirmação questiona o comportamento do usuário em validar a abordagem, se pretende utilizá-la ou não. Foi utilizada a escala Likert para definir as respostas, onde os usuários são capazes de classificar como "Concordo Totalmente", "Concordo Parcialmente", "Neutro", "Discordo Parcialmente"e "Discordo Totalmente".

5.2.3 Análise de Resultados

Análise do perfil dos participantes: A Tabela 5 apresenta os valores obtidos no questionário sobre o perfil dos usuários, no qual descreve opiniões e aspectos dos participantes. Os dados foram coletados no período do dia 06 de novembro até 19 de novembro de 2023. No total, 12 participantes realizaram os questionários.

Considerando os resultados, a idade dos participantes ficou entre 20 e 49 anos, onde 33,3% têm 30 a 39 anos, já 16,7% está com 40 a 49 anos e, por fim, 50% se encontra na faixa etária dos 20 aos 29 anos. Já no nível de escolaridade, 58,3% são estudantes de graduação ou formados em graduação, e apenas 41,7% são estudantes de mestrado ou formados em mestrado. Todos os entrevistados têm conhecimento na área da computação.

A maioria dos participantes dispõe de experiência com desenvolvimento de *software*, e a maior parcela contendo 50% dos participantes possuí 7 anos ou mais de experiência, 25% dos participantes têm 2 a 4 anos de experiência, já 16,7% dos participantes conta com 4 a 6 anos de experiência, e 8,3% está abaixo de 2 anos de experiência. Já sobre a experiência com manutenção de *software*, a maior parcela conta com 33,3% com 7 anos ou mais de experiência, já 50% está dividido entre 4 a 6 anos e 2 a 4 anos de experiência, e 16,7% está abaixo de 2 anos de experiência. Analisando os resultados da experiência dos participantes com desenvolvimento de diagramas de classe UML, observa-se que a maior parte dos participantes se encontra entre 2 a 4 anos de experiência ou abaixo de 2 anos de experiência, obtendo 33,3% e 41,7% res-

Tabela 5 – Resultados referentes ao perfil dos usuários

Característica e opinião (n = 12)	Resposta	#	%
Idade	Menor que 20 anos	0	0
	20-29 anos	6	50%
	30-39	4	33,3%
	40-49	2	16,7%
	Acima de 49 anos	0	0
Nível de escolaridade	Graduação	7	58,3%
	Mestrado	5	41,7%
	Doutorado	0	0
	Outros	0	0
Nível de conhecimento em inglês	Avançado	5	41,7%
	Intermediário	5	41,7%
	Iniciante	2	16,7%
Experiência com desenvolvimento de *software*	Abaixo de 2 anos	1	8,3%
	2-4 anos	3	25%
	4-6 anos	2	16,7%
	Acima de 6 anos	6	50%
Experiência com manutenção de *software*	Abaixo de 2 anos	2	16,7%
	2-4 anos	3	25%
	4-6 anos	3	25%
	Acima de 6 anos	4	33,3%
Experiência com desenvolvimento de diagramas UML	Abaixo de 2 anos	5	41,7%
	2-4 anos	4	33,3%
	4-6 anos	0	0
	Acima de 6 anos	3	25%
Sistemas com aprendizado de máquina poderiam auxiliar no desenvolvimento de diagramas UML	Sim	12	100%
	Não	0	0

pectivamente. Já apenas 25% dos participantes conta com 7 anos ou mais de experiência em desenvolvimento de diagramas de classe UML.

Sobre a questão do conhecimento de nível de inglês, 41,7% dos participantes informou ter um conhecimento avançado no assunto, já 41,7% diz ter um conhecimento intermediário da língua inglesa, e por fim 16,7% informa ter pouco conhecimento, essa pergunta sobre o inglês é pertinente ao estudo pelo motivo da abordagem proposta retornar os resultados em inglês. Todos os participantes concordam com a questão se sistemas com aprendizado de máquina poderiam auxiliar no desenvolvimento de diagramas de classe UML. Por fim, foi realizado o questionário com uma pequena amostra de participantes, mas apropriada para um parecer inicial da abordagem proposta.

Análise do questionário TAM: A Figura 7 apresenta as informações e resultados coletados no questionário. Ao aplicar o questionário TAM foi reunido informações sobre intenção de comportamento, percepção de utilidade e facilidade de uso, no que se refere a utilização da abordagem sendo proposta.

