

André Murillo Geraldo Corrêa

# Jupiterweb Chatbot

São Paulo

2018

André Murillo Geraldo Corrêa  
Engenharia Mecatrônica

# Jupiterweb Chatbot

Este relatório é apresentado como requisito parcial para a obtenção do grau de Engenheiro Mecatrônico da Escola Politécnica da Universidade de São Paulo, desenvolvido em conjunto com o departamento de Engenharia Mecatrônica e Sistemas Mecânicos (PMR).

Orientador: Prof. Dr. Fabio Gagliardi Cozman

São Paulo

2018

Este relatório é apresentado como requisito parcial para a obtenção do grau de Engenheiro Mecatrônico na Escola Politécnica da Universidade de São Paulo. É produto de meu próprio trabalho, exceto onde indicado no texto. O relatório pode ser livremente copiado e distribuído desde que a fonte seja citada.

#### Catálogo-na-publicação

Corrêa, André  
Jupiterweb Chatbot / A. Corrêa -- São Paulo, 2018.  
64 p.

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1.Chatbot 2.Inteligência Artificial 3.Agentes Conversacionais  
I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos II.t.

# AGRADECIMENTOS

Esse trabalho não seria possível sem o apoio das pessoas que me acompanharam ao longo do projeto. Gostaria de agradecer especialmente ao meu orientador, Prof. Dr. Fabio Gagliardi Cozman, por ter proposto esse tema tão interessante e pela forma como conduziu o trabalho. Sempre me deu autonomia para desenvolvê-lo, mas ao mesmo tempo aconselhou e direcionou minhas decisões para o caminho certo.

Também devo agradecer imensamente aos meus colegas de curso que me auxiliaram nos testes e avaliações do projeto. A comunidade de alunos da Escola Politécnica se mostrou extremamente solícita e empática, prestando um serviço de suma importância para a conclusão desse trabalho. Os incentivos e elogios foram muito bem recebidos. Mais uma vez, muito obrigado.

Finalizo agradecendo meus familiares e amigos próximos por todo o apoio e suporte que me proveram ao longo do ano. Sem eles eu não teria alcançado meus objetivos e é a eles que dedico este trabalho.

Muito obrigado,

André

## RESUMO

Nos últimos anos, tem-se observado uma mobilização das grandes corporações e altos investimentos em novas formas de interação com o consumidor. Dentre elas destaca-se o desenvolvimento de chatbots, que são programas capazes de interagir com os usuários sob forma de diálogo – seja escrito ou falado – com o intuito de buscar informações, auxiliar compras, tirar dúvidas, investir capital, entre outros. Ao mesmo tempo que a comunidade científica vem quebrando as fronteiras da análise de texto e processamento de linguagem natural, grandes empresas como Google, Facebook, Amazon, Microsoft e IBM têm investido fortemente nesse segmento.

O presente trabalho descreve a construção e implementação de um chatbot capaz de sugerir disciplinas de acordo com os interesses do usuário e de buscar informações como por exemplo horários de oferecimento e corpo docente responsável da disciplina. Tal sistema visa oferecer aos alunos uma alternativa à utilização do sistema tradicional de gestão acadêmica – Jupiterweb. Essa iniciativa tem como objetivo tornar mais agradável a tarefa de consolidação da grade curricular, através de uma interação baseada em diálogo, na qual boa parte da complexidade da tarefa fica a cargo do programa. Os experimentos realizados indicam que o chatbot possui uma boa capacidade de interpretação de texto, atingindo um valor de F1-score de 86,5% no teste de classificação de intenções. Além disso, a partir de uma pesquisa de satisfação com os usuários, estimou-se que, em 75% dos casos observados, o programa compreendeu e executou corretamente as tarefas solicitadas, atingindo um nível de satisfação geral de 78%. Os próximos passos do trabalho envolvem o refinamento da capacidade de resposta do robô e a possível integração do chatbot com o sistema Jupiterweb, aumentando assim o leque de funcionalidades e o valor gerado pelo programa.

**Palavras-chave:** Chatbot, Inteligência Artificial, Agentes Conversacionais

## **ABSTRACT**

In the last years we've seen a great mobilization and high investments in new ways of interacting with the consumer lead by big companies. Among them, we can highlight the development of chatbots, which are programs capable of interacting with the consumer through dialogue – written or spoken – with the objective of searching for information, helping with shopping, answering questions, investing capital, etc. While the scientific community is breaking the text analysis and natural language processing' barriers, big companies like Google, Facebook, Amazon, Microsoft and IBM are heavily investing in this segment.

This work describes the development and implementation of a chatbot capable of suggesting courses according to user's interests and capable of searching courses' information, like time schedule and teachers. The system intends to offer an alternative to the traditional system Jupiterweb. This initiative intends to make more pleasant the task of choosing the right course through an interaction based on dialogue in which the complexity of the task is mostly handled by the chatbot. The tests indicated that the chatbot has a good ability in text comprehension, reaching a F-1 Score value of 86,5% in the classification test. Besides that, through a user satisfaction survey, it was estimated that in 75% of the observations the program understood and executed correctly the tasks, reaching an overall satisfaction score of 78%. The next steps involve improving the robot's comprehension skills and possibly integrating the program with the Jupiterweb system, in order to boost its capabilities and its value.

**Keywords:** Chatbot, Artificial Intelligence, Dialog Agents

# SUMÁRIO

---

Introdução.....	11
1. Graphical User Interface.....	12
2. Agentes conversacionais.....	13
2.1. Task-oriented dialog agents .....	13
2.2. Chatbot .....	14
3. Sistema Jupiterweb .....	14
Objetivos e Motivação.....	16
Estado da Arte .....	18
Metodologia.....	22
1. Escolha do Framework de desenvolvimento .....	22
2. Construção da Base de Dados .....	24
3. Definição da Arquitetura .....	25
4. Especificação do Sistema .....	28
5. Modelo de Estados .....	31
Resultados.....	35
1. Análise do Diálogo .....	35
2. Análise de Performance.....	38
2.1. Metodologia .....	38
2.2. Resultados .....	40
3. Análise de Usabilidade .....	45
3.1. Metodologia .....	46
3.2. Descrição do Experimento .....	49
3.3. Resultados .....	50
4. Discussão e Continuação .....	56
Referências Bibliográficas.....	58
Apêndices .....	61
1. Apêndice A – Conjunto de dados do Teste de Classificação .....	61
2. Apêndice B – Conjunto de dados do Teste de Falsos Positivos.....	64

# LISTA DE FIGURAS

---

Figura 1 - Arquitetura base do chatbot .....	26
Figura 2 - Fluxo de processamento e resposta em um cenário ilustrativo.....	28
Figura 3 - Exemplo de busca de informações da disciplina. ....	29
Figura 4 - Exemplo de sugestão de disciplina. ....	29
Figura 5 - Continuação da conversa após sugestão de disciplina.....	30
Figura 6 - Modelo de estados da busca de informações de disciplinas. ....	31
Figura 7 - Modelo de estados da função de sugestão de disciplinas. ....	33
Figura 8 - Modelo de estados completo do sistema.....	34
Figura 9 - Exemplo completo de interação com o robô. ....	36
Figura 10 - Matriz de confusão do teste de classificação. ....	41
Figura 11 – Estrutura do framework PARADISE.....	47
Figura 12 - Resultado do cumprimento das tarefas solicitadas. ....	51
Figura 13 - Coeficiente de Pearson entre os parâmetros de interesse. ....	53
Figura 14 - Análise da colinearidade entre k e Rejeição. ....	54
Figura 15 - Análise do resíduo da regressão entre a variável de interesse e os parâmetros explicativos.....	55



## LISTA DE TABELAS

---

Tabela 1 – Matriz de comparação dos frameworks disponíveis. ....	23
Tabela 2 – Lista de informações disponibilizadas pelo chatbot. ....	38
Tabela 3 – Matriz de confusão de um classificador binário. ....	39
Tabela 4 – Métricas de avaliação de classificadores. ....	39
Tabela 5 – Cálculo de Accuracy e Error Rate para cada classe. ....	42
Tabela 6 – Valores de Precision, Recall e F1-Score para cada classe. ....	43
Tabela 7 – Métricas de custos da interação. ....	48
Tabela 8 – Pontuação das respostas categóricas. ....	50
Tabela 9 – Parâmetros de custo da interação levantados no experimento. ....	50
Tabela 10 – Matriz de parâmetros do experimento. ....	52
Tabela 11 – Resultado da regressão múltipla. ....	55

# LISTA DE ABREVIACOES E ACRONIMOS

---

AIML	Artificial Intelligence Markup Language
API	Application Programming Interface
FAQ	Frequently Asked Questions
IA	Inteligncia Artificial
JSON	JavaScript Object Notation
MRS	Mean Recognition Score
MQO	Mnimos Quadrados Ordinrios
NLP	Natural Language Processing
ORM	Object Relational Mapper
SDK	Software Development Kit
USAT	User Satisfaction Score
USP	Universidade de So Paulo
XML	Extensible Markup Language

# INTRODUÇÃO

---

A man's character may be learned from the adjectives  
which he habitually uses in conversation.

Mark Twain

O desenvolvimento da linguagem foi o marco chave da evolução humana, principalmente do ponto de vista cultural e científico. Dentro desse domínio, o diálogo é provavelmente o mais fundamental aspecto da linguagem e é o primeiro a ser aperfeiçoado, ainda na fase da infância. É sob o diálogo familiar que se constrói aos poucos a personalidade e o caráter do indivíduo. Não é à toa que a primeira palavra pronunciada pela criança é vista como um momento tão celebre do seu crescimento. É certamente intrigante o ato de criar algo e ser capaz de dialogar com o mesmo. Dizem as lendas que, ao finalizar sua estátua de Moises, Michelangelo achou seu trabalho tão vívido que tocou levemente no joelho da figura e a ordenou que falasse! (JURAFSKY e MARTIN, 2018).

O diálogo é a forma de linguagem mais empregada nas atividades diárias, seja fazendo compras no mercado, organizando uma reunião ou discutindo o resultado do campeonato esportivo. Enquanto o ser humano é capaz de interagir sob condições de informações desestruturadas, mudança frequente de ritmo e contexto, as máquinas estão no lado oposto desse domínio. Estas preferem um ambiente onde a troca de informação ocorre de maneira estruturada, dentro de um escopo definido com o mínimo de ambiguidade possível. Não é surpreendente que até hoje a principal forma de interação entre homem e máquina é feita através de comandos diretos dentro de um conjunto de comandos possíveis. Apesar de serem formas de interação distintas, a crença de que as máquinas seriam capazes de compreender nossa língua e de que o homem poderia, portanto, interagir com estas através do diálogo, tem lugar privilegiado dentre os temas favoritos da ficção científica. Isto reflete uma convicção de que o diálogo seria a forma mais natural e potente de interface entre homem e máquina. É comum encontrarmos, dentro dessa vertente da literatura, máquinas comandadas por voz, agentes virtuais e robôs com inteligência artificial que simulam o comportamento humano. Tais temas vem há muito tempo despertando o interesse dos leitores e dos telespectadores.

Com os avanços da tecnologia, o que uma vez foi ficção científica, passa a estar cada vez mais próximo da realidade. Tem-se observado uma atenção especial a esse tema por parte tanto do meio científico quanto por parte do mundo corporativo. Enquanto os pesquisadores quebram as fronteiras da análise de texto e processamento de linguagem natural, grandes empresas também realizam altos investimentos neste meio, criando soluções de atendimento ao consumidor e resolução de problemas através de sistemas baseados em diálogo.

Nesta seção, ressaltaremos o contexto de desenvolvimento do trabalho tecendo uma breve discussão acerca da evolução nas formas de interação entre homem e máquina, bem como introduzindo os conceitos de *dialog agents* e *chatbots*.

## 1. Graphical User Interface

Atualmente, a forma com que nós buscamos e tratamos a informação na web está muito ancorada no conceito de GUI (*Graphical User Interface*). Essa tecnologia fornece uma plataforma na qual o usuário pode interagir com dispositivos eletrônicos através de ícones, símbolos, barras de rolagem, botões, imagens e hyperlinks. Um website é um exemplo típico desta interface no qual, através de elementos como campos de busca e barras de navegação, o usuário pode buscar informações, fazer o upload de imagens e salvar arquivos (NIELSEN, 2012).

Essa interface surgiu nos anos 70 com o sistema Xerox Alto, num contexto onde a CLI (*Command Line Interface*) era predominante. Esta última é caracterizada pela necessidade de introduzir os comandos diretamente no teclado e, portanto, necessita conhecimento técnico elevado e possui uma curva de aprendizado mais lenta. Uma vez que a quantidade de comandos possíveis pode ser alta e esses comandos em geral necessitam de uma série de parâmetros, a incidência de erros é maior e o trabalho é laborioso (ABHISHEK AGARWAL, 2014).

Em outras palavras, a interface gráfica surgiu como uma nova forma de interação com o usuário, mais simples e intuitiva, visando cobrir as deficiências conhecidas da interface via linha de comando no que diz respeito à facilidade de utilização e diminuição de erros. Tal interface teve papel fundamental na evolução da capacidade de interação entre homem e máquina, principalmente na popularização dos computadores e smartphones.

A utilização da interface gráfica visa facilitar o acesso do usuário aos comandos disponíveis no sistema, entretanto, cabe ainda a ele decidir quais comandos deve utilizar e como gerenciar a complexidade do sistema. Ou seja, o usuário é responsável por extrair as informações necessárias, processá-las, filtrá-las e agrupá-las, para assim obter as conclusões desejadas. Em sistemas muito extensos ou mal estruturados, essa pode ser uma tarefa bastante árdua e demorada, principalmente no cenário atual no qual a informação é encontrada em abundância, mas geralmente disposta sob forma desestruturada.

Nas últimas décadas, a forma como consumimos informação mudou drasticamente com o surgimento de novos meios de comunicação - redes sociais, fóruns, motores de busca, aplicações mobile, reconhecimento de voz, entre outros. Vivemos em um período onde a informação é abundante e acessível. A problemática atual deixa de estar relacionada ao acesso à informação, mas sim referente à dificuldade em encontrar informações pertinentes em meio a tantas outras. A verdade é que o desafio do século é muito mais amplo: refere-se à capacidade de cruzar uma vasta gama de informações, analisar tendências e propor sugestões aos usuários de forma sistematizada. Nesse contexto, a Inteligência Artificial, que vem progredindo intensamente nos últimos anos,

tem um papel fundamental. Com o surgimento de motores de busca cada vez mais robustos, agentes virtuais inteligentes e algoritmos sofisticados de aprendizado de máquina, percebe-se uma tendência na qual a complexidade de encontrar a informação, processá-la e apresentá-la fica sob responsabilidade da IA, e não mais do ser humano.

Assim como a interface gráfica foi um avanço tecnológico sobre a interface via linha de comando, visando cobrir as deficiências de dificuldade de utilização e incidência de erros, seria a interação através de diálogo e baseada em IA o próximo avanço na relação homem-máquina?

## 2. Agentes conversacionais

Na prática, nos últimos anos, tem-se observado uma mobilização das grandes corporações e altos investimentos em novas formas de interação com o consumidor, principalmente através de *dialog systems* e agentes conversacionais. Estes são programas que interagem com o usuário sob a forma de linguagem natural, seja ela falada ou escrita, com o intuito de entretê-los ou realizar alguma tarefa. Estes sistemas, em geral, se classificam em duas categorias: *task-oriented dialog agents* e chatbots.

### 2.1. Task-oriented dialog agents

*Task-oriented dialog agents* são sistemas projetados para executar tarefas específicas, dentro de um domínio definido, como por exemplo, agendar uma passagem aérea, informar a previsão do tempo, encontrar restaurantes próximos, investir capital ou pedir *delivery* de comida. Geralmente são projetados para simular conversas curtas, com foco em buscar as informações necessárias para completar a tarefa solicitada pelo usuário (JURAFSKY e MARTIN, 2018). Esse tipo de sistema é comumente encontrado nos agentes virtuais dos *smartphones* e computadores atuais, como Siri, Cortana e Alexa. Muitas empresas empregam esse tipo de sistemas em seus websites visando auxiliar os consumidores na busca de informações e na resolução de problemas. Uma prática bastante comum atualmente é transformar o típico FAQ em um sistema baseado em diálogo capaz de responder as mesmas perguntas frequentes, mas com uma característica mais natural, simulando um atendente pessoal do consumidor, melhorando assim a experiência do usuário e potencialmente fidelizando o cliente.

A empresa Baidu lançou recentemente o agente MELODY, um agente conversacional cujo objetivo é conversar com pacientes, realizar diagnóstico e fornecer orientações básicas. Esse sistema oferece aos médicos as informações dos pacientes quanto aos sintomas, uso de medicação contínua, histórico familiar, entre outros, agilizando o atendimento dos médicos e tornando a experiência do paciente mais agradável. Atualmente, o sistema está disponível unicamente na China, mas já tem sido utilizado por milhares de pacientes (LARSON, 2016).

DoNotPay é um agente conversacional que ajuda os usuários com procedimentos jurídicos, como recorrer a multas de trânsito ou solicitar o reembolso de bilhetes aéreos.

Mais de 250 mil casos já foram solucionados pelo agente, representando uma taxa de sucesso de 64% (KEACH, 2016).

## 2.2. Chatbot

**Chatbots**, por outro lado, são sistemas com um domínio de interesse muito mais amplo e visam produzir extensas interações, imitando as características de uma conversa entre indivíduos reais, como a desestruturação do texto e as frequentes transições de um assunto a outro. Esses sistemas são comumente colocados à prova em testes similares ao teste de Turing, para medir a verossimilhança com as conversas humanas. Além disso, tem geralmente um valor de entretenimento, ao invés de um objetivo de conclusão de tarefas, como nos sistemas anteriores. A gigante Microsoft desenvolveu recentemente o XiaoIce, chatbot que conversa com as pessoas nas plataformas sociais e possui um comportamento muito semelhante ao ser humano, no que diz respeito à capacidade de dialogar (MICROSOFT, 2014).

Apesar dessa distinção de agente conversacional pelo seu objetivo principal, tanto a mídia quanto o senso comum costumam rotular todo agente como um chatbot, sem fazer esta distinção. Apesar do projeto aqui apresentado se encaixar na definição de um *task-oriented dialog agent*, por uma questão de simplicidade e proximidade com o contexto midiático, não faremos essa distinção e o consideraremos simplesmente chatbot, ou seja, um agente capaz de simular uma conversa em linguagem natural e realizar as tarefas solicitadas.

