
**CENTRO UNIVERSITÁRIO DA GRANDE DOURADOS**

**DIONES DE SOUZA CALÇA**
**MAURÍCIO ARAÚJO SCOLARI**


# BEL - SISTEMA ITERATIVO PARA METODOLOGIA DE ENSINO-APRENDIZAGEM


**Dourados**
**2019**

CENTRO UNIVERSITÁRIO DA GRANDE DOURADOS

DIONES DE SOUZA CALÇA
MAURÍCIO ARAÚJO SCOLARI

# BEL - SISTEMA ITERATIVO PARA METODOLOGIA DE ENSINO-APRENDIZAGEM

Monografia apresentada ao Curso de Ciência da Computação da Faculdade de Ciências Exatas e Agrárias do Centro Universitário da Grande Dourados. Como pré-requisito para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Profº Andre Martins do Nascimento

Dourados
2019

## RESUMO

Apresentando uma série de conceitos e argumentações que, embasam os pressupostos pelos quais o projeto se orientou, como o Processamento de Linguagem Natural, Inteligência Artificial, bibliotecas de captura e sintetizadores de voz, esta monografia descreve as características e possibilidades de usos do aplicativo Bel, software idealizado como ferramenta de ensino-aprendizagem na disciplina de Arquitetura de Computadores.



**Palavras-chave:** Bel, Inteligência Artificial, Processamento de Linguagem Natural,ensino-aprendizagem

## ABSTRACT

This monograph describes the characteristics and possibilities of uses of the Bel application, presenting a series of concepts and arguments that support the assumptions underlying the project, such as Natural Language Processing, Artificial Intelligence, capture libraries and voice synthesizers. software idealized as a teaching-learning tool in the discipline of Computer Architecture.



**Keywords:** Bel, Artificial Intelligence, Natural Language Processing, Teaching-learning

## LISTA DE ILUSTRAÇÕES

**SUMÁRIO**

# 1 INTRODUÇÃO

A comunicação é um fenômeno que se torna difícil estabelecer uma definição exata por ser tão generalizado. Mais no geral, ela é um processo de interação no qual compartilhamos e recebemos mensagens, ideias, sentimentos e emoções. Para o humanos, que tem uma grande facilidade de se comunicar sem muito esforço, levando em conta o conhecimento prévio comum entre os interlocutores, como por exemplo a mesma língua, a mesma bagagem cultural e assim por diante, é utilizado um sistema de sinais estruturado e complexo para interagir, expressar o raciocínio e trocar informação, denominado de linguagem natural.

Dentro dos campos de estudos de Inteligência Artificial (IA), o Processamento de Linguagem Natural (PLN) é uma área utilizada para a compreensão e produção de linguagem natural, possibilitando que as comunicações homens—máquina seja da forma mais "natural"possível. Este campo abre as diversas Possibilidades de aberturas de aplicações que possam simular a própria capacidade dos seres humanos em máquinas, para automatização de determinadas tarefas, dentre elas, ferramentas de ensino-aprendizagem.

Essas ferramentas para fins educacionais demonstram-se interessantes, visto que permite que usuários se relacionem de forma mais natural com o computador. Elas também oferecem outras vantagens em um ambiente de aprendizado, algumas delas está na disponibilidade da ferramenta para o aluno 24 horas por dia, além do controle e auxilio da aplicação de conteúdos ministrados pelo orientador.

Esse trabalho busca então mostrar através de um levantamento bibliográfico, investigar tanto os mecanismos que possibilitam a comunicação entre seres humanos e computadores quanto o uso da inteligência artificial e o processamento de linguagem natural como uma ferramenta complementação no aprendizado de estudantes focada no ensino de arquitetura de computadores.

## 1.1 OBJETIVOS

### 1.1.1 OBJETIVO GERAL

O objetivo geral deste trabalho é a avaliação de uma ferramenta didática-pedagógica autoral, focada no ensino de arquitetura de computadores. Utilizando-se de recursos de processamento de linguagem natural e interação realizados por voz, a avaliação da ferramenta envolve os aspectos que vão desde a capacidade de reconhecer a linguagem natural do ser humano e atender as suas necessidades, até a sua utilização complementar no contexto de sala de aula.

### 1.1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos podem ser identificados como:

- Avaliar a facilidade de interação que será proporcionada pelo sistema.
- Promover uma pesquisa que enfoque as ideias e componentes necessário para o desenvolvimento do projeto.
- Analisar bibliotecas necessárias para o desenvolvimento do projeto.
- Avaliar a usabilidade da ferramenta dentro do contexto de aprendizagem.

## 1.2 JUSTIFICATIVA E MOTIVAÇÃO

As tecnologias são importantes na educação e podem auxiliar na aplicação de conteúdos ministrados por um tutor ou professor. No entanto, as aplicações na área educacional parecem estar em uma fase inicial. Isso nos leva a acreditar que o uso deste tipo de ferramenta é ainda muito pouco explorado em questões tanto de estudo quanto em desenvolvimento. Com isso, juntamente com o interesse mútuo de pesquisa na área de inteligencia artificial com foco em desenvolvimento de um sistema capaz de realizar interações utilizando a linguagem natural se surge a motivação de colaborar com essa demanda que fornecerá mais uma alternativa de ferramenta didática a fim de contribuir com algo mais valioso: o conhecimento.

## 1.3 METODOLOGIA

Visando atingir o objetivo proposto para este trabalho, foi realizado uma pesquisa aplicado sobre processamento de linguagem natural, utilizando técnicas de aprendizagem de maquina, linguagem de programação Python, banco de dados PostgreSQL, Framework para desenvolvimento web Django e ambiente de desenvolvimento Sublime text. Foram utilizadas as bibliotecas gTTS e SpeechRecognition para interação e o nltk para o processamento de linguem natural.

## 1.4 ORGANIZAÇÃO DO TEXTO

No capítulo 1 é apresentado a introdução, objetivos, metodologia e justificativas para o desenvolvimento do projeto.

No capítulo 2 é descrito um breve conceito de inteligência artificial, apresentando uma história a qual deu origem a esse campo de estudo, e o que define uma máquina ser inteligente. os aspectos de estudo da linguagem natural necessárias para que um sistema seja capaz de compreender e produzir linguagem natural, os problemas dentro do uso do PLN, suas limitações e áreas de aplicação, além de descrever sobre a linguística de corpus, sua definição e qual a sua importância dentro do PLN. Por fim, é apresentado o funcionamento das etapas do PLN dentro de um sistema, de uma forma conceitual.