Iniciando pela facilidade de uso percebida pelos participantes, 91,7% dos entrevistados concordam que a abordagem proposta é fácil de aprender, onde 50% concordam totalmente e 41,7% concordam parcialmente, e apenas 8,3% responderam que são neutros sobre esta questão, já na pergunta sobre achar a abordagem fácil de usar, 100% dos entrevistados concordam que a abor-

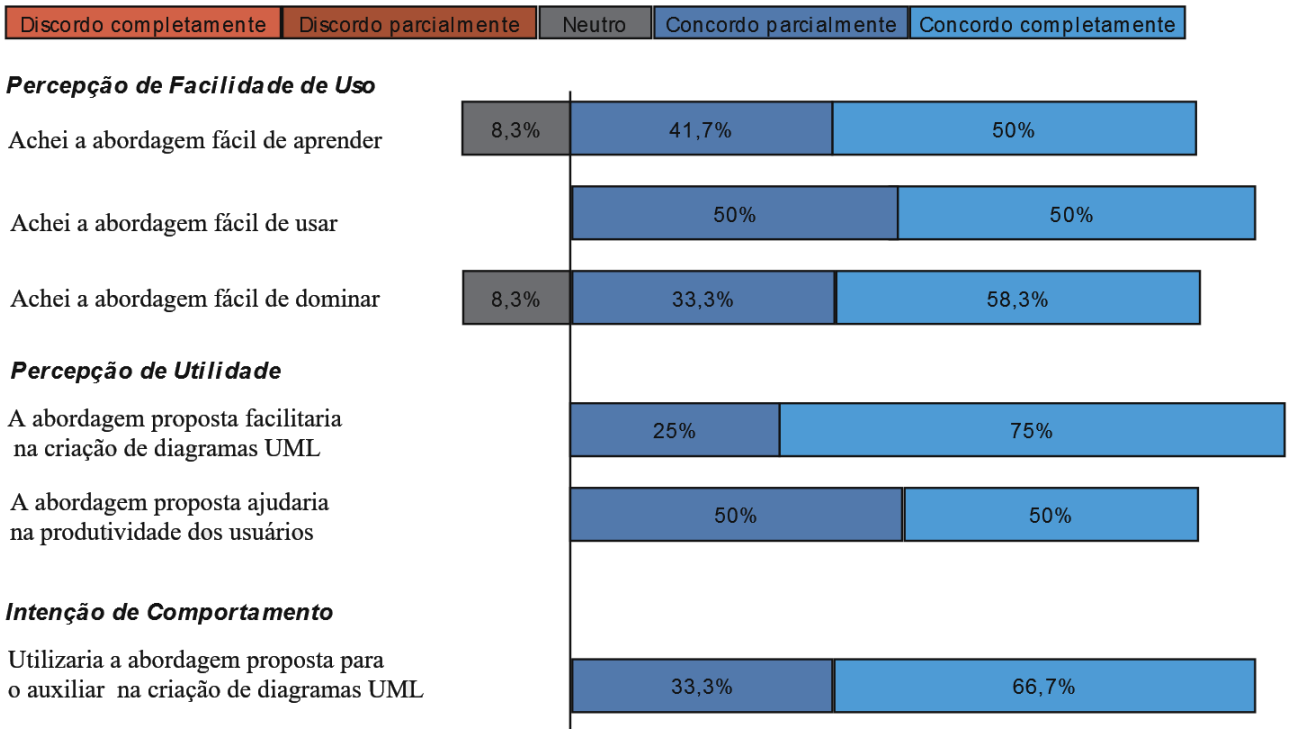

Figura 7 – Resultados relativos ao questionário TAM

dagem é fácil de usar, no qual 50% concordam totalmente e 50% concordam parcialmente, o que demonstra que não necessitaria de demasiado esforço para utilizar as funcionalidade apresentadas pela abordagem proposta, sobre a pergunta de achar a abordagem fácil de dominar, 91,6% dos entrevistados concordam que acharam a abordagem fácil de dominar, neste caso, 58,3% concordam totalmente e 33,3% concordam parcialmente, e apenas 8,3% dos entrevistados ficou neutro sobre esta questão.