## 3. Sistema Jupiterweb

O Jupiterweb<sup>1</sup> é o Sistema de Gestão Acadêmica da Pró-Reitoria de Graduação da USP e é um dos sistemas mais utilizados pelos integrantes da universidade. É neste sistema que os alunos realizam as inscrições nas disciplinas, acompanham a evolução no curso, fazem solicitações administrativas, consultam a grade horária, entre outras funcionalidades.

Esse é o portal que consolida as informações sobre as disciplinas e serve de suporte para que os alunos escolham adequadamente a sua grade curricular. No entanto, essa tarefa é especialmente complexa, uma vez que existe uma grande variedade de disciplinas e as ferramentas de busca disponibilizadas na plataforma têm uma funcionalidade limitada.

Na prática, os alunos buscam as disciplinas através de um conjunto de critérios, como, por exemplo, diagonal curricular, tema de interesse, horários disponíveis e quantidade de créditos. Porém, na maioria dos casos, esse sistema não permite uma busca exaustiva que englobe a totalidade dos critérios do aluno.

---

<sup>1</sup> <https://uspdigital.usp.br/jupiterweb>

A ferramenta de pesquisa de disciplinas permite a busca apenas pelo nome ou pela sigla da disciplina, o que não é a tarefa mais simples, a não ser que o aluno já saiba a matéria que está buscando. O usuário não consegue buscar as disciplinas dentro do tema Inteligência Artificial, por exemplo. Tampouco buscar as disciplinas ministradas por um professor específico. Em vez disso, o aluno deve traçar rotas alternativas para concluir essa tarefa, como partir da grade curricular de cursos relacionados e analisar uma a uma a descrição das disciplinas oferecidas até encontrar aquela que se encaixe no tema desejado.

Além disso, os resultados das buscas geralmente retornam listas de disciplinas e cabe ao usuário processar essa informação. Uma prática comum é a utilização de diversas abas no navegador de forma a visualizar e comparar diferentes disciplinas sem perder a pesquisa atual, o que sugere uma deficiência quanto à usabilidade do sistema.

Não é possível aplicar filtros sobre às informações retornadas nem cruzar os resultados para se obter uma pesquisa mais precisa. Ou seja, não seria possível, por exemplo, buscar uma disciplina que é oferecida de quarta-feira e pertencente ao tema Metalurgia. A maioria dos elementos condicionais – E, OU, MENOR, MAIOR, etc. – não se encaixam no sistema atual e não podem ser usados para realizar uma busca mais inteligente.

Apesar de ser um sistema completo quanto à disponibilidade de informações, a utilização do Jupiterweb para as tarefas descritas ainda é difícil. A falta de ferramentas mais complexas de busca e filtragem introduz limitações claras na utilização da plataforma. Toda a complexidade de encontrar a informação, processá-la, cruzá-la e agrupá-la ainda está a cargo do usuário. Não é raro o aluno passar diversas horas analisando cada disciplina para escolher aquela que mais o interessa, uma vez que a escolha da disciplina certa é ponto chave da formação do estudante. Assim, esse projeto visa dar um primeiro passo rumo a uma plataforma mais inteligente e uma interação com o usuário mais agradável, através do uso de ferramentas de processamento de linguagem natural e inteligência artificial.

# OBJETIVOS E MOTIVAÇÃO

---

O objetivo do presente trabalho é projetar e construir um chatbot capaz de auxiliar os alunos na tarefa de consolidação da grade curricular, atualmente realizada através do sistema de gestão acadêmica JupiterWeb. A proposição do projeto é oferecer aos alunos uma solução alternativa ao sistema tradicional, e que faz uso dos recentes avanços científicos no quesito processamento de linguagem natural para atribuir mais flexibilidade e inteligência à utilização do sistema. Através de uma interação baseada em diálogo, esta solução visa complementar o sistema tradicional e solucionar algumas deficiências quanto à usabilidade e às limitações na busca de disciplinas, além de prover ao usuário uma experiência de utilização agradável e familiar. Quanto mais natural for a conversa e quanto mais precisas forem as informações fornecidas pelo programa, maior será o nível de satisfação do usuário e, portanto, melhor será a performance do chatbot.

De forma prática, o programa deve ser capaz de atingir dois objetivos principais, sendo estes segmentados nas seguintes tarefas elementares.

1. Buscar informações de uma disciplina de escolha.
  - a. Fornecer a descrição da disciplina.
  - b. Fornecer os professores responsáveis.
  - c. Fornecer os horários de oferecimento.
  - d. Informar os requisitos da disciplina, se possuir.
  - e. Fornecer a quantidade de créditos.
  - f. Fornecer a carga-horária da disciplina.
  
2. Sugerir disciplinas a partir dos critérios de interesse do aluno.
  - a. Sugerir uma disciplina com base em um tema de interesse.
  - b. Sugerir uma disciplina dada uma preferência de professor.
  - c. Sugerir uma disciplina a partir de uma restrição de horário.

O programa deve ser capaz de interagir com sentenças e perguntas as quais ele não foi previamente treinado, identificando corretamente as intenções dos usuários em cada caso.

Além disso, o diálogo deve ser construído de tal forma que o usuário saiba, a todo momento da conversa, quais as opções de interação disponíveis e qual o escopo possível de perguntas. Caso o usuário solicite algo fora do escopo de ação do chatbot, este deve reagir adequadamente informando ao usuário as limitações da conversa e corrigindo o rumo do diálogo.

No contexto universitário, a linguagem coloquial é bastante frequente. Para se adequar ao público alvo, o chatbot deve ser capaz de extrapolar sentenças com erros



ortográficos, considerar abreviações e sinônimos, aumentando assim a fluidez da conversa e verossimilhança com uma interação humana.

De forma geral, o programa deve ser capaz de interpretar as sentenças em língua portuguesa e detectar as intenções dos usuários com precisão. Espera-se uma precisão de pelo menos 80% nesta competência. Finalmente, o programa deve ser regulado adequadamente para manter controlado o trade-off entre a precisão e a incidência de falsos positivos.

Vale ressaltar que este projeto também visa a verificação da viabilidade de se aplicar um tal sistema no contexto da universidade, através de pesquisas de satisfação e entrevistas com os usuários. Uma vez comprovado o valor gerado por tal sistema, pode-se atuar nos próximos passos do projeto, que envolvem a devida integração com outros sistemas da universidade e complementação das funcionalidades do robô. Para isso, é importante termos um modelo escalável de avaliação da performance do robô para então atuarmos sistematicamente nos fatores que afetam o seu desempenho.

Este trabalho tem primeiramente uma motivação social, buscando facilitar a realização de uma tarefa de suma importância no ciclo de vida do estudante universitário, que é a consolidação da grade curricular. As escolhas das disciplinas são decisivas no percurso do aluno e impactam diretamente na formação do profissional que será em breve incorporado ao mercado de trabalho.

Além disso, o projeto tem motivação científica e visa contribuir para o desenvolvimento de chatbots, cada vez mais capazes de compreender o diálogo humano. Dado que esta vertente de pesquisa é relativamente nova, existem diversos caminhos a serem explorados, principalmente para chatbots na língua portuguesa. Vale ressaltar também a vasta aplicabilidade destes bots, que podem ser utilizados em serviços de atendimento ao consumidor, serviços de agendamento, serviços de notícias, entre outros. Portanto, o trabalho aqui desenvolvido pode ser replicado em outros contextos além do universitário, gerando assim, ainda mais valor.

## ESTADO DA ARTE

---

Um chatbot é um programa de computador capaz de simular uma conversa inteligente com um ou mais indivíduos via recursos textuais e visuais, estruturados na forma de diálogo. Um dos primeiros chatbots foi desenvolvido por (WEIZENBAUM, 1966) e chama-se ELIZA. O funcionamento de ELIZA se baseia na decomposição das frases recebidas a partir de algumas regras que são ativadas de acordo com as palavras-chave que são encontradas no texto. Para ilustrar essa dinâmica, consideremos a seguinte frase como exemplo: “Eu estou muito infeliz”. A partir desse input, a resposta de ELIZA provavelmente seria: “A quanto tempo você está muito infeliz?”. Nesse caso em particular, a palavra-chave “Eu estou” desencadeou uma regra na qual o programa começa a resposta com “A quanto tempo você está” e termina a frase com tudo que sucedia a palavra-chave, nesse caso, a sentença “muito infeliz”. Ou seja, caso a entrada fosse “Eu estou com fome”, a resposta seria “A quanto tempo você está com fome?”. Devido a esse comportamento de indagação recursiva, o bot ELIZA aparenta ter o comportamento de um profissional do ramo da psicologia.

O bot ELIZA é capaz gerar respostas a partir de regras simples e pré-determinadas, conforme visto no exemplo anterior. No entanto, este programa apresenta uma séria de limitações.

- Não é capaz de processar linguagem natural (*Natural Language Processing - NLP*), ou seja, realizar a análise sintática e morfológica das frases de forma a extrair os elementos e a relação entre eles dentro do contexto da frase. ELIZA é capaz apenas de buscar por palavras-chave e disparar regras de resposta.
- Apesar de ser capaz de gerar respostas, na maioria dos casos essas respostas são vagas e não agregam informação ao diálogo.
- ELIZA não é capaz de aprender novas respostas, dado que se baseia em regras pré-determinadas e num banco de dados estático.
- ELIZA não lida bem com várias intenções na mesma frase. Ou seja, caso o programa encontre mais de uma palavra-chave na sentença, ele lidará apenas com a primeira e ignorará todas as outras, fazendo com que a resposta pareça muito simplista.
- O programa não é capaz de identificar erros de digitação.

Com os avanços tecnológicos dos últimos anos, principalmente nas áreas de aprendizado de máquina, processamento de linguagem natural e processamento e armazenamento de dados, os chatbots evoluíram muito desde o surgimento de ELIZA.

Um dos chatbots mais célebres é A.L.I.C.E (*Artificial Linguistic Internet Computer Entity*) (WALLACE, 1995), que foi inspirado no programa ELIZA. O bot A.L.I.C.E já recebeu três vezes o prêmio Loebner como reconhecimento de suas

conquistas na área de interação homem-máquina. Este programa é construído sobre a linguagem AIML (*Artificial Intelligence Markup Language*), que é inspirado no XML e se trata basicamente de uma forma estruturada de armazenar conhecimento para a construção de diálogos em linguagem natural. Estes estudos foram fundamentais para o aumento das iniciativas de desenvolvimento de chatbots nas mais diversas áreas. Hoje em dia, contamos com chatbots atuando em serviço de atendimento ao consumidor, serviço de diagnóstico, serviços de agendamento, entre outros.

Nesse contexto, em (GUPTA, BORKAR, *et al.*, 2015) os autores desenvolvem um chatbot responsável por facilitar as interações entre os clientes e uma plataforma Ecommerce. Ou seja, ao invés de o usuário realizar diversas buscas e aplicar diferentes filtros para encontrar o item que deseja comprar, os autores propõem facilitar essa tarefa através de uma pesquisa feita por diálogo com um bot. Por uma questão de simplicidade, os autores preferem não utilizar nenhuma ferramenta de NLP, mas apenas um conjunto extenso de padrões textuais para identificar as intenções do usuário. As principais tarefas deste chatbot são:

- Identificar os itens desejados pelo cliente.
- Levantar, através de questionamentos, todas as informações necessárias para completar a pesquisa (preço máximo, cor, tamanho, etc.)
- Transformar estas informações numa consulta estruturada.
- Consultar o banco de dados do Ecommerce.
- Retornar ao cliente, em linguagem natural, o resultado da pesquisa.

Apesar de ser capaz de auxiliar os clientes em suas buscas, por não utilizar NLP, o programa tem uma capacidade de compreensão limitada caso as frases não correspondam com os padrões textuais definidos. Além disso, por não definir uma ontologia do domínio, o programa não é capaz de compreender a relação entre elementos dentro de um contexto, reduzindo assim sua capacidade de entendimento.

O termo ontologia surgiu na filosofia como a área de estudo da natureza do ser, das coisas existentes no mundo. A ontologia é um aspecto da metafísica que procura categorizar o que é essencial e fundamental em determinada entidade. Na computação, ontologia consiste de uma técnica de formalização da representação do conhecimento e é empregado atualmente nas áreas de Inteligência Artificial, Web Semântica e Arquitetura da Informação (FREITAS e VIEIRA, 2008). Ou seja, ontologia nesse contexto é uma técnica formal e estruturada de representar o conhecimento, composta de elementos (classes) e da relação entre estes. Atualmente as ontologias têm sido empregadas junto aos chatbots com o objetivo de melhorar a capacidade de entendimento dos mesmos, dado que com uma ontologia é possível descrever, de uma forma estruturada, como os elementos de um domínio em específico se relacionam entre si.

Em (MILWARD e BEVERIDGE, 2003) os autores tecem uma rica discussão acerca das vantagens de se empregar uma ontologia do domínio num sistema de diálogo,

principalmente devido ao fato de que nem sempre as sentenças correspondem aos padrões textuais armazenados no programa. Através de uma ontologia é possível extrair relações de hiperônimo e hipônimo da frase e, assim, direcionar as questões ou então inferir determinada resposta. Para ilustrar, consideremos o exemplo abaixo que utiliza uma ontologia do domínio da patologia (S se refere à pergunta do sistema e U se refere à resposta do usuário).

S: Você tem histórico familiar de doenças crônicas?

(Resposta esperada: câncer de pulmão, leucemia, sarcoma...)

U: Sim, câncer.

S: Que tipo de câncer?

U: Câncer de pulmão.

Nesse exemplo, o sistema esperava uma resposta específica, mas o usuário forneceu uma resposta genérica. Dado que câncer é um hiperônimo de câncer de pulmão e essa relação pode ser descrita na ontologia utilizada, o sistema formulou uma questão de forma a estreitar as possibilidades e caracterizar o tipo de câncer.

S: Você já teve alguma doença crônica?

(Resposta esperada: câncer, hipertensão, diabetes...)

U: Sim, Leucemia.

Nesse exemplo, o sistema esperava uma resposta mais genérica para a questão, porém o usuário forneceu uma resposta específica. Dado que Leucemia é um hipônimo de câncer, ou seja, é um tipo específico de câncer, o sistema é capaz de inferir a resposta esperada, sem a necessidade de realizar outra pergunta.

Outra vantagem de se utilizar ontologias, segundo os autores, é a extração de relações parte-todo. Consideremos o seguinte exemplo.

S: Onde está doendo?

(Resposta esperada: cotovelo, punho, ombro...)

U: No braço.

S: Em que parte do braço?

U: No cotovelo.

Nesse exemplo, o sistema espera uma parte específica do corpo humano como resposta, no entanto o usuário forneceu como resposta uma parte genérica. Como a relação entre essas partes está descrita na ontologia (o cotovelo é parte do braço), o sistema é capaz de formular uma questão de forma a direcionar o usuário para fornecer a resposta esperada.

Em (VEGESNA, JAIN e PORWAL, 2018), os autores desenvolvem um Chatbot para auxiliar os usuários nas compras em um site Ecommerce, porém, diferentemente do que foi visto anteriormente em (GUPTA, BORKAR, *et al.*, 2015), os autores definem uma ontologia do domínio para potencializar as capacidades de compreensão e resposta do programa. Outra diferença importante é que neste trabalho os autores utilizam NLP para processar as frases de entrada e algoritmos de *Machine Learning* para treinar o

modelo e ser capaz de identificar frases que não estão na base de padrões textuais. Neste trabalho, os autores utilizam o site Ebay como fonte de dados, definem a ontologia utilizando PROTEGÉ e utilizam as ferramentas de NLP do framework Dialogflow. Uma vez construído o chatbot, este é capaz de identificar as entidades das frases, buscar na ontologia a relação entre essas entidades e então realizar a busca no site Ecommerce conforme as instruções do usuário, ou então, realizar questões específicas caso necessite de informações adicionais. No entanto, as respostas fornecidas pelo chatbot são bastante direcionadas e diretas e não se parecem muito com um diálogo de fato.

# METODOLOGIA

---

O projeto de um chatbot, de modo geral, requer que o projetista defina a estratégias do diálogo, as mensagens de erro, os contextos possíveis da conversa, o data set de treino, entre outros. Essas definições costumam seguir os princípios de *user-centered design*, uma metodologia de projeto que se apoia no tripé: foco no comportamento do usuário, realização constante de medidas empíricas e desenvolvimento iterativo (JOHN D. GOULD, 1985).

Nesta seção serão discutidas as características do projeto no que diz respeito às ferramentas utilizadas no desenvolvimento do programa, à estrutura do diálogo, à construção da base de dados e à modelagem do seu funcionamento.

## 1. Escolha do Framework de desenvolvimento

O desenvolvimento de um *dialog agent* como um chatbot é uma tarefa complexa, principalmente se o desenvolvedor não possuir as ferramentas adequadas. Se fosse construir do zero um tal sistema, o desenvolvedor precisaria definir desde a estrutura de diálogo que o programa aceita, até desenvolver os algoritmos de *parsing* e *classification* de texto, que são a chave das técnicas de NPL. Nesse contexto, os frameworks de desenvolvimento de chatbots permitem aos projetistas e desenvolvedores acelerarem o trabalho, pois possuem uma estrutura propícia para o desenvolvimento de tais sistemas e contam com uma série de bibliotecas e funções integradas à framework, simplificando substancialmente o trabalho.

Existem atualmente diversos frameworks disponíveis. A escolha do framework mais adequado depende principalmente das especificações do sistema a ser construído, da língua de escolha e do orçamento disponível. Nesse quesito, apesar de muitos frameworks apresentarem exclusivamente versões pagas, algumas plataformas disponibilizam licenças gratuitas que são bastante completas e funcionais.

Para escolha do framework a ser utilizado no projeto, foi feito um estudo comparativo das opções disponíveis no mercado sob três principais aspectos: capacidade de detecção, suporte de *machine learning* em português (pt-br) e preço. Para tal, foi analisado um benchmark realizado pela consultoria Intento, que faz uma interessante análise estatística da performance dos principais frameworks disponíveis: IBM Watson, Dialogflow (antigo API.ai), Microsoft LUIS, Amazon Lex, Wit.ai, Recast.ai, SNIPS (SAVENKOV, 2017).