No capítulo 3 é apresentado a Ferramenta NLTK, Seu funcionamento, características, sua estrutura e motivo de escolha comparada as outras ferramentas que auxiliam o desenvolvimento de Softwares para Processamento de linguagem natural.

No capítulo 4 é apresentada a fundamentação teórica para o desenvolvimento da ferramenta, apresentação das bibliotecas utilizadas, modo de treinamento, e as análises de resultados obtidos no projeto bem como a apresentação dos resultado da pesquisa.

No capítulo 5 por fim, é apresentado as considerações finais e a conclusão obtida dos resultados.

# 2 INTELIGÊNCIA ARTIFICIAL

Quando Alan Turing (1950) começou a "criar"o campo da computação, ele já tinha ideia do que seria uma inteligência artificial. Em 1950, ele publicou essas ideias em seu trabalho "computing machinery and intelligence"onde propõe o "Jogo a Imitação". Nesse jogo, um homem e uma mulher estão em um local que não podem ser vistos como uma sala, e se comunicam com o interrogador de uma forma que não deixe claro quem é o homem ou a mulher como em um chat, e o interrogador deve descobrir quem é quem. O papel do homem é confundir o interrogador, e o da mulher é ajudar interrogador a fazer a escolha certa. Para Turing, máquinas poderiam ser consideradas inteligentes se participassem do jogo e conseguissem confundir o interrogador e se passar por um humano respondendo as mensagens. Esse teste ficou conhecido como "Teste de Turing".

A partir dessa ideia, a Inteligência Artificial (IA) começou a ser vista como um campo experimental e estudos, mais especificamente no ano de 1956 na Conferência de Darthmouth College juntamente com os pioneiros Allen Newell, Herbert Simon, John McCarty e Marvin Minsky (MCCORDUCK, 2004). Contudo, desde o começo a IA gerou polêmica, começando pelo seu próprio nome sendo considerado presunçoso por alguns, até a definição de seus objetivos e metodologias (BITTENCOURT, 2011). Além disso, tinha também o desconhecimento dos princípios que fundamentam a inteligência, por um lado, e dos limites práticos da capacidade de processamento dos computadores, por outro, levou periodicamente a promessas exageradas e às correspondentes decepções (BITTENCOURT, 2011).

Dada a impossibilidade de uma definição formal precisa para IA, visto que para tanto seria necessário definir, primeiramente, a própria inteligência, foram propostas algumas definições operacionais:

> However, work on artificial intelligence, especially general intelligence, will be improved by a clearer idea of what intelligence is. One way is to give a purely behavioural or black-box definition. In this case we have to say that a machine is intelligent if it solves certain classes of problems requiring intelligence in humans, or survives in an intellectually demanding environment. This definition seems vague; perhaps it can be made somewhat more precise without departing from behavioural terms, but we shall not try to do so (MCCARTHY; HAYES, 1969, p. 4).

> Artificial Intelligence (AI) is the part of computer science con-cerned with designing intelligent computer systems, that is, systems that exhibit the characteristics we associate with intelligence in human behavior — understanding language, learning, reasoning, solving problems, and so on (BARR; FEIGENBAUM, 1981, p. 3).

Entre os teóricos, existe uma discussão do que é possível fazer com a IA. onde se consideram duas propostas básicas, as chamadas “IA fraca", sistemas criados capaz se serem melhores em tarefes especificas, como em jogos, robótica, prova de teoremas, resolução de problemas gerais, percepção como a visão e fala, compreensão da linguagem natural, resolução de problemas especializados dentre elas a matemática simbólica, além de análises químicas, diagnósticos e até mesmo em projetos de engenharia. Já uma IA forte ou uma inteligência de propósito geral, seria capaz de aprender todo tipo de assunto e não só classificar informação, mas também capaz de produzir informação nova, a qual nós tomamos como o pensamento (RUSSELL; NORVIG, 2004).

Dado essas definições, podemos presumir que a IA se dá ao fato aos processos e raciocínio que um sistema pode ter com o seu comportamento, e a fidelidade do que seria de um desempenho humano na questão de tomadas de decisões em “fazer a coisa certa” dada às informações as quais foram treinadas e serem melhores em tarefas específicas. Com isso, o objetivo principal de um sistema IA é executar funções da mesma forma que um ser humano fosse executar (RUSSELL; NORVIG, 2004).

Segundo Azevedo (2005 apud ALVES *et al.*, 2017, p. 3), e Bittencourt (2011), existem duas linhas de pesquisa principais e uma mais atual que dividem as investigações de inteligência artificial: a linha conexionista, linha simbólica e a linha evolutiva. A linha conexionista propõe a modelagem da inteligência humana por meio de simulações dos neurônios e suas interligações como sendo a mais comum os modelos de redes neurais artificiais. A linha simbólica envolve a manipulação de símbolos e conceitos abstratos, é um campo de pesquisa de sistemas especialistas, aprendizagem, representação de conhecimento, aquisição de conhecimento, tratamento de informação imperfeita, visão computacional, robótica, controle inteligente, inteligência artificial distribuída, modelagem cognitiva, arquiteturas para sistemas inteligentes, linguagem natural e interfaces inteligentes. E já a linha evolutiva é baseada na observação de mecanismos evolutivos encontrados na natureza, tais como a auto-organização e o comportamento adaptativo.

## 2.1 APRENDIZADO DE MÁQUINA

Aprendizado de máquina (AM), é uma área da Inteligência artificial que lida com problemas de aprendizado computacional a fim de adquirir conhecimento de forma automática (MATOS *et al.*, 2009). Seu funcionamento é baseado em experiências acumuladas, por exemplo, regras de decisão de um conjunto de dados tal que essas regras possam ser aplicadas a novos dados (BATISTA; MONARD, 1998).