No tocante à percepção de utilidade, todos os entrevistados concordam que a abordagem proposta facilitaria na criação de diagramas de classe da UML, com 75% dos participantes concordam totalmente e 25% dos participantes concordam parcialmente. Já na questão da abordagem proposta ajudar na produtividade dos usuários, todos os participantes concordam, no qual 50% concordam totalmente e 50% concordam parcialmente. Estes resultados demonstram uma aceitação boa em utilizar a abordagem proposta tanto para a produtividade quanto para a facilitação da criação de diagramas de classe da UML. Sobre a intenção de comportamento, todos os participantes concordam que teriam intenção de usar a abordagem proposta para o auxílio na definição de entidade, relações e atributos de enunciados para diagramas, onde 66,7% concordam totalmente e 33,3% concordam parcialmente. Por fim, os dados coletados e avaliados sugerem que a abordagem proposta tem um potencial para ser aceito por pessoas com um perfil semelhante ao dos participantes. Os resultados demonstram esse potencial de utilizar o modelo para auxiliar na definição de entidades, relações e atributos para o desenvolvimento de diagramas de classe da UML. Portanto, os questionários foram aplicados com uma amostra pequena de participantes, mas considerada adequada para uma análise inicial da abordagem

proposta.

Resultado da avaliação TAM: Os dados coletados e avaliados sugerem que a abordagem proposta tem um potencial para ser aceito por pessoas com um perfil semelhante ao dos participantes. O Entidare contempla as intenções de comportamento dos usuários, obtendo 100% neste tópico, assim como conseguindo 100% quando o assunto é facilitar a criação de diagramas de classe da UML, isso demonstra que o Entidare tem potencial para auxiliar na definição de entidades, relações e atributos de diagramas de classe da UML.

5.3 Discussão

Esta seção aborda os principais tópicos para discussão, identificados no desenvolvimento da abordagem. Percebidos através da análise dos resultados do estudo de caso, identificando os possíveis pontos de ampliação do Entidare. Abaixo serão detalhados algumas destas discussões.

Sensitividade do Contexto e Domínio do Problema. A identificação de contexto de um enunciado para elaboração de diagramas de classe da UML é um grande problema para o ChatGPT (CÁMARA et al., 2023) e também para o Entidare. Uma percepção obtida nos resultados analisados é de que quanto mais informações de contexto o enunciado necessitar para gerar as entidades, relações e atributos de um diagrama, maior a dificuldade da abordagem proposta e do ChatGPT de conseguir definir corretamente todas as variáveis necessárias. Conforme analisado nos resultados, é imperativo buscar melhorias na identificação de contexto para facilitar o processo de levantamento dos requisitos e com isso melhorar o processo de aprendizagem sobre diagramas de classe da UML.

Tecnologia de Inteligência Artifical no Aprendizado. Com o aumento do uso de ferramentes no qual fazem uso de inteligência artificial, é de extremo valor buscar avaliar o uso desta tecnologia no ramo educacional. Esta abordagem realizou um estudo sobre a utilidade desta tecnologia para o levantamento de diagramas de classe da UML, e com os resultados obtidos é possível expandir o contexto aplicado pelo *machine learning* para especializar esta abordagem no contexto de levantamento de requisito para diagramas de classe da UML. É percebido que simplificando a aplicação destas ferramentas e disponibilizando de forma prática propicia uma facilidade no uso.

Identificação de Cardinalidade em Enunciados. A identificação de cardinalidade através de um enunciado é um problema tanto para o Entidare quanto para o ChatGPT, porém o ChatGPT até consegue levantar de forma básica as cardinalidades de certos enunciados. O maior problema gerado pela cardinalidade é de que cardinalidades N X N em um diagrama de classe da UML geram uma nova classe de entidade através desta relação. Com os resultados obtidos, é possível observar que quando um enunciado não trata sobre esta entidade gerada através de uma cardinalidade N X N, tanto o ChatGPT quanto a abordagem proposta não foram capazes de identificar esta nova classe, isso acaba por gerar falhas no levantamento das entidades do diagrama. Uma possível solução foi identificada durante a execução deste trabalho, uma

de melhoria do enunciado antes de aplicar tanto no ChatGPT quanto para o Entidare, e outra de buscar na abordagem identificar as cardinalidades e com isso tentar definir uma nova entidade com essa identificação.