Para a análise da performance de cada *provider*, os consultores utilizaram um data set<sup>2</sup> contendo cerca 15 600 frases em linguagem natural que mapeavam sete principais intenções, cerca de 2 mil frases para cada intenção, sendo estas:

---

<sup>2</sup> <https://github.com/snipsco/nlu-benchmark>

- Buscar um programa de TV.
- Solicitar a previsão do tempo.
- Reservar um restaurante.
- Tocar uma música.
- Adicionar uma música a uma playlist.
- Avaliar um livro.
- Buscar uma sessão de cinema.

Segundo os autores, o data set foi construído através de métodos de *crowdsourcing*. Todos os frameworks são treinados com este data set para identificar as intenções do usuário a partir de uma frase em linguagem natural. No estudo, a precisão na identificação de intenções de cada framework é avaliada segundo as estatísticas de *precision*, *recall* e *F1-score*. A Tabela 1 mostra a comparação da capacidade de identificar intenções a partir de frases para cada *provider*. O valor apresentado é a média normalizada do *F1-score* calculado sob a identificação das sete intenções apresentadas anteriormente.

Tabela 1 – Matriz de comparação dos frameworks disponíveis.

Framework	Detecção de Intenções	Suporte pt-br	Preço / mil requests
IBM Watson	0.997	Sim	R\$ 10.00
Dialogflow	0.996	Sim	Grátis
Microsoft LUIS	0.992	Sim	R\$ 4.98
Amazon Lex	0.965	Não	R\$ 3.00
Recast.ai	0.970	Não	Grátis
Wit.ai	0.974	Sim	Grátis
SNIPS	0.975	Não	N/D

Pode-se perceber que o IBM Watson, Dialogflow e Microsoft LUIS se destacam na detecção de intenções, medida que será de suma importância na construção do sistema, uma vez que permite treinar o chatbot de forma satisfatória com um set de treino significativamente menor. Como não temos um data set de treino pronto para o projeto, ser capaz de construir um chatbot com um comportamento coerente treinando-o com a menor quantidade de informações possível agiliza o desenvolvimento do trabalho. Sendo assim, escolheu-se os três *providers* citados anteriormente como fortes candidatos ao projeto.

Além disso, dado o contexto de trabalho, é imprescindível que o framework tenha total suporte à língua portuguesa, incluindo detecção de intenções, identificação de entidades e *machine learning* nessa língua. Alguns frameworks utilizam um sistema de tradução para uma língua suportada pelo sistema, geralmente o inglês, na qual ocorre toda a dinâmica de processamento de linguagem natural, a resposta é então traduzida de volta para a língua de início e enviada ao usuário. No entanto, nessa transcrição do texto entre uma língua e outra se perde muito da capacidade de interpretação do chatbot, uma vez que a tradução não é perfeita. Esse tipo de tradução não será considerado como um

suporte à língua desejada. Os três candidatos levantados anteriormente possuem um apoio bastante completo à língua portuguesa.

Por fim, é analisado o custo de utilização de cada plataforma. A Tabela 1 mostra uma estimativa do custo a cada mil requisições de cada framework. Como boa parte dos *providers* oferece uma precificação em planos com cotas limites, foi assumido como especificação técnica do sistema pelo menos 10 mil requisições mensais, garantindo assim a escalabilidade do projeto. O IBM Watson, apesar de ser aquele com o maior poder de detecção, é também o mais caro e, portanto, mais adequado a aplicações corporativas. Das três opções, o Dialogflow é a única opção gratuita e possui ainda uma boa capacidade de detecção. Esse framework é distribuído pelo Google e já vem com um SDK propício para desenvolvimento desse tipo de sistema e foi, portanto, o framework escolhido para a realização deste trabalho.

## 2. Construção da Base de Dados

No desenvolvimento de qualquer agente conversacional, é imprescindível que as respostas às questões do usuário estejam ao alcance do sistema, caso contrário, por melhor que o sistema identifique as intenções do usuário, o mesmo não será capaz de responder adequadamente. No contexto deste projeto, o sistema deve ser capaz de responder às questões dos alunos quanto à descrição, carga horária e disponibilidade das disciplinas da instituição. Sendo assim, é necessário que o agente tenha acesso à base de dados que armazena essas informações.

O acesso direto à base de dados da Universidade de São Paulo implicaria em algumas dificuldades quanto à governança e segurança da informação. Para contornar esse problema e focar os esforços no desenvolvimento do agente conversacional em si, optou-se por construir uma base de dados própria a partir das informações disponibilizadas no Jupiterweb. Afinal, estas são as informações comumente acessadas pelos estudantes e evita-se, assim, a exposição pública de informações sensíveis. Entende-se que, uma vez construído o chatbot e validado o interesse dos alunos, almeja-se em um segundo momento do projeto realizar a efetiva integração da ferramenta com o Jupiterweb, podendo assim potencializar as funcionalidades do robô e oficializar a distribuição da solução aos estudantes.

A base de dados construída se apoia numa arquitetura não-relacional. Em termos práticos, essa arquitetura não armazena os dados sob forma de tabelas, mas sim sob a forma de documentos que possuem uma estrutura padronizada, como o formato JSON. Essa arquitetura tem se popularizado dentre os desenvolvedores nos últimos anos, principalmente por sua escalabilidade e flexibilidade no armazenamento de dados (STORL, HAUF, *et al.*, 2015).

No contexto de aplicação desse projeto, essa arquitetura apresenta algumas vantagens, conforme discutido nos itens abaixo.

- Não é necessário definir a estrutura de armazenamento dos dados, ou *schema*, garantindo assim maior flexibilidade à utilização da base.



- Se comparado ao modelo relacional, essa arquitetura não é tão robusta do ponto de vista de integridade dos dados. No entanto, no contexto de aplicação a base será estática, ou seja, não haverá inserções ou exclusões de elementos no decorrer da utilização do programa. Conjuntamente com o fato de ser uma base de dados relativamente pequena, com menos de 100 documentos, pode-se garantir a fiabilidade dos dados através da inspeção dos elementos.
- Essa arquitetura é compatível com orientação a objeto, o que facilita a interação com a API de consulta desenvolvida, por não necessitar o emprego de um ORM para gerenciar o modelo de dados.

A flexibilidade e simplicidade dessa arquitetura são aliados valiosos quando a finalidade é, por exemplo, a prototipagem do sistema. No entanto, entende-se que, conforme o chatbot escala, diversifica suas funcionalidades e aumenta seu escopo de conhecimento, teremos de migrar para uma arquitetura relacional de armazenamento de dados, garantindo assim maior consistência e robustez no modelo de dados.

Por fim, a base de dados foi preenchida com 61 disciplinas da diagonal de formatura do curso de Engenharia Mecatrônica da USP. Foram armazenadas as principais informações presentes no Jupiterweb, tais quais o objetivo e programa da disciplina, horários de oferecimento, método de avaliação, bibliografia, corpo docente responsável, quantidade de créditos, requisitos de disciplinas e carga horária semestral.

### 3. Definição da Arquitetura

Uma vez definido o framework de desenvolvimento e o modelo de armazenamento dos dados, o próximo passo é definir a arquitetura do sistema, ou seja, qual o fluxo da informação dentro do programa e por quais elementos ele passa. Apesar de existirem chatbots muito diferentes uns dos outros, a maioria segue uma estrutura relativamente parecida, que envolve alguns elementos básicos, como uma caixa de diálogo constituindo a interface entre o usuário e o programa, um motor de processamento de linguagem natural, responsável por identificar as intenções dos usuários e eventuais entidades dentro da frase, uma função de acesso a uma base de dados e por fim um motor de geração de resposta e envio à caixa de diálogo. O Jupiterweb chatbot segue uma estrutura bastante similar, podendo ser esquematizada pelo diagrama abaixo.

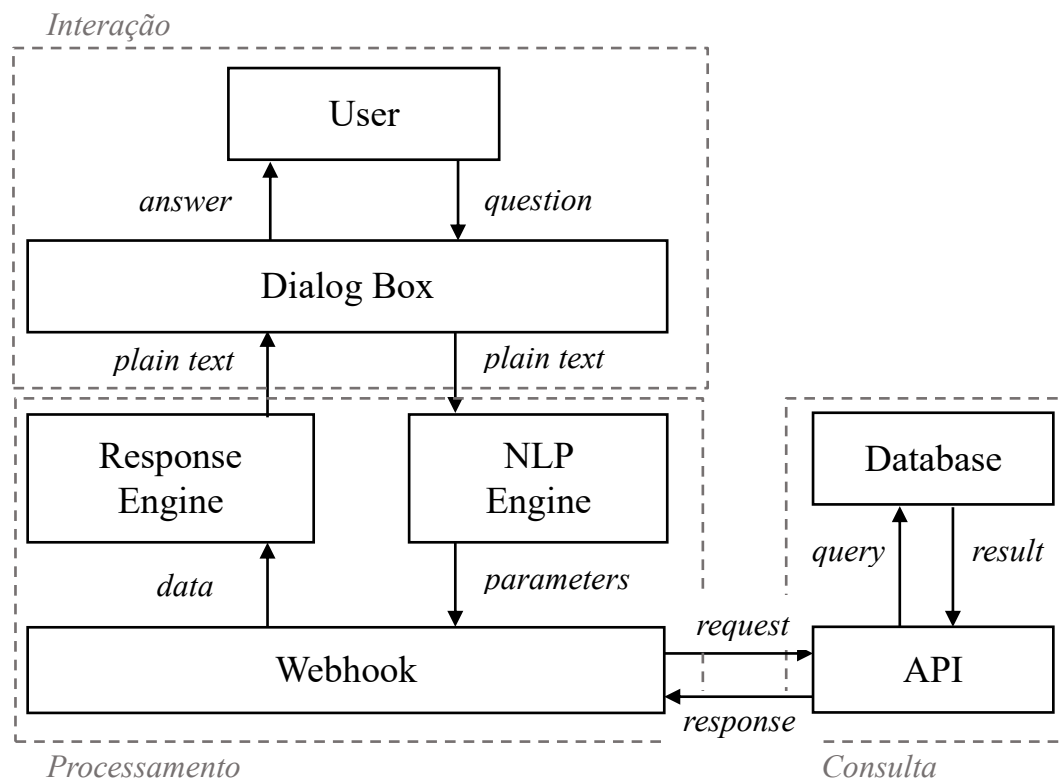


Figura 1 - Arquitetura base do chatbot

O programa possui então três módulos principais: o módulo de Interação, Processamento e Consulta.

- **Módulo de Interação** é composto pela caixa de diálogo (*Dialog Box*) e é responsável por realizar a interface entre o usuário e o sistema. O framework utilizado já possui uma caixa de diálogo construída e fornece uma integração direta com uma página html.
- **Módulo de Processamento** é o motor de *natural language processing* (NLP), responsável pelo processamento das frases, pela identificação da necessidade do usuário e pela detecção das entidades dentro da sentença. Também é responsável por criar as respostas a partir dos dados extraídos e das consultas realizadas. O framework já provê um motor de NLP com suporte para *machine learning*, assim, a performance do motor é função da quantidade e da qualidade do data set de utilizado para treinar o agente. Quanto mais exaustivo e diversificado forem esses dados, melhor será a capacidade de interpretação do agente.
- **Módulo de Consulta** é responsável por encontrar as respostas para as perguntas dos usuários. Uma vez identificadas as necessidades e extraídas as entidades da frase, o programa envia essas informações para uma API que por sua vez irá realizar a devida consulta no banco de dados e montar uma resposta. Essa funcionalidade não vem nativa

do framework, então foi inteiramente desenvolvida utilizando a linguagem Node.js. Para facilitar a comunicação com o sistema, a API foi hospedada no Heroku, que é uma *cloud platform as a service* (PaaS) e possui uma opção gratuita que atende aos requisitos mínimos de funcionamento do sistema.

Para ilustrar o funcionamento do sistema dentro do contexto de estudo, vamos considerar o exemplo representado pela Fig. 2, no qual o usuário questiona o chatbot acerca do horário de uma disciplina específica, no caso Cálculo I. O usuário insere na caixa de diálogo a frase “Segunda tem aula de Cálculo I?”. Essa frase é processada pelo motor do chatbot que identifica dois principais elementos: intenções e entidades. Dada a estrutura da frase e do fato do sistema ter sido previamente treinado para identificar frases similares, o mesmo é capaz de identificar que a intenção do usuário é obter os horários de uma disciplina – representado pelo símbolo \$.

Da mesma forma, o sistema é capaz de identificar duas entidades: dia da semana e disciplina – representadas pelo símbolo @. Uma entidade representa um tipo de informação e cada entidade possui um parâmetro associado, ou seja, o valor que a mesma representa naquele contexto. Nesse caso, a entidade @dia-da-semana possui o parâmetro segunda-feira, enquanto a entidade @disciplina possui o parâmetro Cálculo I. Percebe-se que o sistema é capaz de identificar sinônimos, uma vez que o usuário inseriu a palavra “segunda” e o programa foi capaz de relacioná-la ao parâmetro “segunda-feira”. Assim é possível cobrir diferentes formas de explicitar a mesma ideia.

Após a identificação destes elementos, o chatbot constrói uma requisição com os parâmetros identificados, a intenção detectada e realiza um POST dessa requisição na rota principal da API, via webhook disponibilizado pelo framework. Esta, por sua vez, analisa a intenção – nesse caso obter os horários de uma disciplina - e os parâmetros recebidos – segunda-feira e Cálculo I. Com esses valores, a API monta uma nova consulta no banco de dados buscando saber os horários da disciplina Cálculo I. Nesse caso, a resposta da consulta informa que Cálculo I é oferecido terça-feira. Com essa informação, a API compara o dia da semana enviado na requisição com o dia da semana que a disciplina é de fato oferecida e atribui o valor “não” para essa comparação, uma vez que são dias distintos de oferecimento.

A API devolve então para o webhook o valor da comparação e o dia da semana em que a disciplina é oferecida. Com essa informação, o motor de geração de resposta é capaz de montar uma resposta adequada a partir de um formato padrão de resposta representado no esquema por “Sim/Não, <@disciplina> é oferecido <@dia-da-semana>!”. Preenchendo sequencialmente o formato de resposta com o valor da comparação entre os dias da semana, o nome da disciplina e o dia em que é oferecida, o chatbot é capaz de montar e enviar a resposta: “Não, Cálculo I é oferecido terça-feira!”. Evidente que caso a pergunta tivesse sido “Terça tem Cálculo I?”, a resposta seria “Sim, Cálculo I é oferecido terça-feira!”, seguindo então o mesmo formato.

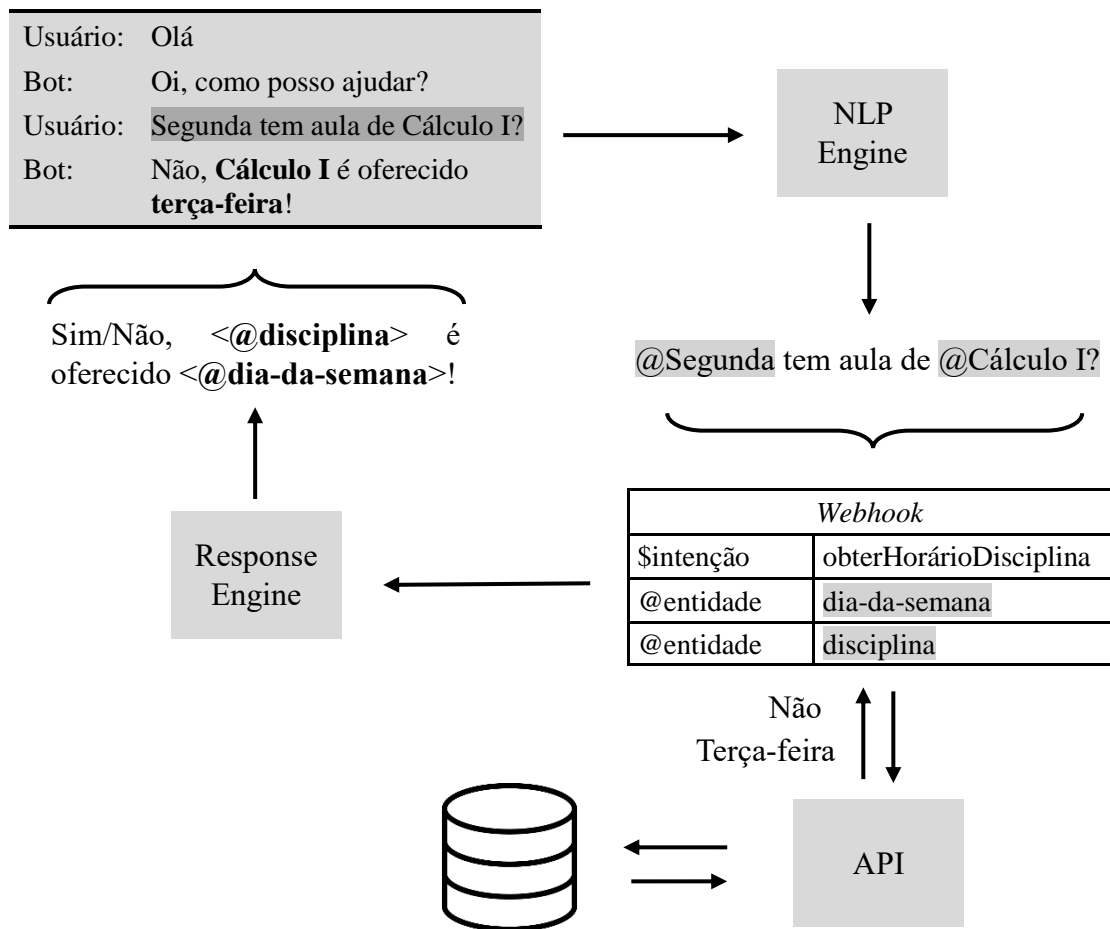


Figura 2 - Fluxo de processamento e resposta em um cenário ilustrativo.

#### 4. Especificação do Sistema

O programa possui duas funcionalidades principais: obter informações de uma determinada disciplina e oferecer uma sugestão de disciplinas de acordo com os interesses do usuário. Para ilustrar essas duas funções, consideraremos um exemplo para cada uma delas, conforme apresentado abaixo.

Para exemplificar a capacidade do chatbot de encontrar informações de uma determinada disciplina, vamos considerar o caso no qual um usuário que deseja saber os professores responsáveis e a descrição da disciplina Projeto de Máquinas, representado pela Fig. 3.