Segundo Monard e Baranauskas (2003 apud MATOS *et al.*, 2009, p. 7) classificam o aprendizado de máquina em alguns paradigmas, que compreendem:

- Simbólico, buscam aprender construindo representações simbólicas de um conceito através da análise de exemplos e contraexemplos como expressão lógica, árvore de decisão, regras ou rede semântica. Exemplo: Algoritmos de árvore de decisão.
- Estatístico, consiste em utilizar modelos estatísticos para encontrar uma boa aproximação do conceito induzido. Exemplo: aprendizado Bayesiano.
- Baseado em Exemplos, classifica um novo exemplo com base em uma classificação similar conhecida. Exemplo: Raciocínio baseado em caso e método do k-vizinhos mais próximos.
- Conexionista, são construções matemáticas simplificadas inspiradas no modelo biológico do sistema nervoso. Exemplo: Redes Neurais.
- Evolutivo, modelo biológico de aprendizado. Exemplo: Analogia com a teoria de Darwin (seleção natural).

Uma das abordagens do aprendizado de máquina utiliza é o princípio da indução, que é sua forma de inferência lógica, com o intuito de obter conclusões genéricas a partir de um conjunto de exemplos. Para essa indução derivar conhecimento, deve haver uma quantidade suficiente e suas classes devem estar bem definidos, sendo assim quanto mais exemplos relevantes selecionados para treinar o indutor, mais bem classificado será o novo conjunto de dados (DUQUE, 2010). O aprendizado indutivo segundo Duque (2010) pode ser dividido em supervisionado e não supervisionado. Conforme mostra a figura.


Figura 1 – Hierarquia do aprendizado

Fonte: (MONARD; BARANAUSKAS, 2003 apud DUQUE, 2010, p. 37)

- Não-supervisionado: o indutor analisa os exemplos e tenta determinar se alguns deles podem ser agrupados de alguma maneira, formando agrupamentos ou *clusters*.
- Supervisionado: é fornecido ao algoritmo de aprendizado um conjunto de exemplos de treinamento para os quais o rótulo da classe associada é conhecido. Neste tipo de aprendizagem existe um "professor"que avalia a resposta da rede ao padrão atual de entrada. As alterações dos pesos são calculadas de forma a que a resposta da rede tenda a coincidir.

O aprendizado de máquina pode ser incorporado em ferramentas usadas para um propósito específico e, em outros casos, os usuários adaptaram aplicativos de aprendizado de máquinas de uso geral para suas necessidades ou desenvolvem suas próprias aplicações. Neste trabalho, utilizaremos o aprendizado de máquina em sua forma supervisionado. É o tipo de aprendizagem que é relativamente fácil avaliar o desempenho da rede para um determinado estado do nosso sistema e, consequentemente, implementar um "Tutor inteligente", sendo aquela que vamos usar.

### 2.1.1 SISTEMAS TUTORES INTELIGENTES

No início da década de 1970, os pesquisadores tinham como um de seus objetivos, adotar um tutor humano aplicando as técnicas de inteligência artificial tendo assim uma característica inteligente. Na época se acreditava que os sistemas conseguiam a metade do impacto que os tutores reais causam em grupos de alunos, nesse se alavancou os estudos nessa área (DAMASCENO, 2011). Com isso, se desencadeou uma evolução foi de uma forma muito primitiva, de instrução assistida por computador, variando várias formas de sistemas de e-learning, progredindo para formar sistemas adaptativos para os alunos, para os modernos sistemas tutores inteligentes.

O princípio fundamental de um programa tutor é adaptar sistemas computacionais com modelos de conteúdo instrucionais utilizados no auxílio de processo ensino-aprendizagem, capazes de fazer inferências e de simular decisões como se fossem os próprios professores orientadores especialistas (FADEL, 2005).

Neste tipo de programa, a interface deve ser capaz de oferecer diferentes tipos de ambiente de aprendizagem, ser adaptável, e permitir chegar facilmente ao conhecimento desejado. Para isso, segundo Ahuja e Sille (2013), um típico sistema tutor inteligente deve haver os seguintes quatro componentes básicos;

- **O modelo de domínio**: consiste nas regras, conceitos, fatos, e estratégias de resolução de problemas do domínio em contexto. Item necessário como fonte de conhecimento especializado, um padrão para avaliação do desempenho e diagnóstico do aluno de erros.

- **O modelo do estudante**: é uma sobreposição no modelo de domínio. enfatiza cognitivo e estados afetivos do aluno em relação ao seu evolução à medida que o processo de aprendizagem avança. Enquanto estudante trabalha passo-a-passo através do seu problema processo de solução, o sistema se engaja no modelo processo de rastreamento. Sempre que houver algum desvio o modelo predefinido, o sistema sinaliza como um erro.

- **O modelo de tutoria**: também chamado estratégia pedagógica, utiliza as informações do domínio e modelos de estudantes para regula as interações instrucionais com o aluno. Isto é intimamente ligada ao modelo de estudante, faz uso de conhecimento sobre o aluno e seu próprio objetivo tutorial estrutura, para conceber a atividade pedagógica a ser apresentado. Ele rastreia o progresso do aluno, constrói um perfil de pontos fortes e fracos em relação ao regras de produção.

- **O modelo de interface do usuário**: Integra todos os tipos de informações necessárias para interagir com o aluno, tendo o intuito facilidade de uso e apresentação, sendo elas de maneira gráfica, texto, multimídia, teclado, mouse, etc...

Há diversas possibilidades de aplicação dos STI no ensino-aprendizagem, que pode ir de aplicações operacionais simples, até mesmo a adoção de estratégias pedagógicas previamente desenhadas e definidas de acordo com metodologias de curso, que de acordo com Silva (2006) se destacam algumas delas:

- Incorporação de recursos para identificar os tópicos visitados com maior frequência pelo aluno, com intuito de propor materiais complementares sobre o tema.
- Adotar modelos comportamentalistas, onde o STI pode elaborar e propor avaliações aleatórias, com base no Modelo do Aluno, por meio da seleção de questões objetivas armazenadas em um banco de dados, com o objetivo de verificar o desempenho do aluno em determinados tópicos do conteúdo e, se necessário, indicar conteúdos que devem ser revisados, aprofundados etc...
- Incorporar agentes de monitoramento sobre os acessos dos alunos nos cursos e então emitir sinais de orientação ou alerta, que representem o nível de satisfação do tutor com a frequência do aluno no curso.
- A possibilidade de inclusão de outras tecnologias de agentes, desde agentes que realizam as análises de multimídia (texto, imagens, sons) até mesmo de levantamento de perfil utilizada de forma estratégicas para lidar com a diversidade do público.