6 CONCLUSÃO E TRABALHOS FUTUROS

A utilização de inteligência artificial para diversas etapas vem sendo adotada frequentemente nos últimos tempos, e provavelmente seguirá sendo adotada, a aplicação de *machine learning* é outro processo que já está no mercado por um bom tempo e com o aumento dos modelos de linguagens naturais, encontrar maneiras de melhorar o processo sempre serão necessárias. Este artigo propôs uma abordagem de recomendar entidades e de utilizar inteligência artificial para complementar as entidades, relações e atributos para a elaboração de diagramas de classe UML. Conta com a funcionalidade tanto de *machine learning*, como de *prompt engineering* para a comunicação com o ChatGPT, como inteligência artificial. Foi implementado uma tela de *dashboard* para demonstrar os resultados da aplicação, principalmente da etapa de *machine learning* executada pela linguagem Python e da orientação a objetos aplicado pelo Java.

Ao analisar os resultados, observa-se como uma das contribuições científicas desta abordagem é de que usuários do perfil no qual este estudo foi aplicado, conseguem encontrar utilidade em uma ferramenta que utilize *machine learning* para auxiliar na definição de entidades, relações e atributos com intenção de criar diagramas de classe da UML. Outra contribuição é o fato de a abordagem proposta utilizar uma arquitetura baseada em componentes, o que possibilita o modelo aplicado ser capaz de ser estendido com novos componentes de comunicação e adaptados para outras funcionalidades e trabalhos, ou até outras metodologias de *machine learning* por diversas outras linguagens. E, por fim, os resultados obtidos da análise comparativa demonstra que a abordagem proposta pode ser melhorada para conseguir melhores resultados,e de que para uma primeira avaliação em comparação com o ChatGPT, a abordagem proposta alcançou resultados razoáveis.

Pensando em trabalhos futuros e funcionalidades futuras, a mais essencial seria continuar o que foi planejado no início da abordagem, de integrar o ChatGPT e o utilizá-lo como API para consultar também as entidades, relações e atributos, e com isso possibilitar o usuário de receber a resposta oferecida tanto pelo *machine learning* da aplicação, como da resposta oferecida pelo ChatGPT ou então até fazer uma tomada de decisão para qual oferecer ao usuário. Melhorar o *dataset* com mais questões de enunciados, tanto do ENADE como de outros para exercícios, e possibilitar uma variação de épocas de treinamento de acordo com a quantidade de enunciados inseridos no *dataset*. Outro ponto seria melhorar a identificação. principalmente na questão de relações, e quais entidades estão relacionadas com quais. Assim como aplicar os questionários aplicados nesta abordagem atual com uma quantidade maior de participantes, com isso obtendo resultados mais robustos em questão da aceitação da abordagem proposta. E, por fim, buscar se

aprofundar melhor em questões de *machine learning* para tentar melhorar o algoritmo utilizado na linguagem Python para obter resultados mais robustos em questões de entidades, relações e atributos para serem levantados, assim como tentar levantar a cardinalidade nas relações definidas.

Referências

ALAM, T.; KHAN, A.; ALAM, F. **Bangla Text Classification using Transformers**. 2020.

BERGSTRÖM, G. et al. Evaluating the layout quality of uml class diagrams using machine learning. **Journal of Systems and Software**, Elsevier, v. 192, p. 111413, 2022.

BI, Q. et al. What is machine learning? a primer for the epidemiologist. **American journal of epidemiology**, Oxford University Press, v. 188, n. 12, p. 2222–2239, 2019.

BRAşOVEANU, A. M. P.; ANDONIE, R. Visualizing transformers for nlp: A brief survey. p. 270–279, 2020.

BUKHARI, M. et al. An intelligent model for predicting the students' performance with backpropagation neural network algorithm using regularization approach. **Human-centric Computing and Information Sciences**, v. 12, 09 2022.

CABANE, H.; FARIAS, K. On the impact of event-driven architecture on performance: An exploratory study. **Future Generation Computer Systems**, Elsevier, 2023.

CÁMARA, J. et al. On the assessment of generative ai in modeling tasks: an experience report with chatgpt and uml. **Software and Systems Modeling**, Springer, p. 1–13, 2023.

COMBEMALE, B.; GRAY, J.; RUMPE, B. Chatgpt in software modeling. **Software and Systems Modeling**, Springer, p. 1–3, 2023.