1	Usuário:	Olá
2	Bot:	Oi, como posso ajudar?
3	Usuário:	Quais os professores responsáveis pela disciplina Projeto de Máquinas?
4	Bot:	A disciplina Projeto de Máquinas possui 2 professores responsáveis: Gilberto Francisco Martha de Souza e Julio Cezar Adamowski.
5	Usuário:	Qual o tema dessa matéria?
6	Bot:	A disciplina possui a seguinte descrição: Aprendizado de metodologias empregadas no projeto de máquinas automáticas, abordando técnicas de fabricação e montagem e análises dinâmicas de máquinas. Integração de conhecimentos adquiridos ao longo do curso, visando o projeto e construção de uma máquina a controle numérico (CNC).
7	Usuário:	Muito obrigado.
9	Bot:	Disponha!

Figura 3 - Exemplo de busca de informações da disciplina.

É importante ressaltar que na sentença de número cinco, o usuário não informou novamente o nome da disciplina, ainda assim o programa foi capaz de associar a pergunta à disciplina que estava sendo discutida anteriormente e buscar corretamente a descrição da matéria. Isso é feito através da definição de um contexto. Ou seja, toda vez que uma disciplina é identificada, ela é adicionada ao contexto da conversa, que é passado ao longo das interações. Caso o usuário omita o nome da disciplina de interesse, pois está implícito que ainda está se referindo à disciplina mencionada anteriormente, o programa utiliza como padrão aquela que está armazenada no contexto, garantindo assim fluidez na conversa e se aproximando mais de um diálogo entre seres humanos.

Para ilustrar a capacidade do programa de sugerir algumas disciplinas de acordo com os interesses do usuário, vamos considerar o exemplo no qual o aluno gostaria de uma sugestão de matéria que envolva sistemas hidráulicos, representado na Fig. 4.

1	Usuário:	Olá
2	Bot:	Oi, como posso ajudar?
3	Usuário:	Poderia me ajudar a escolher uma matéria?
4	Bot:	Claro, você tem algum tema de interesse?
5	Usuário:	Sistemas hidráulicos.
6	Bot:	Tem alguma preferência de professor?
7	Usuário:	Indiferente.
8	Bot:	Qual o melhor horário para você?
9	Usuário:	Idealmente segunda-feira.
10	Bot:	Encontrei uma disciplina que se adequa aos seus interesses: PMR3407 - Sistemas Fluidomecânicos. Você gostaria de saber mais sobre a disciplina?

Figura 4 - Exemplo de sugestão de disciplina.

Nesse caso o programa usou três critérios de seleção para a escolha da disciplina: um tema de interesse, uma preferência de professor e uma restrição de horário. Vale ressaltar que nesse caso, o usuário poderia ainda continuar a conversa solicitando mais informações sobre a disciplina encontrada. Isso levaria a uma dinâmica parecida com aquela exposta no exemplo anterior, permitindo ao usuário solicitar a descrição da disciplina, os professores responsáveis, a carga-horária entre outros. Ao sugerir uma disciplina, o programa já armazena essa disciplina no contexto da conversa, assim não seria necessário informar explicitamente o nome da matéria. Uma possibilidade de continuação do diálogo está exposta na Fig. 5.

8	Bot:	Qual o melhor horário para você?
9	Usuário:	Idealmente segunda-feira.
10	Bot:	Encontrei uma disciplina que se adequa aos seus interesses: PMR3407 - Sistemas Fluidomecânicos. Você gostaria de saber mais sobre a disciplina?
11	Usuário:	Qual a descrição da matéria?
12	Bot:	A disciplina possui a seguinte descrição: Automação fluidomecânica baseada em sistemas e circuitos hidráulicos e pneumáticos e introdução de conceitos de máquinas de fluxo (turbinas e bombas).
13	Usuário:	Quais os horários da disciplina?
14	Bot:	A turma 50 é oferecida segunda-feira das 13h10 às 14h50 e terça-feira das 13h10 às 14h50.

Figura 5 - Continuação da conversa após sugestão de disciplina.

Esses dois exemplos possuem uma diferença significativa quanto à dinâmica de resposta. No segundo exemplo, uma vez que o programa precisa de algumas informações adicionais para realizar a sugestão de disciplina, é ele o agente responsável por ditar o ritmo do diálogo. O agente faz perguntas diretas ao usuário, como por exemplo “qual o melhor horário para você?”, e cabe ao usuário interagir de acordo com o que foi solicitado pelo programa. Esse tipo de interação é chamado de *system-initiative*, e simplifica bastante a dinâmica da conversa, uma vez que o sistema sabe exatamente o que o usuário está respondendo e que tipo de informação ele está fornecendo. Fazendo perguntas diretas, o sistema limita o escopo de possíveis respostas e facilita a compreensão das mesmas.

Já no primeiro exemplo, esses papéis se invertem, e cabe ao usuário ditar o ritmo da conversa e solicitar objetivamente as informações que deseja. Esse tipo de interação é chamado de *user-initiative*. Nesse tipo de interação, o usuário pode a qualquer momento mudar o domínio da conversa ou retomar um assunto tratado anteriormente, o que dificulta a compreensão do programa, pois as sentenças do usuário não têm mais um escopo limitado e podem abranger uma série de assuntos distintos. Cabe ao programa identificar as intenções dos usuários e adaptar a resposta de acordo com o contexto atual da conversa. É característica natural de uma conversa entre dois indivíduos que a iniciativa alterne frequentemente entre os participantes. A mesma característica foi conferida ao chatbot construído. Ou seja, ora a iniciativa é do chatbot, ora do usuário, conferindo assim a característica *mixed-initiative* ao sistema.

## 5. Modelo de Estados

No projeto de um chatbot *frame-based*, é fundamental que o diálogo seja previamente modelado. A vantagem desse tipo de sistema é que ele possui um número finito de possibilidades de interações, portanto, pode-se utilizar um modelo de estados para representar a dinâmica do sistema. Conforme descrito anteriormente, o programa possui duas funcionalidades principais que funcionam independente uma da outra, assim, podem ser representadas separadamente. O modelo da Fig. 6 representa os estados da função que busca informações de uma dada disciplina.

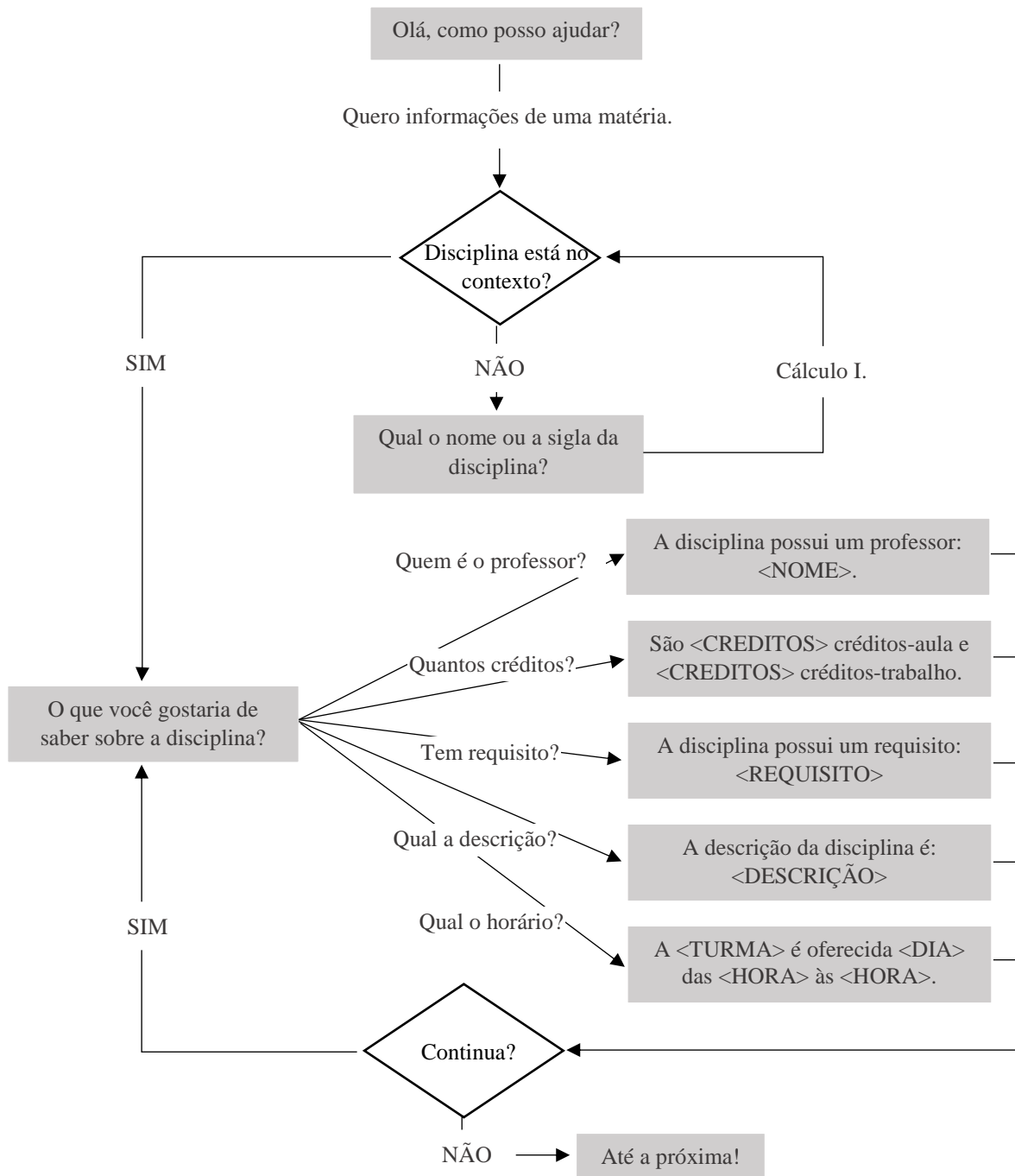


Figura 6 - Modelo de estados da busca de informações de disciplinas.

No modelo acima, os estados do programa são representados pelas frases dentro dos blocos, e as transições de estados são disparadas pelas sentenças do usuário. O estado do sistema pode ser compreendido como o comportamento do agente naquele instante. Percebe-se que nesta representação temos diversos estados paralelos cuja ativação de cada estado depende da demanda do usuário. Isso exprime bem a característica *user-initiative* dessa funcionalidade, conforme explicado anteriormente. Ou seja, o usuário decide o rumo da conversa tem liberdade para decidir o que quiser saber, cabendo ao sistema se adequar a essa dinâmica. No entanto, mesmo o usuário tendo liberdade para guiar a conversa, o escopo de possibilidades que o sistema é capaz de compreender continua sendo finito, que é a característica principal dos agentes *frame-based*.

Alguns estados dependem, não somente da demanda do usuário, mas também das informações conhecidas naquele instante. No nosso caso, o usuário faz a seguinte solicitação: “Quero informações de uma matéria”. Nesse caso, o estado do sistema vai depender se ele possui conhecimento de qual matéria o usuário se refere ou não. Caso ele não tenha esse conhecimento, irá perguntar explicitamente a qual disciplina o usuário se refere.

Existem duas formas de obter a disciplina de interesse, a primeira é através da identificação de entidades na frase. Caso a sentença seja “quero informações sobre Cálculo I”, o programa é capaz de identificar que a disciplina a que o usuário se refere é Cálculo I, sem ter de demandar diretamente ao usuário. A segunda forma é através do contexto. Caso a conversa já tenha um histórico razoável, existem grandes chances de o usuário já ter mencionado a disciplina em outro ponto do diálogo e esta foi armazenada dentro do contexto da conversa. Caso esse contexto esteja definido, o programa infere que o usuário está se referindo a esta mesma disciplina e usa esse valor para dar prosseguimento à conversa.

Alguns estados representam o retorno das informações solicitadas. Essas informações são representadas no modelo pelo sinal <RETORNO>. A característica desses estados é que eles são responsáveis por fazer a interface entre o sistema e a base de dados. São exclusivamente nesses estados que são criados as requisições que buscam as informações solicitadas e são estes os estados responsáveis por gerar as respostas a partir das informações colhidas da base de dados.

A segunda funcionalidade do sistema, que permite o agente sugerir uma disciplina de acordo com os interesses do usuário, também pode ser representada por um modelo de estados, como pode ser visualizado na Fig. 7. No entanto, este modelo possui um formato muito mais linear, característico dos agentes *system-initiative*. O objetivo do sistema é solicitar todas as informações que ele precisa para completar a tarefa que lhe foi solicitada. Isso é feito de forma sequencial, ou seja, cada informação que o sistema precisa é representada por uma pergunta direta ao usuário, que deve responder diretamente o que foi solicitado. Essa dinâmica de busca sequencial dos parâmetros necessários para completar a tarefa é chamada de *slot filling*.

Nesse caso, o sistema precisa de três informações para realizar a tarefa: um tema de interesse, uma preferência de professor e uma restrição de horário. Percebe-se no modelo que cada informação é coletada sequencialmente no diálogo e ao obter todas elas, o sistema realiza a consulta com as restrições informadas e sugere então uma disciplina.



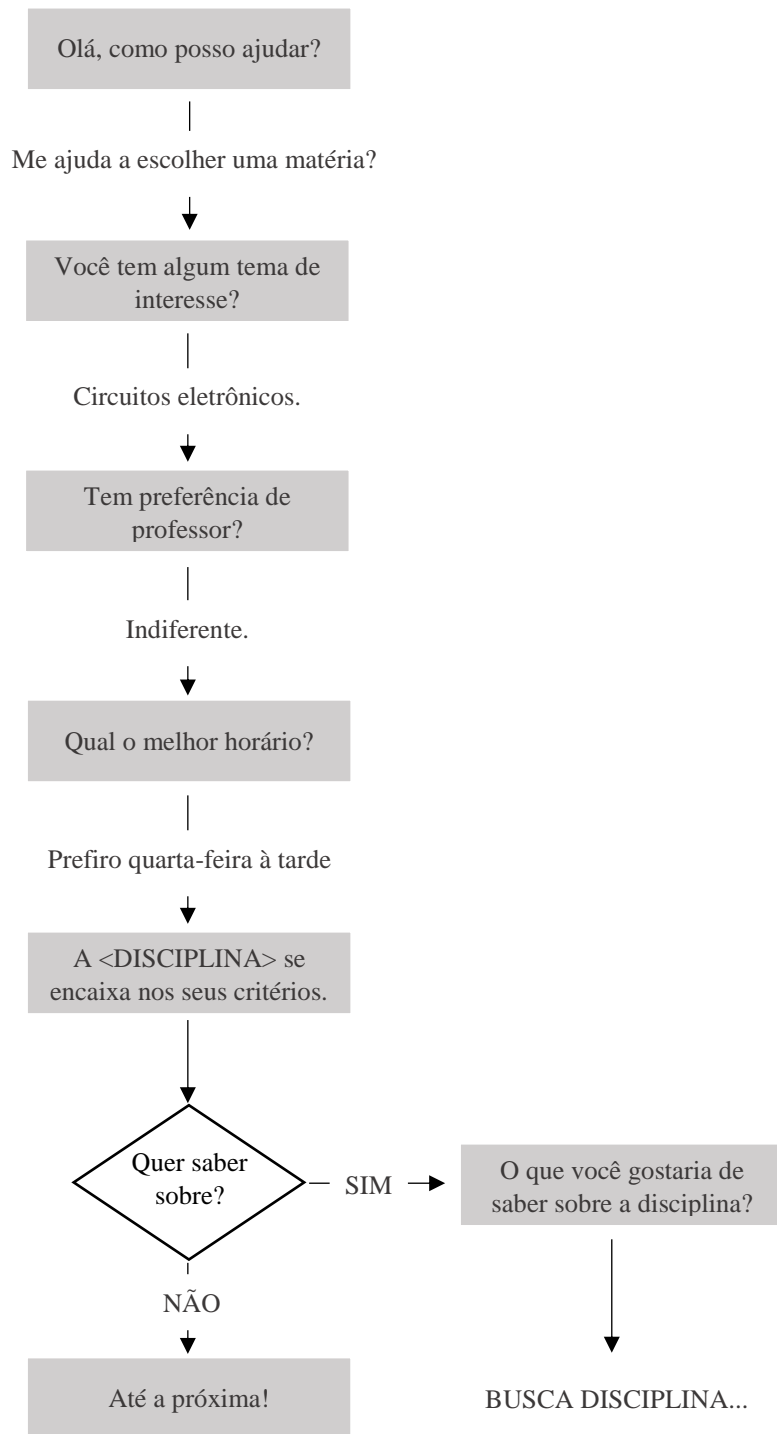


Figura 7 - Modelo de estados da função de sugestão de disciplinas.

Percebe-se que nessa representação, após a sugestão de disciplina, o usuário pode ainda solicitar mais informações acerca da mesma. Nesse momento, a conversa volta para a dinâmica descrita anteriormente, no qual o usuário escolhe o que deseja saber sobre a disciplina e o agente responde de acordo. Assim, pode-se verificar que existe uma relação entre as funcionalidades do sistema. Estas foram anteriormente representadas separadamente, no entanto, pode-se criar uma representação macroscópica do sistema que

engloba as duas dinâmicas *system-initiative* e *user-initiative*, compondo assim o sistema inteiro. O modelo é representado na Fig. 8.