## 2.2 PROCESSAMENTO DE LINGUAGEM NATURAL

O Processamento de Linguagem Natural é uma área da Inteligência Artificial (IA) utilizada para a compreensão e produção de linguagem natural, como por exemplo o português e o inglês. E trata computacionalmente os diversos aspectos comunicação humana, como sons, palavras, sentenças e discursos, considerando formatos e referências, estruturas e significados, contextos e usos (SILVA; LIMA, 2016).

Segundo Rabuske (1995 apud FARINON, 2015, p. 12),

> O entendimento da linguagem natural é difícil, pois requer conhecimentos de linguística e do domínio do discurso. Estes conhecimentos, em boa dose, o ser humano adquire naturalmente, desde tenra idade. À medida que a pessoa se desenvolve intelectualmente, acumula mais e mais conhecimentos e experiências que vão enriquecendo a possibilidade de comunicação.

Ao analisar a linguagem e seus significados, os sistemas PLN Segundo Oliveira (2013), possuem aplicações em diversas gamas de papeis e que consistem basicamente em dois segmentos:

- **Aplicações baseadas em texto**: são sistemas que procuram documentos específicos em uma base de dados como por exemplo Recuperação de informações, Tradução automática de texto entre linguagens, analise de sentimentos/opiniões e Alinhamento de textos.
- **Aplicações baseadas em diálogos**: por serem baseadas em diálogos, referem-se às interfaces de linguagem natural para bancos de dados, os sistemas tutores e os sistemas que interpretam e respondem a comandos expressados em linguagem escrita ou falada, por exemplo Chatbots e Assistentes virtuais inteligentes.

Historicamente, a ideia central do processamento de linguagem natural nasceu em 1950, quando Alan Turing próprios o jogo da imitação (TURING, 1950). Anos após, em 1954, a experiência de Georgetown-IBM começou a ser mostrado as possibilidades da tradução automática dentro do campo computacional, inicialmente com a tradução russo para inglês (NYE, 2016).

Alguns sistemas PLN começou a aparecer conforme o tempo, tais alguns está o SHRDLU, um programa para entender a linguagem natural que trabalhava em "blocks worlds", escrito por Terry Winograd no Laboratório de Inteligência Artificial do MIT na década de 1968-70 (WINOGRAD, 2014).

Outro famoso por sua época que utilizava o PLN foi a ELIZA criado por Joseph Weizenbaum no laboratório de Inteligência Artificial do MIT, entre os anos de 1964 e 1966. A ideia básica é simular a conversação entre homem e máquina, inclusive o próprio autor ficou surpreso com a quantidade de pessoas que atribuíram à Eliza características muito semelhantes aos sentimentos humanos (GRANATYR, 2016).

A partir de 1980, os algoritmos de aprendizado de máquina deram um novo rumo a PLN, pois dessa maneira seria possível deixar de lado os conjuntos de regras de manuscritos extensos e complexos e tornar a máquina apta a aprender. Pesquisas recentes têm se concentrado cada

vez mais em algoritmos de aprendizagem não-supervisionada e a aprendizagem supervisionada (ROZA, 2016).

Apesar dos avanços, os sistemas de linguagem natural que foram implantados para aplicativos do mundo real ainda não podem executar o raciocínio de senso comum ou utilizar o conhecimento do mundo de maneira geral e robusta. Podemos esperar que esses difíceis problemas de inteligência artificial sejam resolvidos futuramente, mas nesse meio tempo é necessário viver com algumas limitações severas nas capacidades de raciocínio e conhecimento dos sistemas de linguagem natural (BIRD; KLEIN; LOPER, 2009).

Assim, desde o início, um importante objetivo da pesquisa da PNL tem sido o progresso na difícil tarefa de construir tecnologias que "entendam a linguagem", usando técnicas superficiais, mas poderosas, em vez de capacidades irrestritas de conhecimento e raciocínio.

## 2.3 ASPECTOS DE ESTUDOS DA LINGUAGEM NATURAL

As pesquisas em PLN requerem um conhecimento da língua que será utilizada na comunicação entre o usuário e o sistema computacional no caso neste trabalho, a língua portuguesa. Para a interpretação de uma sentença na linguagem natural por um sistema computacional, a mesma passa por algumas etapas, para poder ser compreendida, ou seja, é necessário armazenar informações morfológicas, sintáticas e semânticas juntamente com as palavras que o sistema compreende (AGOSTI; BRANDAO, 2010). Este estudo está voltado em três aspectos da comunicação em língua natural.

- **Som**: fonética e fonologia
- **Estrutura**: análise sintática e morfológica
- **Significado**: análise semântica e pragmática

Em um sistema de dialogo por exemplo que utiliza PLN, pode ser encontrado diferentes tipos de conhecimento linguístico que segundo Bird, Klein e Loper (2009) os quais podem ser separados em cinco etapas de processamento conforme a imagem abaixo:

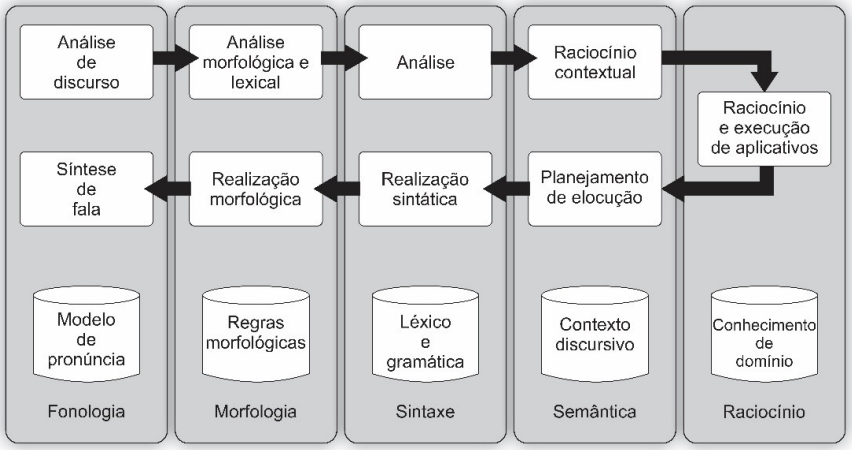


Figura 2 – Etapas Processamento de linguagem Natural

Fonte: Adaptado (BIRD; KLEIN; LOPER, 2009)

### 2.3.1 FONOLOGIA

A primeira etapa Fonologia, é onde a E/S do texto é realizada. Na Fonologia, o fonema é uma unidade acústica que não é dotada de significado. Isso significa que os fonemas são os diferentes sons que produzimos para exprimir nossas ideias, sentimentos e emoções a partir da junção de unidades distintas. Essas unidades, juntas, formam as sílabas e as palavras (SANTOS, 2006).