DOMíNGUEZ-JIMéNEZ, J. et al. A machine learning model for emotion recognition from physiological signals. **Biomedical Signal Processing and Control**, v. 55, p. 101646, 2020. ISSN 1746-8094. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1746809419302277>.

FARIAS, K. Empirical evaluation of effort on composing design models. **arXiv preprint arXiv:1610.09012**, 2016.

FARIAS, K. et al. Evaluating the effort of composing design models: a controlled experiment. **Software & Systems Modeling**, Springer, v. 14, p. 1349–1365, 2015.

GONÇALES, L. J. et al. Comparison of software design models: An extended systematic mapping study. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 52, n. 3, p. 1–41, 2019.

GRANIĆ, A.; MARANGUNIĆ, N. Technology acceptance model in educational context: A systematic literature review. **British Journal of Educational Technology**, Wiley Online Library, v. 50, n. 5, p. 2572–2593, 2019.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine learning and deep learning. **Electronic Markets**, Springer, v. 31, n. 3, p. 685–695, 2021.

JI, Z.; WEI, Q.; XU, H. Bert-based ranking for biomedical entity normalization. v. 2020, p. 269–277, 2020. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7233044/>.

JÚNIOR, E.; FARIAS, K. Modelgame: A quality model for gamified software modeling learning. In: **15th Brazilian Symposium on Software Components, Architectures, and Reuse**. [S.l.: s.n.], 2021. p. 100–109.

LI, L.; ZHANG, Y.; CHEN, L. Personalized transformer for explainable recommendation. **arXiv preprint arXiv:2105.11601**, 2021.

LIU, P. et al. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. **ACM Computing Surveys**, ACM New York, NY, v. 55, n. 9, p. 1–35, 2023.

POWERS, D. M. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. **arXiv preprint arXiv:2010.16061**, 2020.

REN, R. et al. Using the socio chatbot for uml modelling: A family of experiments. **IEEE Transactions on Software Engineering**, IEEE, v. 49, n. 1, p. 364–383, 2022.

RUBERT, M.; FARIAS, K. On the effects of continuous delivery on code quality: A case study in industry. **Computer Standards & Interfaces**, Elsevier, v. 81, p. 103588, 2022.

RUKMONO, S. A.; CHAUDRON, M. R. Guiding peer-feedback in learning software design using uml. p. 122–133, 2022.

SANCHEZ, S. A.; ROMERO, H. J.; MORALES, A. D. A review: Comparison of performance metrics of pretrained models for object detection using the tensorflow framework. **IOP Conference Series: Materials Science and Engineering**, IOP Publishing, v. 844, n. 1, p. 012024, may 2020. Disponível em: <https://dx.doi.org/10.1088/1757-899X/844/1/012024>.

SHRIVASTAVA, M.; KUMAR, S. A pragmatic and intelligent model for sarcasm detection in social media text. **Technology in Society**, v. 64, p. 101489, 2021. ISSN 0160-791X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0160791X20312926>.

SINGH, S.; MAHMOOD, A. The nlp cookbook: Modern recipes for transformer based deep learning architectures. **IEEE Access**, v. 9, p. 68675–68702, 2021.

STöRRLE, H. On the impact of layout quality to understanding uml diagrams: Diagram type and expertise. p. 49–56, 2012.

URDANGARIN, R. G.; FARIAS, K.; BARBOSA, J. Mon4aware: A multi-objective and context-aware approach to decompose monolithic applications. In: **XVII Brazilian Symposium on Information Systems**. [S.l.: s.n.], 2021. p. 1–9.

VIEIRA, R. D.; FARIAS, K. Cognide: a psychophysiological data integrator approach for visual studio code. In: **Proceedings of the XXXIV Brazilian Symposium on Software Engineering**. [S.l.: s.n.], 2020. p. 393–398.

VIEIRA, R. D.; FARIAS, K. Usage of psychophysiological data as an improvement in the context of software engineering: A systematic mapping study. In: **XVI Brazilian Symposium on Information Systems**. [S.l.: s.n.], 2020. p. 1–8.

WHITE, J. et al. A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv, 2023. Disponível em: <https://arxiv.org/abs/2302.11382>.

ZHAO, X. et al. Fine-tuning bert model for materials named entity recognition. p. 3717–3720, 2021.