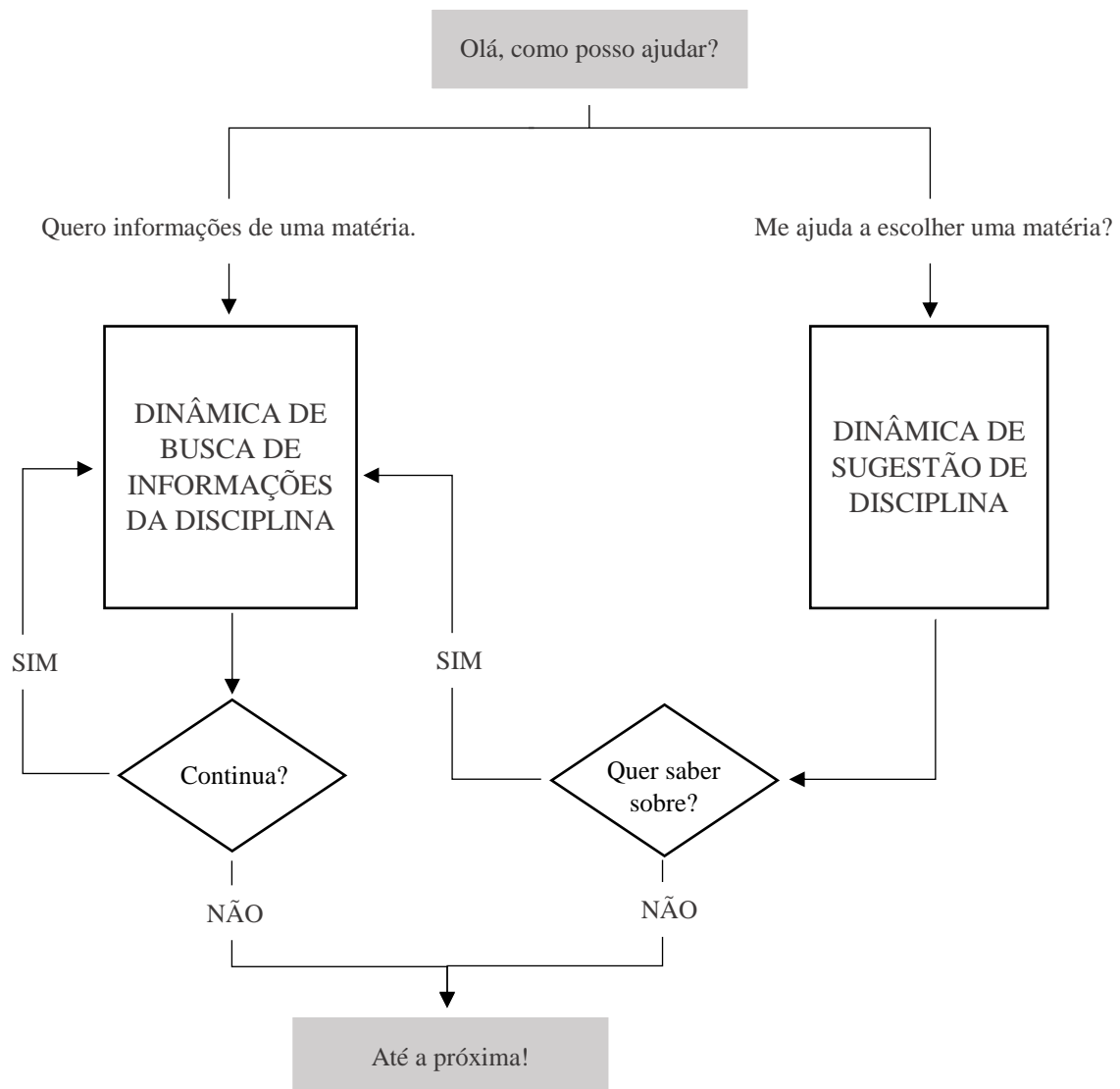


Figura 8 - Modelo de estados completo do sistema.

A modelagem do diálogo é parte fundamental do projeto de um chatbot *frame-based*. Isso permite ao projetista simular as interações e garantir que não existe nenhum comportamento aberrante ou loop na conversa. Nesse caso o sistema é relativamente pequeno, então essa análise parece intuitiva, no entanto, no caso de um sistema muito maior, esse tipo de modelagem é fundamental. Além disso, esta representação auxilia o desenvolvimento do programa base do chatbot, uma vez que existe isomorfismo entre os estados do modelo e as classes do programa, bem como entre os gatilhos de transição de estado e o processo de treinamento do robô para identificação de texto.

# RESULTADOS

---

A execução do projeto pode ser resumida nas seguintes atividades: construção da base de dados, desenvolvimento da API de consulta, design e implementação do diálogo e treino do programa. Esta seção visa analisar a qualidade do chatbot criado sob três principais aspectos: estrutura do diálogo, performance na interpretação de texto e usabilidade do sistema. Para cada aspecto serão propostas metodologias de avaliação, em conjunto com os devidos resultados e discussões.

## 1. Análise do Diálogo

Abaixo é apresentado uma interação completa entre um usuário e o chatbot. Essa interação percorre boa parte dos estados apresentados no modelo de estados descrito anteriormente, fornecendo assim uma ideia da capacidade de resposta do robô.

1	Usuário:	Oi
2	Bot:	Olá, sou o JupiterBot! Posso te dar informações sobre as disciplinas ou te ajudar a escolher uma matéria.
3	Usuário:	Preciso escolher uma disciplina, pode me ajudar?
4	Bot:	Você tem algum tema de interesse?
5	Usuário:	Turbinas Hidráulicas.
6	Bot:	Você tem alguma preferência de professor?
7	Usuário:	Não tenho.
8	Bot:	Qual o melhor horário para você?
9	Usuário:	Segunda e terça das 10h às 18h.
10	Bot:	Encontrei 1 disciplinas com as suas preferências: PMR3407.
11	Usuário:	Sobre o que é PMR3407?
12	Bot:	A disciplina PMR3407 - Sistemas Fluidomecânicos para Mecatrônica possui a seguinte descrição: Automação fluidomecânica baseada em sistemas e circuitos hidráulicos e pneumáticos e introdução de conceitos de máquinas de fluxo (turbinas e bombas).
13	Usuário:	Quem dá a matéria?
14	Bot:	A disciplina possui os seguintes professores: Emilio Carlos Nelli Silva.
15	Usuário:	Qndo é oferecida?
16	Bot:	A turma 01 tem horário seg às 15:00 - 16:40 . A turma 02 tem horário seg às 15:00 - 16:40 . A turma 50 tem horário seg às 13:10 - 14:50 ter às 13:10 - 14:50 .
17	Usuário:	Quantos créditos?

18	Bot:	São 5 créditos aula e 0 créditos trabalho
19	Usuário:	Preciso fazer alguma matéria antes dessa?
20	Bot:	A disciplina não possui requisitos!
21	Usuário:	Me dê a descrição de projemaq
22	Bot:	A disciplina PMR3411 - Projeto de Máquinas possui a seguinte descrição: Aprendizado de metodologias empregadas no projeto de máquinas automáticas, abordando técnicas de fabricação e montagem e análises dinâmicas de máquinas. Integração de conhecimentos adquiridos ao longo do curso, visando o projeto e construção de uma máquina a controle numérico (CNC).
23	Usuário:	Qual a carga horária total?
24	Bot:	A disciplina possui 60 horas no semestre.
25	Usuário:	Qual a velocidade da luz?
26	Bot:	Desculpe, não entendi. Podemos recomeçar?
27	Usuário:	Não se preocupe, obrigado!
28	Bot:	Disponha! O que achou da nossa conversa? Por favor, dê uma nota de 1 a 5 para a nossa interação.
29	Usuário:	Nota 4.
30	Bot:	Obrigado e até mais!

Figura 9 - Exemplo completo de interação com o robô.

A partir dessa interação, teceremos algumas análises da conversa, com base no número que identifica de cada frase, explicitado entre parênteses.

- Em (2), o sistema **inicia descrevendo brevemente suas capacidades**, limitando assim o escopo da conversa e facilitando a interação. É uma prática comum em chatbots *frame-based*.
- O usuário começa solicitando uma sugestão de disciplina. Em (7) percebe-se que **não é necessário especificar todos os critérios** de seleção. Nesse caso, o usuário não especificou uma preferência de professor e ainda sim o programa conseguiu lidar corretamente com essa solicitação.
- Em (9), a restrição de horário pode ser **tanto em dias quanto em horas, ou ambos**. A definição não precisa necessariamente ser entre dois horários. O usuário poderia, por exemplo, ter escolhido o horário “segunda depois das 15h”.
- O chatbot **fornece todas as disciplinas que se encaixam nos critérios** do usuário. Nesse caso, em (10), foi encontrada apenas uma disciplina.
- Em (10) termina a dinâmica de sugestão de disciplina. Percebe-se que nesta interação, o chatbot era quem solicitava informações ao usuário num

modelo sequencial, conforme exposto no modelo de estados dessa dinâmica. Essa interação identifica a **característica *system-initiative* do chatbot**.

- Em seguida, em (11) o usuário solicita informações da disciplina encontrada. Percebe-se então que **é possível mudar da dinâmica de sugestão de disciplina para a dinâmica de informações das disciplinas**, conforme descrito anteriormente no modelo de estados.
- Como visto em (13), **não é necessário mencionar novamente o nome da disciplina** em questão para solicitar novas informações. Isso porque **o chatbot guarda o contexto da conversa**, ou seja, o chatbot intui que o usuário se refere a mesma disciplina que estava sendo discutida anteriormente.
- Em (15), percebe-se que **o chatbot consegue gerenciar abreviações e contrações das palavras**. Nesse caso, ao invés de utilizar a palavra “quando”, o usuário contraiu para “qndo”. Essa utilização é muito comum entre estudantes universitários e mostra que a capacidade de interpretação do programa está adequada ao público alvo.
- Em (23), o usuário utiliza também uma contração, mas dessa vez é uma contração não da língua portuguesa, mas sim do contexto específico da instituição de ensino. Nesse caso, ao invés de mencionar a disciplina “Projeto de Máquinas”, o usuário utilizou “projemaq”, como é popularmente conhecida entre os alunos. Isso mostra que **o programa é capaz de identificar abreviações específicas do contexto de atuação**.
- De (21) a (24) o usuário altera o contexto da conversa e passa a falar de outra disciplina. Percebe-se que o chatbot é capaz de responder adequadamente e guardar o contexto da nova conversa, mostrando assim **boa capacidade de interação no modelo *user-initiative*** descrito anteriormente.
- Em (25) o usuário faz uma solicitação fora do contexto para o qual o programa foi criado. De fato, como visto em (26), o chatbot não sabe interagir com essa solicitação e sugere recomeçar. Isso mostra que **o programa é capaz de reconhecer interações fora de escopo** e lidar propriamente com elas. O extremo oposto seria caso o programa identificasse esta como uma solicitação válida (falso positivo) e continuasse uma conversa que não faria sentido para o usuário.
- O programa finaliza em (28) solicitando ao usuário uma avaliação da interação. Isso é importante para que a **dinâmica do robô seja refinada de acordo com o feedback dos usuários** e assim melhorar o nível de satisfação dos mesmos.

Com essa análise, conclui-se que o chatbot tal qual foi construído respeita as especificações de projeto no que diz respeito às funcionalidades definidas previamente.

Ou seja, é capaz buscar informações de uma disciplina de interesse e também é capaz de sugerir disciplinas que se adequam aos critérios de seleção do usuário. Tudo isso através de uma interação homem-máquina baseada no diálogo na língua portuguesa. Além disso, o robô se mostra adequado ao público alvo composto majoritariamente de estudantes universitários, uma vez que é capaz de lidar com contrações e abreviações das palavras.

## 2. Análise de Performance

Nesta seção será realizada a análise da capacidade de identificar corretamente as intenções do usuário através de um teste de classificação. Para tal, será primeiramente descrito a metodologia utilizada no teste bem como alguns conceitos necessários para o entendimento do mesmo. Em seguida serão apresentados os resultados e uma análise de cada indicador obtido.

### 2.1. Metodologia

A análise de performance consiste basicamente na avaliação da capacidade do robô de buscar corretamente as informações solicitadas de uma disciplina em questão. O programa é capaz de buscar seis informações de cada disciplina, apresentadas na Tabela 2. Para saber qual informação o robô deve buscar, este identifica a intenção do usuário a partir das frases de entrada. Assim, cada tipo de informação está associado a uma intenção distinta, que estão relacionadas aos estados apresentados anteriormente no modelo de estados. A tabela abaixo identifica estas intenções com uma breve descrição e um exemplo de sentença que ativa a realização de cada tarefa.

Tabela 2 – Lista de informações disponibilizadas pelo chatbot.

<b>ID</b>	<b>Intenção</b>	<b>Descrição</b>	<b>Exemplo</b>
1	GetCourseDescription	Busca a descrição ou resumo da disciplina.	“Sobre o que se trata essa matéria?”
2	GetCourseSchedule	Busca os horários de oferecimento da disciplina.	“Quando a disciplina é oferecida?”
3	GetCourseTeacher	Busca os docentes responsáveis pela disciplina.	“Quem são os professores?”
4	GetCourseCredit	Busca a quantidade de créditos da disciplina.	“Qual o total de créditos?”
5	GetCourseWorkload	Busca a carga horária semestral da disciplina.	“Quantas horas ao todo?”
6	GetCourseRequirements	Busca os requisitos da disciplina.	“Tem algum pré-requisito?”

Essa funcionalidade do chatbot pode ser compreendida como um simples classificador multi-classe. Em outras palavras, o robô deve classificar, a partir de uma sentença de entrada, a intenção do usuário, dentre as seis intenções possíveis. Uma vez identificada corretamente a intenção, o programa realiza a consulta na base de dados que retorna a informação solicitada. Dado que essa consulta é determinística, toda a performance dessa funcionalidade se resume à capacidade do programa identificar corretamente a intenção do usuário, que depende essencialmente da sentença utilizada. Existe uma infinidade de variações possíveis de sentença que exprimem uma mesma intenção na língua portuguesa. Então, a performance do robô está fortemente associada à capacidade de identificar corretamente os subconjuntos de sentenças relativos a uma mesma intenção.

Uma abordagem típica na avaliação de classificadores é a análise da matriz de confusão que correlaciona a classificação de cada elemento com seu real valor. Dessa matriz são derivadas métricas que exprimem a capacidade do classificador identificar corretamente alguma classe. Considerando um simples classificador binário, que identifica uma classe entre positivo e negativo, podemos montar a seguinte matriz de confusão.

Tabela 3 – Matriz de confusão de um classificador binário.

	<b>Predicted Positive</b>	<b>Predicted Negative</b>
<b>Actual Positive</b>	True Positive (TP)	False Negative (FN)
<b>Actual Negative</b>	False Positive (FP)	True Negative (TN)

A partir dessa matriz, são calculados cinco métricas que exprimem a performance do classificador (HOSSIN e SULAIMAN, 2015).

Tabela 4 – Métricas de avaliação de classificadores.

<b>Métrica</b>	<b>Fórmula</b>	<b>Descrição</b>
Accuracy (acc)	$\frac{TP + TN}{TP + FP + TN + FN}$	Exprime a proporção de avaliações corretas sobre o total de avaliações.
Error rate (err)	$\frac{FP + FN}{TP + FP + TN + FN}$	Exprime a proporção de avaliações incorretas sobre o total de avaliações.
Precision (p)	$\frac{TP}{TP + FP}$	Exprime a proporção de classes identificadas como positivas que são realmente positivas.
Recall (r)	$\frac{TP}{TP + FN}$	Exprime a proporção de classes identificadas como positivas dentre todas as classes positivas do conjunto.
F1-score (f1)	$\frac{2 * p * r}{p + r}$	Exprime a média harmônica entre <i>precision</i> e <i>recall</i> do classificador.

Além das métricas expostas acima, existe um interesse especial à incidência de falsos positivos, pois estes têm um grande impacto negativo na experiência do usuário ao interagir com o robô. Então, também foi realizado um teste que mediu a quantidade de falsos positivos identificados dentro de um data set de sentenças que estão deliberadamente fora do escopo de resposta do programa.

## 2.2. Resultados

Nesta seção serão descritos em detalhes os testes realizados. Além disso serão apresentados os resultados de performance obtidos em conjunto com uma breve discussão.

### 2.2.1. *Teste de Classificação*

O teste de classificação consiste basicamente na análise do par obtido-esperado na classificação das intenções do usuário. Para tal, foi construída uma base de teste contendo 109 frases, cerca de 19 frases para cada uma das 6 intenções possíveis. Para cada frase desse conjunto foi definida a intenção esperada. Para que não exista relação entre a base de sentenças utilizadas para treinar o robô e a base de testes, que acarretaria numa estimativa de performance com viés, optou-se pela construção de tal base de testes via *crowdsourcing*. Nesse caso, contou-se com a contribuição de 6 colaboradores que sugeriram um conjunto de sentenças e a respectiva intenção esperada, constituindo então a base de testes do experimento. O data set de testes pode ser encontrado no Apêndice A.

Cada uma das sentenças do conjunto foi enviada ao chatbot e a intenção identificada foi mapeada. Comparando tal intenção com a intenção esperada definida previamente, pode-se construir a seguinte matriz de confusão multi-classe, representada sob forma de mapa de calor.



Actual Class	Predicted Class						
	GetCourseCredit	GetCourseInfo	GetCourseRequirements	GetCourseSchedule	GetCourseTeacher	GetCourseWorkload	Rejeicao
GetCourseCredit	10	0	0	0	0	0	4
GetCourseInfo	0	19	0	0	0	0	3
GetCourseRequirements	0	1	16	0	0	0	2
GetCourseSchedule	0	1	0	16	0	1	3
GetCourseTeacher	0	2	0	0	16	0	0
GetCourseWorkload	0	0	0	0	0	10	5
Rejeicao	0	0	0	0	0	0	0

Figura 10 - Matriz de confusão do teste de classificação.

Na matriz acima, temos no eixo das ordenadas a intenção esperada e no eixo das abscissas a intenção detectada pelo chatbot. Assim, pode-se avaliar cada classe da matriz separadamente pela observação da linha e da coluna correspondente. Em outras palavras, para cada intenção do conjunto, podemos analisar a matriz da seguinte forma:

- O encontro da linha e da coluna de uma classe corresponde a uma intenção identificada corretamente pelo chatbot, ou seja, corresponde às classificações **true positives** (TP).
- As células da linha de uma classe, com exceção da célula da diagonal principal, correspondem às classificações **false negatives** (FN)
- As células da coluna de uma classe, com exceção da célula da diagonal principal, correspondem às classificações **false positives** (FP).
- Todas as células que não correspondem à coluna e à linha de uma classe correspondem às classificações **true negatives** (TN).

Na matriz existe uma classe a mais, denominada Rejeição. Essa classe corresponde ao evento no qual o robô não entende a solicitação do usuário. Ou seja, não associa a frase de entrada a nenhuma das intenções possíveis, então dispara uma reação de contingência, geralmente expressa em frases como: “Desculpe, não entendi.”. Como essa não é uma intenção esperada, e sim um resultado claro da performance imperfeita do

robô, esta não possui um valor na diagonal principal e seu efeito é apenas diminuir a precisão da identificação das outras intenções.

### 2.2.2. Accuracy & Error Rate

A partir da matriz de confusão multi-classe e das fórmulas expostas anteriormente, podemos calcular os valores de accuracy e error rate do chatbot. À primeira vista, verifica-se que o robô possui uma boa accuracy e classifica corretamente as intenções na grande maioria dos casos, conforme a Tabela 5.