O formato das E/S fica a critério de cada desenvolvimento, no caso desse projeto, as interações serão feitas por voz e fala, com isso, as conversões de Fala-para-Texto e de Texto-para-fala devem ser realizadas nessa etapa (FILHO, 2009). Com isso outro estudo é importante nessa etapa, a fonética. Cabe a ela descrever os sons da linguagem e analisar suas particularidades acústicas e perceptivas. Ela fundamenta-se em estudar os sons da voz humana, examinando suas propriedades físicas independentemente do seu "papel linguístico de construir as formas da língua". Sua unidade mínima de estudo é o som da fala, ou seja, o fone (SANTOS, 2006).

A Fonética e a Fonologia são duas disciplinas separadas mais ao mesmo tempo são interdependentes, uma vez que, para qualquer estudo de natureza fonológica, é imprescindível partir do conteúdo fonético, articulatório e/ou acústico, para determinar as unidades distintivas de cada língua. Desta forma, a Fonética e a Fonologia não são divididas em dois termos, pois a Fonética trata da substância da expressão, enquanto a Fonologia trata da forma da expressão, constituindo, as duas ciências, dentro de um mesmo plano de expressão (SANTOS, 2006).

### 2.3.2 MORFOLOGIA

A análise morfológica podemos considerar a primeira etapa pela qual a sentença é analisada, pois ela identifica palavras ou expressões isoladas em uma sentença, sendo este processo auxiliado por delimitadores como pontuações e espaços em branco (AGOSTI; BRANDAO, 2010). As palavras identificadas são classificadas de acordo com seu tipo de uso ou em categoria gramatical.

Quando nos referimos às categorias gramaticais, nos referimos às dez classes gramaticais, as quais representam a base de uma boa parte do sistema linguístico que são: substantivos, artigos, pronomes, verbos, adjetivos, conjunções, interjeições, preposições, advérbios e numerais (DUARTE, 2009). Esse passo é fundamental para a compreensão da frase, pois, para extrair o significado da frase, é necessário entender todos os vocábulos componentes (CURADO, 2010).

A análise morfológica é realizada em duas etapas, a tokenização e a análise léxica:

- **Tokenização**: É a quebra a sequência de caracteres em um texto localizando o limite de onde uma palavra termina e outra começa. Os delimitadores como já foram mencionados, podem ser por pontuações tais como ponto final, vírgulas, aspas, hífens ou espaço em branco. Dentro do PLN, as palavras identificadas são chamadas de tokens.

- **Análise léxica**: Segundo SCAPINI (1995 apud GONZALEZ; LIMA, 2003, p. 07) termo "léxico" significa uma relação de palavras com suas categorias gramaticais e seus significados. Em relação a uma determinada língua, um léxico é o universo de todos os seus itens lexicais, que seus falantes utilizam, já utilizaram ou poderão vir a utilizar.

Em alguns casos, utiliza-se o termo "léxico" para identificar o componente de um sistema de PLN com informações semânticas e gramaticais sobre itens lexicais. A tarefa básica da análise léxica é relacionar variantes morfológicas aos seus lemmas, que nada mais são do que as formas canônicas das palavras, ou a forma em que as palavras se encontram no dicionário (GONZALEZ; LIMA, 2003). Para exemplificar, em uma oração "A menina é bonita", aplicando a análise morfológica podemos separar e classificar a oração da seguinte maneira: "A" = artigo definido singular feminino, "menina" = substantivo, "é" = verbo, "bonita" = adjetivo.

Para esse procedimento, se torna necessário de uma "base de dados Lexical", ou "dicionário" que provem uma grande coleção de informações lexicais sobre as palavras. Em sistemas de PLN, essas informações lexicais devem ser apresentadas de um formato estruturado e acessível (GONZALEZ; LIMA, 2003).

### 2.3.3 SINTAXE

A análise sintática (*parsing*) é a fase do procedimento que avalia os vários modos de como combinar regras gramaticais e as informações do analisador morfológico, com a finalidade transformar um texto de entrada em uma estrutura de dados, em geral em uma árvore, o que é conveniente para processamento posterior e captura a hierarquia implícita desta entrada (OLIVEIRA, 2013).

A análise sintática de uma oração em português segundo Oliveira (2013), deve levar em conta os seguintes sintagmas: termos essenciais (sujeito e predicado), termos integrantes (complementos verbal e nominal), termos acessórios (adjunto adverbial, adjunto adnominal e aposto), análise do período (simples ou composto), sua composição (por subordinação, por coordenação) e a classificação das orações (absoluta, principal, coordenada ou subordinada). A figura abaixo ilustra a árvore de derivação criada como resultado da análise sintática da frase “A menina é bonita”.

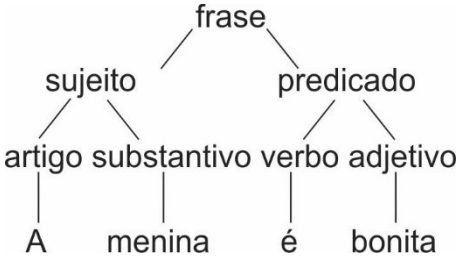


Figura 3 – Árvore de derivação da análise sintática

Fonte: Adaptado (OLIVEIRA, 2013)

Numa árvore sintática, como no exemplo acima, as folhas são sempre representadas pelas as palavras pertencentes oração enquanto os demais nós são sempre representados pelos nomes das unidades componentes da oração.

Tanto a análise morfológica quanto a análise sintática fazem parte do processamento morfossintático. Enquanto o analisador morfológico lida com a estrutura das palavras e com a classificação das mesmas em diferentes categorias, o analisador sintático trabalha em nível de agrupamento de palavras, analisando a constituição das frases (GONZALEZ; LIMA, 2003).