Tabela 5 – Cálculo de Accuracy e Error Rate para cada classe.

Class	Accuracy	Error Rate	Samples
GetCourseCredit	96.33%	3.67%	14
GetCourseInfo	93.58%	6.42%	22
GetCourseRequirements	97.25%	2.75%	19
GetCourseSchedule	95.41%	4.59%	21
GetCourseTeacher	98.17%	1.83%	18
GetCourseWorkload	94.50%	5.50%	15
<b>Average/Total</b>	<b>95.81%</b>	<b>4.19%</b>	<b>109</b>

Essas duas métricas são relativamente fáceis de calcular e de compreender. No entanto, elas possuem limitações claras e são adequadas para casos em que não há desequilíbrio no conjunto de dados do teste de classificação. Ou seja, a quantidade de classes positivas e negativas é próxima e o tamanho da amostra de cada uma das classes de um classificador multi-classe também é próximo. Essas condições dificilmente são encontradas em problemas reais e esses indicadores acabam por favorecer as classes predominantes dentro do conjunto de dados (STAPOR, 2018).

Percebe-se que o data set em estudo não é simétrico, uma vez que, para cada classe analisada, a quantidade de observações negativas é maior que a quantidade de observações positivas. Considerando, por exemplo, a classe GetCourseCredit, tem-se no data set 14 observações positivas – soma da linha correspondente a essa classe – e 95 observações negativas – soma de todas as linhas com exceção da linha correspondente a essa classe. Existe então um forte desequilíbrio no data set em favor das observações negativas. Com isso, conclui-se que o valor de accuracy atribui um peso muito maior à identificação correta de classes negativas, em detrimento da identificação correta de classes positivas. Uma vez que a capacidade de interação do robô está fortemente associada à sua capacidade de identificar corretamente as classes positivas, ou seja, a correta identificação da intenção do usuário, conclui-se que, no caso de estudo, **accuracy e error rate não são boas estimativas de performance**, devido ao desequilíbrio no data set em favor das classes negativas.

### 2.2.3. Precision, Recall & F1-score

O problema de assimetria de classes é frequente em grande parte das aplicações e pesquisas que envolvem a classificação de elementos. Casos como detecção de fraudes, detecção de terroristas e diagnósticos médicos são casos típicos onde o desequilíbrio do conjunto em favor das classes negativas levou os pesquisadores a buscarem outras métricas mais robustas. Existem diversas métricas que podem ser utilizadas para obter uma análise mais crível da performance de um classificador, dentre elas: G-means, Precision, Recall, F-score, Balanced Accuracy, ROC Curve, Discriminant Power, Likelihood Ratios, entre outros (BEKKAR, DJEMAA e ALITOUICHE, 2013).

No caso de estudo, serão utilizadas três métricas, sendo essas: Precision, Recall e F1-score. Estas são métricas relativamente simples de calcular do ponto de vista de complexidade computacional e também tem um significado direto na aplicação em questão.

Conforme visto anteriormente, a precision se refere à proporção de classes identificadas como positivas que realmente são positivas, enquanto o recall se refere à proporção de classes identificadas como positivas dentro o conjunto de todas as classes positivas do data set. Por fim, F1-score é a média harmônica entre precision e recall.

Com base na matriz de confusão exposta anteriormente, é possível calcular essas três métricas para cada uma das intenções do conjunto, conforme exposto na Tabela 6.

Tabela 6 – Valores de Precision, Recall e F1-Score para cada classe.

Class	Precision	Recall	F1-score	Samples
GetCourseCredit	100.00%	71.43%	83.33%	14
GetCourseInfo	82.61%	86.36%	84.44%	22
GetCourseRequirements	100.00%	84.21%	91.43%	19
GetCourseSchedule	100.00%	76.19%	86.49%	21
GetCourseTeacher	100.00%	88.89%	94.12%	18
GetCourseWorkload	90.91%	66.67%	76.92%	15
<b>Average/Total</b>	<b>95.24%</b>	<b>79.82%</b>	<b>86.47%</b>	<b>109</b>

Percebe-se que o robô tem uma precisão alta, cuja média ponderada pela amostra de cada classe é aproximadamente 95%. Esse valor resulta da baixa proporção de falsos positivos em cada classe analisada. Em outras palavras, quando o robô identifica uma intenção, esta intenção corresponde na maioria dos casos à intenção do usuário.

No entanto, percebe-se que o chatbot possui um valor de recall mais baixo, aproximadamente 80% para a média ponderada das classes. Isso resulta da proporção de casos no qual o programa não conseguiu identificar nenhuma intenção e rejeitou a solicitação, representado no mapa de calor pela classe Rejeição.

Um valor alto de precision e um valor moderado de recall sugerem que robô possui uma característica restritiva, ou seja, só define uma classe quando tem certeza de que se

trata da classe correta. A contrapartida disso é que o sistema acaba por identificar apenas uma pequena parte do conjunto de elementos pertinentes, daí a conotação restritiva do classificador.

Percebe-se que existe uma relação intrínseca entre precision e recall. Assim, poderíamos mudar o comportamento do classificador para que este tivesse um recall elevado, próximo de 100%. No entanto, provavelmente isso afetaria muito o valor de precision, pois teríamos um classificador que identifica praticamente todos os elementos pertinentes do conjunto, mas que identifica incorretamente boa parte dos elementos, representado por uma alta proporção de falsos positivos.

O valor de F1-score serve para exprimir o *trade-off* entre recall e precision. O cálculo via média harmônica tem o intuito de penalizar valores muito extremos entre esses dois parâmetros, uma vez que o equilíbrio entre eles é o que representa uma boa performance de um classificador. No caso de estudo, temos um F1-score em aproximadamente 86%, o que mostra uma boa capacidade de identificação das intenções dos usuários.

#### 2.2.4. *Teste de Falsos Positivos*

Dado que o programa possui um valor de recall moderado, uma alternativa para melhorar a performance seria diminuir o *threshold* de identificação do classificador. Assim, reduziríamos o efeito restritivo da classificação e o recall aumentaria em detrimento da queda do precision, fazendo com que o F1-score aumente.

No entanto, a contrapartida desse efeito é o aumento da incidência de falsos positivos. Do ponto de vista de experiência do usuário, esses falsos positivos tem um impacto devastador, dado que eles mudam o rumo da conversa para uma direção que o usuário não deseja.

Consideremos o seguinte cenário ilustrativo. O usuário faz uma pergunta que esteja fora do escopo de resposta do programa em estudo, como por exemplo, “Quem é o reitor da USP?”. O esperado é que o chatbot identifique que esta pergunta não faz parte do seu escopo de conversa e gerencie a situação através de uma reação de rejeição. Uma resposta esperada seria algo como “Desculpe, não entendi”. No entanto, o programa pode identificar incorretamente que o usuário deseja saber quem é o professor da disciplina do contexto e, assim, fornecer uma informação que não foi solicitada pelo usuário, gerando uma percepção negativa por parte do utilizador.

Reduzir o threshold provavelmente aumentaria a capacidade de classificação do chatbot, no entanto, reduziria a qualidade da experiência do usuário com a introdução de falsos positivos na conversa.

Para garantir uma conversa fluida e evitar a situação descrita anteriormente, o threshold do classificador foi definido com o objetivo de garantir uma boa performance de classificação, mas mantendo a incidência de falsos positivos controlada, cujo índice deve ser menor do que 30%.

Para avaliar esse índice, foi realizado um segundo teste, no qual foram enviadas ao programa 31 sentenças que estão fora do seu escopo de respostas. Espera-se que o programa identifique todas as sentenças de forma adequada, não as atribuindo a nenhuma das intenções do conjunto classificável. Caso o programa identifique alguma dessas sentenças com uma intenção válida, esse caso é contado como um falso positivo no teste. Foram deliberadamente introduzidas frases que pertencem ao contexto universitário dos alunos, mas que não se encaixam no escopo de resposta do chatbot. A proximidade do contexto permite uma avaliação estressada da capacidade do programa identificar solicitações indevidas e, portanto, conferem uma característica mais crível ao teste. As sentenças do teste podem ser visualizadas no Apêndice B.

Das 31 sentenças, 9 delas resultaram em um falso positivo, ou seja, uma incidência de 29%. Com isso, conclui-se que o programa tal qual ele foi construído respeita o critério de incidência de falsos positivos definido nas especificações do projeto.

### 3. Análise de Usabilidade

A avaliação de usabilidade de agentes conversacionais é um domínio amplamente discutido na comunidade científica. Ainda que existam diversos esforços no sentido de definir métricas de avaliação desses sistemas, até o presente momento não há uma metodologia amplamente empregada. A usabilidade de agentes conversacionais está muito associada ao contexto de aplicação desse programa, que confere uma grande variabilidade a esse tipo de sistema. Em cada aplicação podem-se utilizar métricas de avaliação específicas do contexto de uso, no entanto, essa prática introduz uma dificuldade de generalizar a análise e comparar a usabilidade com outros agentes.

Uma das avaliações mais conhecidas desse tipo de sistema é o próprio teste de Turing (TURING, 1950). No entanto, segundo (VENKATESH, KHATRI, *et al.*, 2018), esse teste não é um mecanismo adequado para a avaliação de chatbots, dado que um agente pode interagir diferentemente de um ser humano, mas ainda sim ser um bom conversador. Além disso, os autores ressaltam que o objetivo do júri do teste não é avaliar a experiência da conversa, mas sim encontrar alguma brecha no diálogo, o que mostra um claro desalinhamento de objetivos.

Outra célebre avaliação desses agentes é o Prêmio Loebner. Nesta competição uma banca de jurados deve interagir durante alguns minutos com uma série de chatbots e pontuá-los sobre o critério de “naturalidade”. Ganha aquele que obter a maior pontuação. Em (SHAWAR e ATWELL, 2007), os autores concluem não ser adequado avaliar um chatbot somente com base em uma referência, tal qual o Prêmio Loebner. Os autores sugerem que a avaliação deve condizer com as necessidades dos usuários. Ou seja, caso o chatbot tenha sido concebido para performar alguma tarefa ou prestar algum serviço, a melhor avaliação do sistema reside sobre a capacidade ou não de cumprir esse objetivo.

Avaliar a capacidade de cumprir um objetivo não é algo determinístico, uma vez que existem diversos caminhos possíveis de serem percorridos para realizar uma tarefa, no entanto, alguns desses caminhos são inviáveis por serem extensos ou redundantes. Ou seja, mesmo que o sistema encontre a informação desejada pelo usuário, do ponto de vista

de experiência de conversa, se o sistema levou demasiado tempo para responder, o objetivo não foi cumprido, uma vez que se espera uma conversa fluida e objetiva. Por isso, esse tipo de avaliação geralmente envolve a apuração junto aos usuários, sob forma de questionário, quanto ao fato do sistema ter sido capaz ou não de prover adequadamente as informações solicitadas. Afinal, quem melhor para definir se o objetivo foi cumprido do que o próprio usuário?

No entanto, essa abordagem possui uma limitação clara, uma vez que não é economicamente viável conduzir uma tal pesquisa a cada evolução do sistema. Em outras palavras, seria muito difícil conduzir melhorias no chatbot com base nesse tipo de métrica pois, a cada ajuste realizado, seria necessário conduzir uma nova pesquisa com os usuários visando compreender se os ajustes contribuíram ou não para a melhoria da usabilidade como um todo, o que significaria mobilizar grandes recursos e, possivelmente inviabilizando tal abordagem. A natureza custosa de métricas baseadas na perspectiva do usuário, tal qual a métrica descrita anteriormente, levou os pesquisadores a buscarem soluções automatizadas de avaliação de agentes conversacionais.

Dentre elas, algumas métricas como BLEU (PAPINENI, ROUKOS, *et al.*, 2002) e METEOR (BANERJEE e LAVIE, 2005) têm sido popularmente aplicadas para a avaliação de chatbots, por serem fáceis de calcular para um determinado data set de diálogos, sem a necessidade de intervenção humana. Essas métricas partem da premissa de que respostas válidas são aquelas que possuem uma significativa sobreposição de palavras com aquelas respostas esperadas como verdadeiras. No entanto, conforme exposto em (LIU, LOWE, *et al.*, 2016), essa hipótese não é válida dentro do contexto de agentes conversacionais, dado que existe uma significativa diversidade no espaço de respostas corretas para um dado contexto e diálogo. Segundo os autores, tais métricas possuem pouca ou nenhuma correlação com o julgamento humano.

Uma outra alternativa é proposta em (WALKER, LITMAN, *et al.*, 1997), no qual os autores introduzem PARADISE (*PARAdigm for Dialog System Evaluation*), um framework para avaliação da usabilidade de agentes conversacionais. Na metodologia proposta, a usabilidade do sistema é definida como função dos fatores que contribuem para a satisfação do usuário, medidos através de experimento empírico. Essa abordagem visa a construção de um modelo que correlaciona a satisfação do usuário (variável dependente) com uma série de métricas do diálogo (variáveis independentes), como duração da conversa, *task success rate* e incidência de erros. Com esse modelo, é possível avaliar progressivamente a usabilidade do chatbot conforme são feitos ajustes e melhorias, sem a necessidade de pesquisas e questionários complexos a cada nova iteração. Além disso, a metodologia proposta permite comparar agentes que realizam tarefas diferentes através da normalização dos parâmetros de interesse. No presente trabalho, será aplicado o framework PARADISE para a avaliação da usabilidade do chatbot desenvolvido.

### 3.1. Metodologia

Dentro do contexto de agentes conversacionais, o conceito de usabilidade é bem descrito pela norma ISO 9241: “*The extent to which a product can be used by specified*

*users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use*” (ABRAN, KHELIFI, *et al.*, 2003). *Effectiveness*, ou eficácia, diz respeito à capacidade do sistema atingir os objetivos esperados, enquanto a *efficiency*, ou eficiência, se refere a quão bem os recursos foram aplicados para atingir tais objetivos. Por fim, a satisfação diz respeito à forma com que o sistema atinge tais objetivos e tangencia diversos parâmetros, tais quais aspectos visuais e facilidade de utilização (RADZIWILL e BENTON, 2017).

Percebe-se que essa definição de usabilidade apresenta isomorfismo com a estrutura do framework PARADISE, como pode-se observar na Fig. 11. Em outras palavras, nesse contexto, a usabilidade é medida como a função que maximiza a satisfação do usuário através da maximização do cumprimento das tarefas solicitadas – associado à eficácia - e da diminuição do custo atribuído à realização dessas tarefas – associado à eficiência.

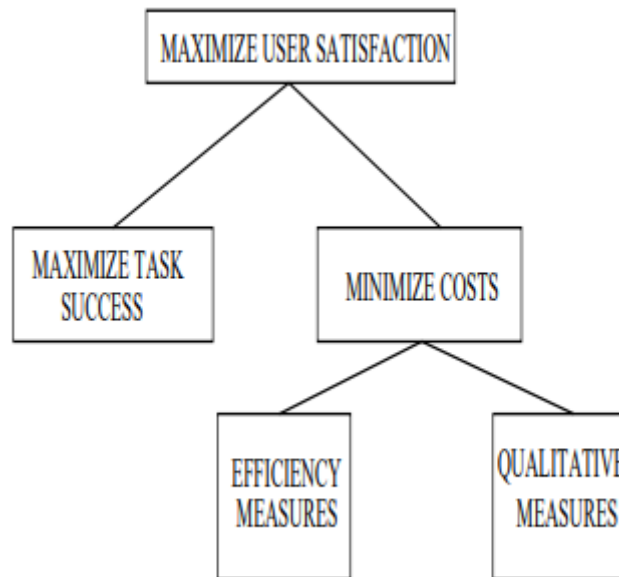


Figura 11 – Estrutura do framework PARADISE. Fonte (WALKER, KAMM e BOLAND, 2000)

A aplicação deste framework envolve a definição de 3 principais parâmetros na avaliação da usabilidade: *user satisfaction score*, *task success rate* e *dialog costs*. Podemos associar cada um desses parâmetros a elementos observáveis, conforme descrito a seguir.

### 3.1.1. *User Satisfaction Score*

Aqui representado como USAT, geralmente é medido através de uma pesquisa de satisfação com os usuários, feita após a utilização do sistema. Em (WALKER, LITMAN, *et al.*, 1997) os autores propõem segregar esse item em outros oito subitens, relacionados a características como velocidade da interação, capacidade de compreensão do sistema e comportamento esperado. Cada item é avaliado com uma nota de 1 a 5 e a pontuação final é a soma de todos os itens, graduada de 8 a 40. O intuito dessa segmentação é auxiliar o

usuário na definição da satisfação abrangendo tais aspectos da conversa. Essa segmentação é indicada para sistemas complexos e sistemas de diálogo falado.

### 3.1.2. Task Success Rate

Representado pelo coeficiente  $\kappa$ , trata-se da métrica que define o atingimento das tarefas solicitadas pelos usuários. Em (WALKER, LITMAN, *et al.*, 1997) os autores propõem que tal coeficiente seja determinado através de uma matriz de confusão que correlaciona as respostas esperadas com as respostas obtidas pelo sistema. No entanto, em um trabalho posterior acerca da mesma metodologia (WALKER, KAMM e BOLAND, 2000), os autores sugerem que a **percepção** do usuário do cumprimento ou não das tarefas é um preditor mais adequado da satisfação. Assim, os pesquisadores sugerem a quantificação da percepção do usuário do cumprimento das tarefas via pesquisa de satisfação.

### 3.1.3. Dialog Costs

Representado pelos coeficientes  $c_i$ , trata-se de uma estimativa dos custos da interação entre o usuário e o sistema. A Tabela 7 indica alguns dos custos propostos pelos pesquisadores.

Tabela 7 – Métricas de custos da interação.

<b>Iterações</b>	Quantidade de interações com o sistema.
<b>Duração</b>	Tempo total da conversa.
<b>MRS</b>	Média do score de classificação da conversa.
<b>Cancelamentos</b>	Quantidade de vezes que o usuário cancelou um pedido.
<b>Ajudas</b>	Quantidade de vezes que o usuário pediu ajuda por não saber o que fazer.
<b>Rejeições</b>	Quantidade de vezes que o sistema não entendeu o pedido.
<b>Erros</b>	Quantidade de vezes que o sistema acusou algum erro.