### 2.3.4 SEMÂNTICA

A semântica está relacionada ao significado, não só de cada palavra, mas também do conjunto resultante delas. O processamento semântico é considerado um dos maiores desafios do PLN, pois se vincula, de um lado, com a morfologia e a estrutura sintática e, de outro lado em alguns casos, com informações da pragmática (GONZALEZ; LIMA, 2003). Com isso, o analisador semântico é a terceira etapa do PLN, pois ela analisa os sentidos das estruturas das palavras que foram reagrupadas pelo analisador sintático e que por sua vez, o analisador morfológico permitiu identificar estas palavras individualmente.

Segundo Oliveira (2013), pode-se dizer que a semântica se desdobra em semântica léxica e em semântica gramatical. A semântica léxica busca uma representação conceitual para descrever o sentido, sendo que, para construir esta representação, pode ser feita a decomposição semântica das unidades léxicas (em primitivas ou em traços semânticos), ou podem ser utilizadas redes semânticas. Esta última forma de representação foi originada da psicologia e leva em conta a forma como os seres humanos categorizam e memorizam conceitos.

É importante compreender bem acerca dos significados e para que possamos entender de forma adequada e conveniente as palavras que melhor expressam a ideia principal da oração que está sendo analisada. Segundo Duarte (2008), dentro das atribuições da semântica, está presente o estudo da:

Sinonímia: relação de sentido entre dois vocábulos que têm significação muito próxima.

Ex. Bondoso = Caridoso

Antonímia: relação de sentido que opõe dois termos (prefixos, palavras, locuções, frases) contrários, seja numa gradação.

Ex. Bondoso x Maldoso.

Paronímia: semelhança entre palavras, quer por motivos etimológicos, quer por convergência fonética parcial.

Ex. Emigrar x Imigrar.

Homonímia: relação entre formas linguísticas que, com significados diferentes, têm a mesma forma gráfica e fônica ou apenas fônica.

Ex. Acento x Assento.

Polissemia: multiplicidade de sentidos de uma palavra ou locução. Exemplos;

"Hidrate bem suas **mãos**."(parte constitutiva do corpo humano)

"Não abra **mão** de seus direitos."(abdicar de algo, desistir)

Conotação: é o sentido figurado, ou seja, aquele cujas palavras, expressões ou enunciados ganham um novo significado.

Ex. "Os poemas são **pássaros** que chegam não se sabe de onde e pousam no livro que lês. (sentido figurado, subjetivo)".

Denotação: é o sentido literal, ou seja, o sentido que carrega o significado básico das palavras, expressões e enunciados de uma língua.

Ex. "Admiro o voo dos **pássaros**. (sentido literal, substantivo)".

A pragmática por outo lado trata-se do ramo da linguística que analisa o uso concreto da linguagem pelos falantes da língua em seus variados contextos. Ela extrapola a significação dada às palavras pela semântica e pela sintaxe, observando o contexto extralinguístico em que estão inscritas. As palavras podem se associar através de dois tipos de relações: paradigmáticas e sintagmáticas. As relações paradigmáticas associam palavras através do significado, como “mergulhar” e “água”. As relações sintagmáticas conectam palavras que são frequentemente encontradas no mesmo discurso, como “água” e “poça” (NETO; TONIN; PRIETCH, 2010).

Segundo Gonzalez e Lima (2003), um dos maiores problemas do processamento semântico é o processo composicional. Assim, como o significado de um constituinte de uma sentença depende dos significados dos seus sub-constituintes, os significados destes podem, por sua vez, ser determinados por regras gramaticais.

Alguns problemas de busca de significado podem ocorrer no processo, o qual, os mais comuns são:

Ambiguidade: palavras que podem ter mais de um sentido ou significado. Que segundo Savadovsky (1988 apud AGOSTI, 2003, p. 33) é um dos problemas mais difíceis de tratar computacionalmente. Exemplos;

"João toca **bateria**"("bateria"nesse caso se refere a um instrumento de percussão)

"O meu celular acabou a **bateria**"("bateria"nesse caso se refere a fonte de energia elétrica)

Anáforas: que possuem repetição da mesma palavra sucessivamente. Exemplo:

**Parto** do princípio do corpo como metáfora da vida.

**Parto** de uma nova vida que ressignifica outro corpo.

**Parto** de uma mãe: luz a uma criança.

**Parto** para outros destinos,

Rumos da maternidade.

Paragens infinitas.

(Letícia Gomes Montenegro)

Elipses: fenômenos pragmáticos textuais que é a ocultação de termos em uma oração. Exemplos;

“Ele gosta de música, eu de viagens.” (omissão do verbo “gostar” no segundo período da oração)

“Na minha mesa, papeis e livros.” (o verbo “haver” está oculto nesta sentença, caso contrário a frase seria: “na minha mesa há papéis e livros”)

É de suma importância que esses problemas sejam evitados, pois os mesmos podem comprometer o desempenho e o resultado de todas as etapas do PLN. isso pode ser evitado por meio de métodos de tratamento, como utilizar cálculo dos significados implícitos, cálculo das relações, análises contextuais, dentre outros (OLIVEIRA, 2013).

Apesar dos avanços, os sistemas de linguagem natural que foram implantados para aplicativos do mundo real ainda não podem executar o raciocínio de senso comum ou utilizar o conhecimento do mundo de maneira geral e robusta. Podemos esperar que esses difíceis problemas de inteligência artificial sejam resolvidos futuramente, mas nesse meio tempo é necessário viver com algumas limitações severas nas capacidades de raciocínio e conhecimento dos sistemas de linguagem natural (BIRD; KLEIN; LOPER, 2009).

Assim, desde o início, um importante objetivo da pesquisa da PNL tem sido o progresso na difícil tarefa de construir tecnologias que "entendam a linguagem", usando técnicas superficiais, mas poderosas, em vez de capacidades irrestritas de conhecimento e raciocínio.

## 2.4 LINGUÍSTICA DE CÓRPUS

Trabalhos práticos em Processamento de Linguagem Natural geralmente utilizam grandes corpos de dados linguísticos, chamados de Córpus (BIRD; KLEIN; LOPER, 2009). Muitos Córpus são projetados para conter uma quantidade cuidadosamente balanceada de material de um ou mais gêneros, o que propicia uma melhor análise estatística e teste de hipótese.

Para Russell e Norvig (2013 apud FARINON, 2015, p. 13) a definição de córpus é,

> Um córpus é uma grande coleção de texto, como os bilhões de páginas que constituem a World Wide Web. O texto é escrito por seres humanos e para seres humanos, e a tarefa do software é facilitar a localização das informações corretas pelos seres humanos. Essa abordagem implica o uso de estatística e aprendizagem para tirar proveito do corpus, e em geral impõe modelos probabilísticos de linguagem que podem ser aprendidos a partir de dados.

Segundo Biber, Reppen e Conrad (1994 apud OLIVEIRA, 2013, p. 59) há duas vantagens da utilização de córpus textual que são:

> 1. They provide a large empirical database of natural discourse, so that analyses are based on naturally-occurring structures and patterns of use rather than intuitions and perceptions, which often do not accurately represent actual use. 2. They enable analyses of a scope not feasible otherwise, allowing researchers to address issues that were previously intractable. This is particularly true of computerbased text corpora, which can be analyzed using (semi-)automatic techniques. Such analyses can examine much more language data than otherwise possible, including more texts, longer texts, a wider range of variation (texts from different language varieties), a wider range of linguistic characteristics, and the systematic cooccurrence patterns among linguistic features.

Em resumo, um córpus é uma grande coleção de texto, que podemos pegar como por exemplo as inúmeras páginas de Web, que forma a internet que podem ser utilizadas como bases de dados e para o PLN.

No Brasil, existem alguns projetos que contribui com o córpus da língua portuguesa, um deles é o projeto Processamento Computacional do Português que foi lançado em 1998, como uma primeira medida para organizar a área da engenharia da linguagem do português. Considerada pelo Ministério da Ciência e da Tecnologia (MCT), Esse projeto resultou na criação de corpora de grandes dimensões, que são livremente disponíveis e distribuíveis para efeitos de utilização em investigação e desenvolvimento de sistemas que processem a língua portuguesa. Todas as informações e consultas desse projeto podem ser realizados por meio do endereço: https://www.linguateca.pt/.

Outro projeto é o Léxico do Português Brasileiro – LexPorBR, que foi lançado em 2012 tendo como objetivo oferecer um corpus psicolinguístico do português brasileiro que disponibilize o máximo de informações metalinguísticas e psicolinguísticas sobre as palavras. Além de possuir um corpus livre e aberto, esse projeto tem um serviço na Internet que permite a consulta on-line a corpora do português por meio do endereço: www.lexicodoportugues.com/stat_ling.php.

# 3 NLTK

Para buscar uma interação mais intuitiva, deixaremos de lado interação por textos, imagens, gestos ou cliques e utilizaremos comandos de voz e respostas por conversa. Essa maneira pode parecer limitado, mais a conversa é a forma que utilizamos o tempo todo para se comunicar e interagir.

Através do trabalho realizado por Andrade Rafael Couto Barros (2016) tendo como metodologia a pesquisa qualitativa de 14 ferramentas que auxiliam o desenvolvimento de softwares para PLN, foram apresentados as principais características, pontos fortes e deficiências dessas ferramentas, o qual se obteve a seguinte tabela de comparação:

Figura 4 – Comparativa entre as Ferramentas de PLN

| | TABELA COMPARATIVA | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Ferramenta / Característica | Open Source | Manual da Ferramenta | Fórum | Suporte Português | Plataforma | Análise Semântica | Análise Sintática | Análise Morfológico |
| OpenNLP | X | X | | | JAVA | X | X | X |
| UIMA | X | X | | X | JAVA | X | X | X |
| NLTK | X | X | X | X | PYTHON | X | X | |
| GATE | X | | X | | C++ | X | X | X |
| Lucene | X | X | | | JAVA | X | X | X |
| Stanford CoreNLP | X | | | | JAVA | X | X | X |
| CoGrOO | X | | | X | C++ | | X | |
| LX-Center | X | X | | | ------ | X | X | X |
| COGCOMP | X | X | | | JAVA | X | X | X |
| Freeling | X | X | | | C++ | | | X |
| WordNET | X | X | | | ------- | | | |
| LingPipe | | X | | X | JAVA | | | |
| ARK Syntactic & SemanticParsing | X | | | | ------- | X | X | X |
| FrameNET | X | X | | | ------- | | | |

Fonte: Adaptado (ANDRADE RAFAEL COUTO BARROS, 2016)

Para o desenvolvimento desse trabalho, é necessário que a ferramenta que iremos utilizar deva atender os seguintes critérios;

- OpenSource
- Manual de ferramentas
- Suporte ao português

Dentre elas, uma ferramenta se destacou por possuir essas características, foi a Natural Language Toolkit ou NLTK, uma plataforma para criação de programas de linguagem natural escrito na linguagem de programação Python. um software de código aberto que foi originalmente criado em 2001 disponível para Windows, Mac OS X e Linux, fornece interfaces fáceis de usar para mais de 50 recursos corpora e lexicais como o WordNet, juntamente com um conjunto de bibliotecas de processamento de texto para classificação, tokenização, stemming, tagging, análise, raciocínio semântico e suporte para reconhecimento da escrita na língua portuguesa (BIRD; KLEIN; LOPER, 2009).

O NLTK disponibiliza para a comunidade um fórum ativo para discusão entre usuários e desenvolvedores e acervo de manuais e guias práticos existentes no site oficial (BIRD; KLEIN; LOPER, 2009). Fora essas qualidades do NLTK que levou a sua escolha como ferramenta deste trabalho, a pesquisa de comparação realizado por Andrade Rafael Couto Barros (2016), o NLTK foi uma das ferramentas que mais se destacou dentre outras ferramentas de PLN, pois esta ferramenta é uma das poucas que possui suporte para reconhecimento da língua portuguesa o que é extremamente importante a esse trabalho, e contém uma linguagem de programação de fácil aprendizado, no caso o Python.

## 3.1 ESTRUTURA

O NLTK é dotado de uma vasta documentação. o site http://nltk.org/ fornece documentação da API para todos os módulos, classes e funções no toolkit, especificando os parâmetros e apresentando exemplos de uso. Abaixo será apresentado uma lista dos módulos mais importantes do NLTK.