### 3.1.4. Usabilidade

Definidos os três parâmetros acima, a usabilidade do sistema pode ser representada pela equação abaixo.

$$Usabilidade = (\alpha * N(\kappa)) - \sum_{i=1}^n \omega_i * N(c_i)$$

Onde  $\alpha$  e  $\omega_i$  são coeficientes obtidos através da regressão múltipla que correlaciona o USAT (variável dependente) com os parâmetros  $\kappa$  e  $c_i$  (variáveis



independentes). Ou seja, podem ser interpretados como o peso da contribuição de cada parâmetro no nível de satisfação do usuário.

Além disso,  $N$  é uma função de normalização Z-score dos parâmetros de interesse. Isso é feito pois os parâmetros não possuem necessariamente a mesma escala. A Duração é medida em minutos, enquanto MRS é um score que varia entre 0 e 1. A normalização é dada pela equação abaixo.

$$N(x) = \frac{x - \bar{x}}{\sigma_x}$$

A construção de tal modelo visa identificar quais os parâmetros realmente afetam a percepção de satisfação dos usuários no contexto de aplicação do agente conversacional. Em outras palavras, dependendo da aplicação do agente, a agilidade na resposta pode ser mais impactante no nível de satisfação do que a precisão da resposta, ou vice-versa. Ou seja, a avaliação desses modelos é circunstancial, daí a dificuldade em se obter uma metodologia amplamente aplicada na comunidade científica.

No entanto, com a aplicação do framework PARADISE, pode-se obter um modelo que identifica os principais fatores que impactam a percepção de satisfação dos usuários e a devida contribuição de cada fator. Além disso, pela normalização dos parâmetros, pode-se obter um modelo minimamente generalizado e, portanto, compará-lo com outros agentes que realizam tarefas distintas. Por fim, essa metodologia permite contornar o problema de mobilização de recursos para pesquisas com os usuários. Nesse caso, é necessário realizar apenas uma pesquisa de satisfação para construir o modelo. Uma vez construído tal modelo, sendo ele estatisticamente significativo, pode-se atuar diretamente nos fatores que impactam a usabilidade e medir a performance do chatbot a cada nova iteração de ajustes e melhorias.

### 3.2. Descrição do Experimento

Com intuito de aplicar a metodologia proposta, foi realizado um experimento no qual os usuários foram convidados a interagir com o chatbot desenvolvido, buscando atingir dois objetivos principais: obter uma sugestão de disciplina com base nos seus critérios de decisão e encontrar informações da disciplina em questão. Que são as duas principais funcionalidades do agente.

Após a interação com o sistema, o usuário deve preencher um questionário que diz respeito à experiência de utilização. Dentre as perguntas do questionário, duas delas dizem respeito ao cumprimento das tarefas solicitadas.

- ✓ “Todas as informações das disciplinas foram encontradas?”
- ✓ “As disciplinas sugeridas estão de acordo com o esperado?”

A resposta de cada uma é categorizada em “Sim”, “Não” e “Parcialmente”. Para fins da aplicação da metodologia proposta, é feita a conversão dessas categorias em pontuação de acordo com a Tabela 8. Dessa forma, pode-se considerar o *task success rate*  $\kappa$  como a média da pontuação dos dois objetivos, para cada usuário do experimento.

Tabela 8 – Pontuação das respostas categóricas.

<b>Categoria</b>	<b>Pontuação</b>
Sim	1
Não	0
Parcialmente	0,5

Além disso, no questionário também é coletada uma nota quanto a satisfação geral do usuário, medida de 1 a 5. Essa nota pode ser correlacionada diretamente com o parâmetro *user satisfaction score* (USAT) da metodologia proposta. Por medida de simplicidade, optou-se por utilizar apenas uma métrica de satisfação, ao invés da soma dos oito parâmetros que compõem a mesma métrica de satisfação, conforme proposto na metodologia. No exemplo proposto em (WALKER, LITMAN, *et al.*, 1997) os autores fazem a mesma proposição.

Além do questionário, também foram analisadas as interações entre os usuários e o chatbot. Dessa análise, foram identificados os seguintes parâmetros que compõem os *dialog costs*.

Tabela 9 – Parâmetros de custo da interação levantados no experimento.

<b>Interações</b>	Quantidade de interações com o sistema.
<b>Duração</b>	Tempo total da conversa medido em minutos.
<b>Cancelamentos</b>	Quantidade de vezes que o usuário cancelou um pedido.
<b>Ajudas</b>	Quantidade de vezes que o usuário pediu ajuda por não saber o que fazer.
<b>Rejeições</b>	Quantidade de vezes que o sistema não entendeu o pedido.

Por uma questão de limitação técnica da plataforma de desenvolvimento que foi utilizada no projeto, não foi possível medir o MRS (*Mean Recognition Score*), então este parâmetro não foi incluído na análise. Além disso, devido à forma com que o sistema foi construído, a incidência de erros, como timeouts, é ínfima. Isso porque a API responsável por buscar as informações solicitadas não é um sistema terceiro, mas sim um sistema desenvolvido pelos próprios autores desse trabalho. Por ser um sistema dedicado para essa tarefa, a incidência de erros é pequena e, portanto, tal parâmetro não foi considerado na análise.

Após realização do experimento e com esses parâmetros em mãos, é possível correlacioná-los para encontrar o modelo que explicita a contribuição de cada parâmetro no nível de satisfação do usuário, conforme metodologia proposta.

### 3.3. Resultados

Nesta seção serão expostos os resultados do experimento apresentado anteriormente, bem como a construção do modelo de avaliação da usabilidade conforme metodologia PARADISE proposta na literatura.

### 3.3.1. Sumário do Experimento

O experimento proposto foi realizado com 23 usuários, dentre alunos e professores do curso de Engenharia Mecatrônica da USP. Foram ao todo 398 interações com o robô durante o experimento. A pesquisa de satisfação mostrou que, segundo os usuários, o programa obteve boa performance na realização das tarefas solicitadas, conforme apresentado na Fig. 12. A média de atingimento de ambos objetivos foi medida em cerca de 75%, considerando a pontuação apresentada anteriormente.

Além disso, a avaliação geral de experiência do usuário mostrou um nível de satisfação de 3.9, sendo 5 a nota máxima. Ou seja, representa um valor médio de 78% no quesito satisfação do usuário.

No contexto de *task-oriented dialog agents*, essas duas métricas – atingimento de objetivo e satisfação do usuário – são uma boa proxy da usabilidade do sistema (CHU-CARROLL, 2000). No entanto, visando obter um modelo escalável e generalizável de avaliação de usabilidade, aplicaremos por completo a framework PARADISE proposta.

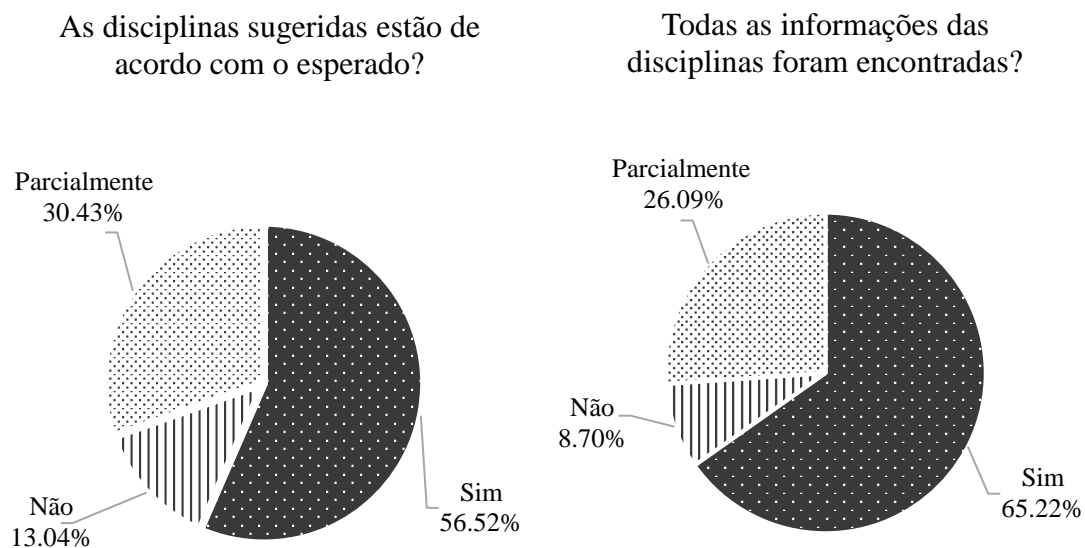


Figura 12 - Resultado do cumprimento das tarefas solicitadas.

### 3.3.2. Matriz de Parâmetros

A partir da realização do experimento, foi possível construir a Tabela 10 que apresenta a atribuição de cada usuário aos parâmetros pertinentes da metodologia apresentada. As colunas Interações, Duração, Rejeição, Ajuda e Cancelamento são referentes aos *dialog costs*. Os coeficientes  $\kappa_1$  e  $\kappa_2$  se referem à pontuação de atingimento das duas tarefas propostas no experimento, sendo a *task success rate*  $\kappa$  a simples média

aritmética entre esses dois coeficientes. Por fim, o *user satisfaction score* é representado pelo coeficiente parâmetro USAT. Para auxiliar a normalização, também foram expostos os valores médios e desvios padrões da distribuição de cada parâmetro.

Tabela 10 – Matriz de parâmetros do experimento.

Usuário	Interações	Duração	Rejeição	Ajuda	Cancelamento	$\kappa_1$	$\kappa_2$	$\kappa$	USAT
1	25	11	8	0	4	1	0.5	0.75	4
2	20	8	2	0	1	0.5	1	0.75	4
3	23	6	5	1	0	1	1	1	4
4	25	5	5	1	0	1	1	1	5
5	21	11	7	0	1	0.5	1	0.75	3
...	...	...	...	...	...	...	...	...	...
19	15	5	7	1	0	0.5	0.5	0.5	2
20	20	12	3	2	1	1	1	1	5
21	7	3	2	2	0	1	0.5	0.75	4
22	11	4	4	2	0	0	0.5	0.25	3
23	29	13	3	3	0	1	1	1	5
$\bar{x}$	<b>17.30</b>	<b>7.43</b>	<b>4.57</b>	<b>1.22</b>	<b>0.39</b>	<b>0.78</b>	<b>0.72</b>	<b>0.75</b>	<b>3.87</b>
$\sigma_x$	<b>7.74</b>	<b>4.70</b>	<b>2.64</b>	<b>0.95</b>	<b>0.94</b>	<b>0.33</b>	<b>0.36</b>	<b>0.27</b>	<b>1.01</b>

### 3.3.3. Correlação

Com o objetivo de construir um modelo que explique a variável de interesse (USAT) a partir dos parâmetros do experimento, começamos analisando a correlação entre os fatores observados. A Fig. 13 apresenta o coeficiente de Pearson entre os parâmetros de interesse, sob forma de mapa de calor. Pode-se observar que a variável de interesse é fortemente correlacionada com dois fatores: *task success rate* ( $\kappa$ ) e a quantidade de rejeições, ou seja, quantidade de vezes que o chatbot não compreendeu a solicitação do usuário.

Como era de se esperar, o coeficiente  $\kappa$  tem uma correlação positiva, ou seja, um maior atingimento dos objetivos significa uma melhor avaliação por parte dos usuários quanto à usabilidade do sistema. A dinâmica contrária é esperada para a quantidade de rejeições, ou seja, uma grande incidência desses eventos exprime uma experiência ruim de interação com o sistema. Partiremos então da hipótese de que os fatores  $\kappa$  e Rejeição são bons candidatos a parâmetros explicativos da variável de interesse USAT.

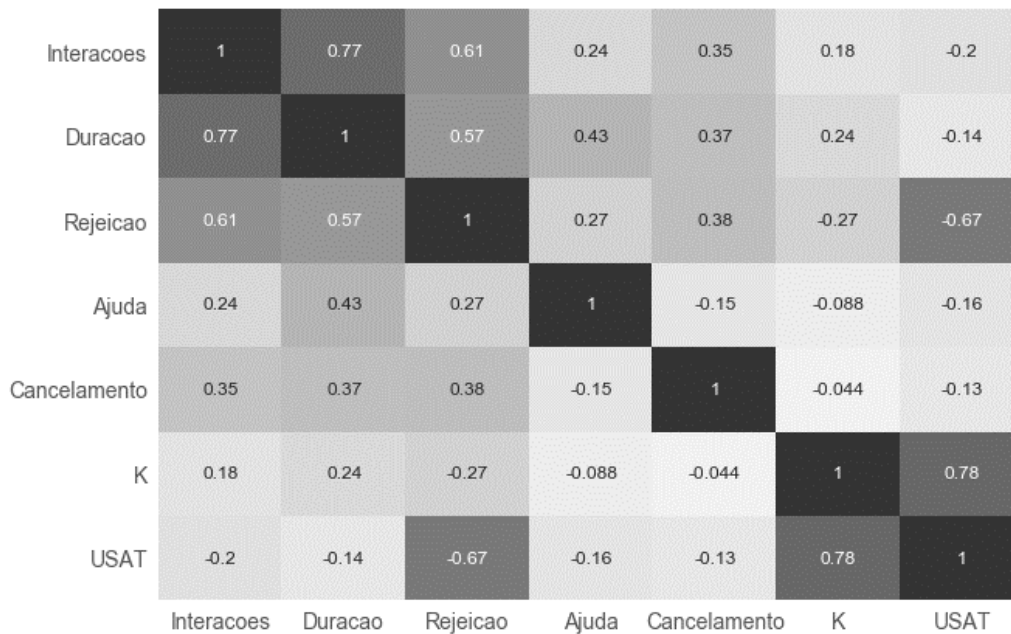


Figura 13 - Coeficiente de Pearson entre os parâmetros de interesse.

### 3.3.4. Multicolinearidade e Heteroscedasticidade

Segundo o framework proposto, os coeficientes que que exprimem a relação entre a variável de interesse e os parâmetros observados podem ser calculados via regressão linear múltipla. Para garantir a correta estimação desses coeficientes, é preciso garantir a validade de duas condições: não existe multicolinearidade entre as variáveis independentes e não existe heteroscedasticidade no resíduo da regressão.

A multicolinearidade é o fenômeno no qual as variáveis independentes possuem uma forte relação linear entre si. Essa característica vai de encontro à interpretação natural de uma regressão linear múltipla, que exprime o efeito da variação unitária de uma variável mantendo todas as outras variáveis constantes. Caso estas variáveis tenham uma forte correlação entre si, esse predicado não é mais válido e os coeficientes estimados serão imprecisos. Para garantir essa condição podemos analisar a dispersão entre as duas variáveis independentes identificadas anteriormente,  $\kappa$  e Rejeição, conforme Fig. 14.

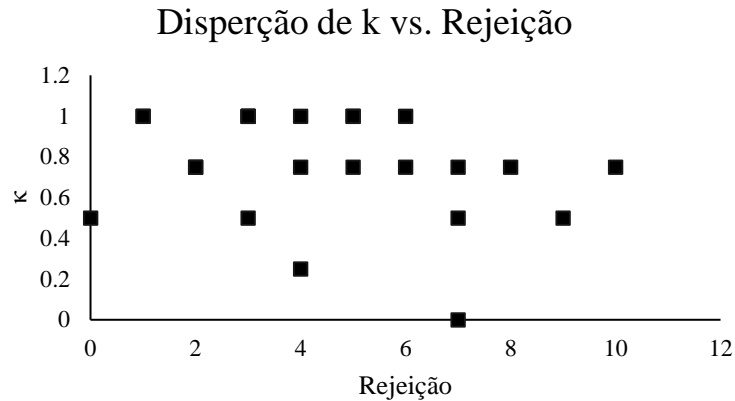


Figura 14 - Análise da colinearidade entre k e Rejeição.

Da dispersão apresentada acima, pode-se perceber que as variáveis não têm uma relação linear explícita. Realizando uma regressão linear entre as duas variáveis, obtemos um valor de  $R^2 = 0.073$ . Ou seja, a variação de um parâmetro explica apenas 7% da variação do outro parâmetro. Um valor pequeno de  $R^2$  é um forte indício de não-linearidade entre os elementos. Assim, podemos rejeitar a hipótese de multicolinearidade entre as variáveis independentes.

A heteroscedasticidade é a característica na qual a variância do resíduo da regressão não é constante ao longo das observações. Essa característica impacta diretamente a regressão ao se utilizar uma técnica como Mínimos Quadrados Ordinários (MQO). Isso porque essa técnica determina os coeficientes da regressão visando minimizar a soma dos quadrados dos resíduos. Então, caso a distribuição apresente regiões de maior variância e regiões de menor variância, o algoritmo atribui implicitamente um peso maior aos quadrados dos resíduos das regiões de maior variância, tornando o estimador MQO ineficiente (PINDYCK e RUBINFELD, 2004).

Uma forma prática de avaliar tal característica é analisar o comportamento do resíduo da regressão entre a variável dependente e os parâmetros de interesse. Na definição da regressão linear, o resíduo é o termo  $\mu$  da equação abaixo.

$$Y = \alpha + X\beta + \mu$$

Na Fig. 15 podemos observar a dispersão entre a variável dependente USAT e os candidatos a parâmetros explicativos, bem como a devida regressão linear entre esses elementos. Pode-se observar que o resíduo de cada regressão é relativamente constante ao longo das observações. Não existe uma tendência de crescimento nem de diminuição do resíduo. Então podemos rejeitar a hipótese de heteroscedasticidade e considerar a homoscedasticidade das variáveis independentes de interesse.

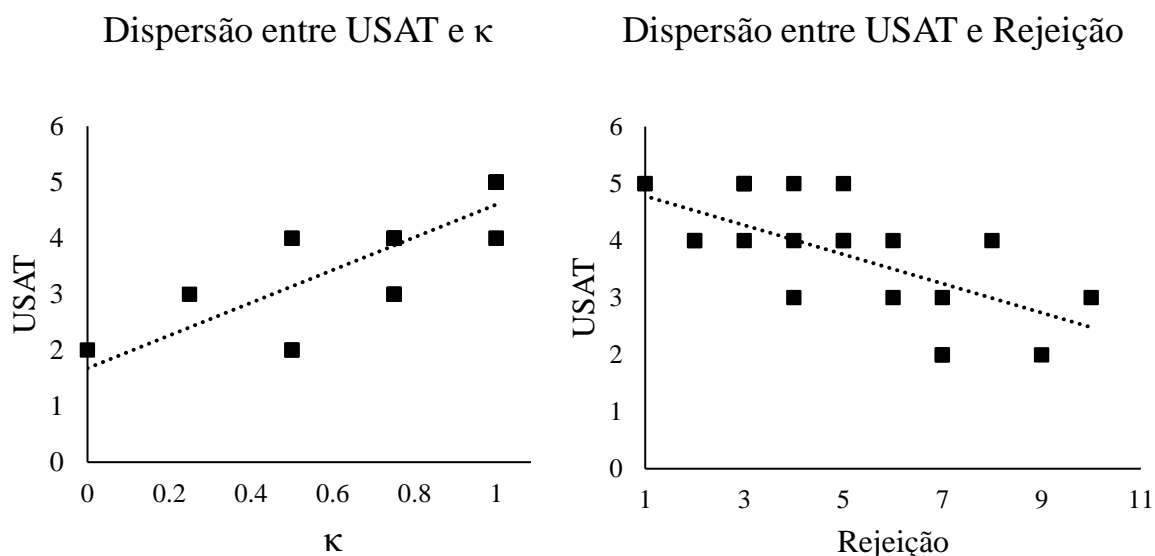


Figura 15 - Análise do resíduo da regressão entre a variável de interesse e os parâmetros explicativos.

### 3.3.5. Modelo de Usabilidade

Podemos enfim calcular os coeficientes  $\alpha$  e  $\omega$  que relacionam a variável de interesse aos candidatos de parâmetros explicativos *Task Success Rate* e Rejeição. Para tal, realizaremos primeiramente a normalização dos parâmetros segundo a média e desvio padrão de cada um, conforme descrito na metodologia. Utilizando o método MQO, obtemos o seguinte resultado da regressão linear múltipla.

Tabela 11 – Resultado da regressão múltipla.

$R^2$	0.83
Amostra	23

Variável	Coefficiente	Desvio Padrão	t	p-valor
Intercepto	3.7826	0.098	38.633	0.000
K	0.7512	0.104	7.227	0.000
Rejeição	-0.4717	0.104	-4.538	0.000

O valor de  $R^2 = 0.830$  sugere que os parâmetros escolhidos são boas variáveis explicativas do valor de USAT. Analisando a estatística t pelo método p-valor, podemos rejeitar a hipótese nula para cada parâmetro com um nível de significância de 1% e concluir que os coeficientes obtidos são estatisticamente significativos.

Adicionando outros parâmetros à regressão, como Duração do diálogo e quantidade de Interações, obtemos valores muito baixos da estatística t e não é possível rejeitar a hipótese nula. Isso confirma a hipótese inicial de que tais parâmetros não são bons candidatos a variável explicativa do fator USAT.

Assim, conforme proposto no framework PARADISE de avaliação de agentes conversacionais (WALKER, LITMAN, *et al.*, 1997), podemos obter o seguinte modelo que exprime a usabilidade do sistema em função dos parâmetros *task success rate* e *dialog costs*.

$$Usabilidade = 0.751 * N(\kappa) - 0.472 * N(Rejeição)$$

Esse modelo exprime os principais fatores que afetam a usabilidade percebida pelos usuários, bem como o grau de contribuição de cada fator na usabilidade geral do sistema. Com tal modelo, é possível ajustar o chatbot no sentido de maximizar a usabilidade do programa, sem ter de realizar inúmeras pesquisas de satisfação com os usuários. Pode-se então avaliar continuamente a performance do sistema, bem como analisar a performance de subdiálogos através da aplicação da mesma metodologia sobre o modelo obtido. Assim, é possível realizar um ajuste fino das funcionalidades do agente.

#### 4. Discussão e Continuação

No presente trabalho foi desenvolvido um chatbot cujo intuito é auxiliar os estudantes na escolha de disciplinas e na busca de informações das mesmas. Essa tarefa é geralmente realizada através do Jupiterweb, portal de disciplinas da Universidade de São Paulo. Tal interação se baseia majoritariamente numa interface gráfica, na qual os usuários dispõem de botões e barras de pesquisas como recursos para realizar tais tarefas. Nesse contexto, toda a complexidade de encontrar, comparar e filtrar os dados fica a cargo do próprio usuário.

Assim, o chatbot construído é proposto como uma alternativa à interação por interface gráfica. Essa solução visa a interação entre usuário e máquina via linguagem natural, na qual os usuários demandam explicitamente as tarefas e a complexidade de processar tais tarefas fica a cargo do próprio sistema. Essa forma de interação é mais intuitiva e familiar, por ser muito próxima do hábito conversacional dos seres humanos, o que tende a elevar o nível de satisfação dos usuários ao utilizar tal sistema.

A forma de interação com o chatbot permite o uso de abreviações das palavras e até erros ortográficos, facilitando assim o seu uso e agilizando o diálogo. Do ponto de vista técnico, o agente apresentou uma boa performance na interpretação de linguagem natural, com um valor de F1-score de 86,50% no teste de classificação das intenções dos usuários. Além disso, a pesquisa de satisfação realizada com os usuários apontou um grau de atingimento das tarefas solicitadas de 75% e uma satisfação geral no quesito usabilidade de cerca de 78%, o que mostra resultados modestos, porém promissores, mesmo com um número limitado de funcionalidades. Com alguns ajustes o chatbot pode



se tornar ainda mais versátil e apresentar ainda mais naturalidade na conversa. O modelo construído que correlaciona a usabilidade com algumas variáveis explicativas poderá então ser usado para avaliar o impacto de tais ajustes na performance do programa.

O programa possui, no entanto, algumas limitações claras, como o fato de não lidar bem como a solicitação de múltiplas tarefas na mesma sentença. Ou seja, caso o usuário demande algo como “me diga os professores e os horários da disciplina”, o programa provavelmente responderia com apenas uma das informações. No caso de utilização em questão, o usuário deve separar cada tarefa em uma interação. Além disso, o chatbot não lida bem com negativas. Ou seja, caso o usuário solicite algo como “me diga as matérias que não envolvem inteligência artificial”, o programa provavelmente buscaria disciplinas nesse tema. O sistema também possui uma limitação técnica quanto a exposição de informações, uma vez que não é possível inserir quebras de linhas nas respostas, dificultando assim a visualização da conversa por parte dos usuários.

Além disso, as principais recomendações dos usuários após a utilização do sistema dizem respeito ao gerenciamento das rejeições. Ou seja, muitas vezes, quando o agente não entendia a solicitação do usuário, acabava por reiniciar a conversa ao invés de oferecer algumas opções de escolha ou mensagens de confirmação. Assim, uma possível continuação do projeto, no que diz respeito à dinâmica do sistema, é melhorar o design do diálogo e focar principalmente no gerenciamento de rejeições. O impacto desse fator também pôde ser observado na construção do modelo de usabilidade.

Por fim, uma vez construído e validado, os próximos passos do projeto envolvem a possível integração de tal sistema com o portal Jupiterweb. Através dessa integração será possível ter acesso a informações de suma importância para diversificar as funcionalidades do robô, como por exemplo as matérias já cursadas do aluno, a quantidade de créditos faltante, a diagonal de formatura, estatísticas das disciplinas, entre outros. Através dessa integração pode-se refinar o programa para que ele não somente realize a busca de informações, mas também realize ativamente algumas tarefas como inscrição em disciplinas e pedidos de requerimento, o que agregaria bastante valor ao projeto.

## REFERÊNCIAS BIBLIOGRÁFICAS

---

ABHISHEK AGARWAL, M. B. **Generation in Graphical User Interface**. International Conference on Multidisciplinary Research & Practice, v. I, n. VIII, p. 673-680, novembro 2014. ISSN ISSN 2321-2705.

ABRAN, A. et al. **Consolidating the ISO Usability Models**. 11th International Software Quality Management Conference and the 8th Annual INSPIRE Conference. [S.l.]: [s.n.]. 2003.

BANERJEE, S.; LAVIE, A. **Meteor: An automatic metric for mt evaluation with improved**. Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. [S.l.]: [s.n.]. 2005. p. 65-72.

BEKKAR, M.; DJEMAA, H. K.; ALITOCHE, T. A. **Evaluation Measures for Models Assessment over Imbalanced Data Sets**. Journal of Information Engineering and Applications, v. 3, n. 10, 2013. ISSN ISSN 2225-0560.

CHU-CARROLL, J. **MIMIC - An Adaptive Mixed Initiative Spoken Dialogue System for**. Proceedings of the sixth conference on Applied natural language processing. Washington: ANLC. 2000. p. 97-107.

FREITAS, L. A. D.; VIEIRA, R. **Ontologias e Língua Portuguesa**. Anais do CELSUL, 2008.

GUPTA, S. et al. **An E-Commerce Website based Chatbot**. International Journal of Computer Science and Information Technologies , v. 6, p. 1483-1485, 2015.

GUZMÁN, I.; PATHANIA, A. **Chatbots in Customer Service**. Accenture Interactive. [S.l.]. 2016.

HOSSIN, M.; SULAIMAN, M. N. **A Review on Evaluation Metrics for Data Classification Evaluations**. International Journal of Data Mining & Knowledge Management Process, v. 5, n. 2, Março 2015.

JOHN D. GOULD, C. L. **Designing for Usability: Key Principles and What Designers Think**. Communications of the ACM, v. 28, p. 300-311, Março 1985.

JURAFSKY, D.; MARTIN, J. H. **Speech and Language Processing - An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition**. 3. ed. [S.l.]: [s.n.], 2018. 422-459 p.

KEACH, S. **This Parking Ticket-Fighting "Robot Lawyer" Is the Hero the World Needs**. TIME, 2016. Disponível em: <[http://time.com/4386361/robot-lawyer-drivers-parking-tickets/?xid=time\\_socialflow\\_twitter](http://time.com/4386361/robot-lawyer-drivers-parking-tickets/?xid=time_socialflow_twitter)>. Acesso em: 25 Setembro 2018.

LARSON, S. **Baidu is bringing AI chatbots to healthcare**. CNN Tech, 2016. Disponível em: <<https://money.cnn.com/2016/10/11/technology/baidu-doctor-ai-melody/index.html>>. Acesso em: 25 Setembro 2018.

LIU, C. W. et al. **How NOT To Evaluate Your Dialogue System - An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation**. [S.l.]: arXiv:1603.08023. 2016.

MICROSOFT. <https://www.msxiaobing.com/>, 2014. Acesso em: 25 Setembro 2018.

MILWARD, D.; BEVERIDGE, M. **Ontology-Based Dialogue System**. Advanced Computation Laboratory, Cancer Research UK, 2003.

NIELSEN, J. **Browser and GUI Chrome**. Nielsen Norman Group, 2012. Disponível em: <<https://www.nngroup.com/articles/browser-and-gui-chrome/>>. Acesso em: 25 Setembro 2018.

PAPINENI, K. et al. **Bleu: a method for automatic evaluation**. Proceedings of the 40th annual meeting on association for computational linguistics. [S.l.]: Association for Computational Linguistics. 2002. p. 311-318.

PINDYCK, R. S.; RUBINFELD, D. L. **Econometria - Modelos & Previsões**. Rio de Janeiro: Elsevier, 2004.

RADZIWILL, N.; BENTON, M. **Evaluating Quality of Chatbots and Intelligent Conversational Agents**. CoRR, v. abs/1704.04579, 2017.

SAVENKOV, K. **NLU/Intent Detection Benchmark**. Inten.to, 2017. Disponível em: <<https://www.slideshare.net/KonstantinSavenkov/nlu-intent-detection-benchmark-by-intento-august-2017>>. Acesso em: 25 Setembro 2018.

SHAWAR, B. A.; ATWELL, E. **Different measurements metrics to evaluate a chatbot system**. Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies. New York: [s.n.]. 2007. p. 89-96.

STAPOR, K. **Evaluating and comparing classifiers - review, some recommendations and limitations**. Proceedings of the 10th International Conference on Computer Recognition Systems. [S.l.]: Advances in Intelligent Systems and Computing. 2018.

STORL, U. et al. **Schemaless NoSQL Data Stores – Object-NoSQL Mappers to the Rescue?** [S.l.]: [s.n.]. 2015.

TURING, A. M. **Computing machinery and intelligence**. Mind 49, p. 433-460, 1950.

VEGESNA, A.; JAIN, P.; PORWAL, D. **Ontology Based Chatbot (For E-Commerce Website)**. International Journal of Computer Applications, v. 179, n. 14, Janeiro 2018.

VENKATESH, A. et al. **On Evaluating and Comparing Conversational**, Janeiro 2018.

WALKER, M. A. et al. **PARADISE: A Framework for Evaluating Spoken Dialogue Agents**. Proceedings of the 35th Annual Meeting of the Association of Computational Linguistics. [S.l.]: [s.n.]. 1997.

WALKER, M.; KAMM, C.; BOLAND, J. **Developing And Testing General Models Of Spoken Dialogue System Performance**. LREC, 2000.

WALLACE, R. **Chatbots. A.L.I.C.E**, 1995. Disponível em: <<https://www.chatbots.org/chatbot/a.l.i.c.e/>>. Acesso em: 8 maio 2018.

WEIZENBAUM, J. **ELIZA - A Computer Program For the Study of Natural Language Communication Between Man and Machine**. Communication of the ACM, 9, Janeiro 1966. 53-62.

# APÊNDICES

---

## 1. Apêndice A – Conjunto de dados do Teste de Classificação

---

### **Intenção esperada: GetCourseCredit**

---

são quantos créditos?  
quantos créditos-aula?  
tem pelo menos 4 créditos?  
preciso que tenha pelo menos 1 crédito  
essa matéria tem 2 créditos-aula?  
essa disciplina tem 1 crédito-aula e 2 créditos-trabalho?  
são qntos créditos?  
tem ao menos 10 crédito?  
sabe me dizer qntos créditos são?  
me informe quantos creditos por favor  
tem o mínimo de créditos que eu preciso para passar?  
tem 4 créditos-aula?  
com essa disciplina eu consigo completar os créditos que faltam?  
com essa eu completo os créditos minimos?

---

---

### **Intenção esperada: GetCourseDescription**

---

qual a descrição?  
pode me dar informações?  
quero saber mais sobre essa disciplina  
me conte mais sobre essa matéria  
estou interessado, pode me falar mais?  
quero saber sobre PMR3407  
pode me contar sobre Sistemas de Informação?  
eu queria saber mais sobre  
pode me infomar? estou interessado  
pode me passar a descrição da matéria?  
tem a descricao da matéria?  
quero saber mais!  
me conte mais  
vc tem mais infos?  
qual o objetivo da disciplina?  
voce tem um resumo da disciplina?  
tem algum resumo?  
é sobre o que essa matéria?  
do que se trata a matéria?

a matéria é interessante?  
pode me mandar o programa resumido?  
gostei, quero saber sobre ela

---

---

### **Intenção esperada: GetCourseRequirements**

---

pmr3304 tem requisito?  
quais os requisitos dessa disciplina?  
algum requisito?  
preciso fazer alguma matéria antes de PMR3304?  
precisa ter feito cálculo I?  
precisa ter feito alguma matéria antes dessa?  
poderia me passar os requisitos da matéria?  
tem algum pré-requisito?  
tem pré requisito?  
tem prerequisite?  
quais as matérias eu tenho de cursar antes dessa?  
se eu não fiz cálculo I posso fazer essa mesmo assim?  
se não passei em física, posso cursar essa matéria mesmo assim?  
quais matérias vêm antes dessa?  
me diga se tem algum pré requisito  
por favor me informa os requisitos  
tem mais de um requisito?  
poderia me dizer se eu cumpri todos os requisitos da matéria?  
poderia me informar se eu validei os requisitos?

---

---

### **Intenção esperada: GetCourseSchedule**

---

qual o horário?  
qual o oferecimento?  
é oferecida em quais dias?  
a matéria é dada em quais dias da semana?  
quero saber os horários da disciplina por favor  
me diz em que dias a matéria é dada  
tem de quarta-feira?  
a matéria é oferecida quarta-feira?  
essa matéria se encaixa na minha agenda de terça-feira?  
tem alguma turma de segunda?  
quais as turmas de terça-feira tem vaga disponível?  
você sabe se oferecem essa turma de sábado?  
tem algum dia da semana entre 8h e 12h?  
quais horários?  
quais dias?  
horas?

é a tarde?  
é de segunda?  
se não me engano é dada terça-feira, correto?  
você sabe se tem alguma turma no período da noite?  
tem à noite?

---

---

---

**Intenção esperada: GetCourseTeacher**

---

---

quem é o professor?  
quem dá essa matéria?  
quem ministra a disciplina?  
quem oferece a matéria?  
quais são os professores?  
quem ministra?  
os professores são bons?  
são professores do departamento?  
quais os nomes dos professores?  
tem mais de um professor?  
qual o melhor professor?  
eu já tive aula com o professor?  
me fale sobre quem oferece a matéria  
quem é o docente?  
quem dá?  
quem oferece?  
qual o coordenador da matéria?  
quem é o coordenador?

---

---

---

**Intenção esperada: GetCourseWorkload**

---

---

qual a carga horária da disciplina?  
qual a carga da matéria?  
quantas horas?  
qual o total de horas?  
quantas horas no semestre?  
são muitas horas no semestre?  
a carga horária é muito alta?  
eu vou dar conta da carga?  
é muito pesado?  
você acha que eu aguento?  
não são muitas horas por semana?  
não são horas demais?  
você não acha que é muita carga?  
quantas horas por semana tem?  
me diga se são muitas horas semanais

---

## 2. Apêndice B – Conjunto de dados do Teste de Falsos Positivos

---

**Intenção esperada: Rejeição**

---

quantos anos tem a faculdade?  
quem é o reitor da USP?  
quanto tempo falta para eu me formar?  
quantos anos você tem?  
qual a hora do intervalo?  
quanto custa o almoço?  
sou aluno de engenharia e você?  
você é um aluno também?  
quero discutir sobre política  
quem é o atual presidente do país?  
qual o resultado das eleições  
vc já se formou?  
você já completou a graduação?  
me conte uma piada  
quero comprar um livro de cálculo I  
estou procurando um artigo científico sobre castores  
como as células se dividem?  
quantos litros de água tem o corpo humano?  
você sabe matemática?  
quantas vezes eu tenho de ir para a aula?  
qual sua música preferida?  
você sabe recitar poemas?  
sou aluno da faculdade de direito, você conhece?  
qual a média de notas dessa disciplina?  
qual a frequência mínima dessa aula?  
qual o método de avaliação dessa matéria?  
quantos alunos estão inscritos?  
qual a bibliografia dessa matéria?  
você pode me sugerir um livro?  
tem algum curso online sobre isso?  
você conhece ensino à distância?

---