- nltk.corpus: Os módulos neste pacote fornecem funções que podem ser usadas para ler arquivos corpus em vários formatos. Essas funções podem ser usadas para ler os arquivos

corpus que são distribuídos no pacote corpus NLTK e os arquivos corpus que fazem parte dos corpora externos.

- nltk.tokenize: Os tokenizadores dividem as strings em listas de substrings. Por exemplo, tokenizers podem ser usados para encontrar palavras e pontuação em uma string. O NLTK também fornece um tokenizador baseado em expressão regular mais simples, que divide o texto em espaço em branco e pontuação.

- nltk.collocations: Modulo para identificar colocações, ou seja, palavras que aparecem frequentemente de forma consecutiva dentro de corpora. Elas também podem ser usadas para encontrar outras associações entre ocorrências de palavras. Encontrar colocações requer primeiro calcular as frequências das palavras e sua aparência no contexto de outras palavras.

- nltk.tag: Esse modulo é utilizado para marcar cada token em uma frase com informações suplementares, como sua *part-of-speech*. O processo de classificar palavras em suas part-of-speech e rotulá-las de acordo é conhecido como part-of-speech tagging, POS-tagging ou simplesmente tagging. Partes do discurso também são conhecidas como classes de palavras ou categorias lexicais.

- nltk.cluster: O nltk.cluster é um modulo que tem a tarefa de descobrir grupos de itens semelhantes com uma coleção grande. Também é descrito como aprendizado de máquina não supervisionado, pois os dados a partir dos quais ele aprende não são anotados com informações de classe, como é o caso da aprendizagem supervisionada.

- RemoveStopWords: Dentro do pré-processamento da linguagem natural é muito importante a identificação e a remoção das chamadas stopwords do idioma. As Stopwords são as palavras que não adicionam muito significado a uma sentença. Eles podem ser ignorados com segurança sem sacrificar o seu significado original. Para utilizarmos essa função, as Stopwords ( “o”, “ele”, “tem”, “para”, “você”, etc.) devem estar armazenadas em uma lista Corpus. O NLTK já disponibiliza mais de 20 listas de palavras em diversos idiomas inclusive o português para a sua utilização, além de claro, o modulo stopwords.words para realizar essa função de tratamento

- nltk.classify: modulo que atribua um vetor a um cluster. Em tarefas de classificação básica, cada entrada é considerada isoladamente de todas as outras entradas e o conjunto de rótulos

é definido antecipadamente. Um classificador é chamado supervisionado se for construído com base na corpora de treinamento que contém o rótulo correto para cada entrada. Alguns exemplos de tarefas de classificação são:

- Decidir se um email é spam ou não.
- Identificação de Gênero.
- Decidir qual é o tópico de um artigo de notícias, como "esportes", "tecnologia"e "política".

Para decidir se um modelo de classificação está capturando com precisão um padrão, devemos avaliar o modelo. O resultado da avaliação é importante para decidir quão confiável é o modelo e para quais objetivos podemos usá-lo. A avaliação também pode ser uma ferramenta eficaz para nos orientar na realização de melhorias futuras no modelo.

# 4 DESENVOLVIMENTO

## 4.1 SISTEMA DE PERGUNTAS E RESPOSTAS

Em geral, os sistemas de perguntas e respostas analisa uma questão de entrada no primeiro passo, analisa a intenção do usuário, e retorna uma ou mais respostas baseado na pergunta. Isto é importante para entendermos o que um usuário necessita se é o nome da pessoa, localização, organização, ou quaisquer outros tipos (LEE *et al.*, 2001). Esse tipo de sistema ajuda a apoiar profissionais em situações críticas e oportunas de tomadas de decisão em áreas como saúde, marketing, negócios, sistemas de segurança e comerciais, e a descoberta de conhecimento (FERRUCCI *et al.*, 2010).

Este projeto utiliza o sistema de perguntas e respostas em sua estrutura, onde o seu funcionamento consiste em três passos importantes; processamento da pergunta, seleção de resposta e processamento da resposta, que são explicado a seguir.

- Processamento de Perguntas - Em geral, um sistema de resposta a perguntas analisa uma questão de entrada no primeiro passo. É importante entender o que um usuário precisa; seja o nome, definição ou qualquer outro tipo de questionamento. Para isso, primeiro classificamos os tipos de respostas possíveis e usamos os Padrões Léxico-Semânticos para determinar o tipo de resposta de uma pergunta.

- Seleção de Resposta - Nós construímos um banco de dados de índice a partir da coleção de informações. Ao analisar as perguntas anteriores e suas respostas, podemos supor que as respostas a uma pergunta geralmente ocorrem comparativamente às palavras-chave correspondentes em um índice do banco de dados. Isso significa que a resposta pode ocorrer em qualquer índice classificado e é melhor selecionarmos as passagens de cada informação e classificá-las por similaridade de classe. Então, podemos usar as palavras-chave para encontrar as respostas do usuário.

- Processamento de Resposta - O processamento de respostas seleciona os candidatos que correspondem ao tipo de resposta de cada passagem e classifica-os. O processamento das respostas é realizado baseando nos pesos que os índices possuem e a sua similaridade com a palavra-chave. Após essa ação e como uma pergunta pode ter n respostas, a resposta é selecionada de forma randômica e apresentada ao usuário.

Ferrucci *et al.* (2010) afirma que o domínio de Sistemas de Resposta a Perguntas consiste em um domínio atrativo e um dos mais desafiadores para a área de Computação e Inteligência Artificial, pois requer a síntese de recuperação de informações, processamento de linguagem natural, representação de conhecimento, aprendizado de máquina e interfaces humano-computador.

## 4.2 BIBLIOTECAS

Esta seção fornece informações específicas acerca de algumas bibliotecas e ferramentas importantes para o desenvolvimento do projeto.

### 4.2.1 DJANGO

No desenvolvimento deste trabalho, foi necessária a utilização de um framework para realizar operações de recebimento, exibição e manipulação da informação. Após um estudo dos melhores frameworks gratuitos do mercado, os que se destacaram foram o Pyramid pela empresa Pylons Project, web2py desenvolvido por Massimo Di Pierro, Flask desenvolvido por Armin Rahner; e DjangoFramework com a empresa Django Software Foundation.

Das opções citadas acima, a escolha foi o Django um framework web, grátis e com código aberto, a sua escolha se deu por utilizar tecnologias ágeis com várias funcionalidades implementadas como padrão, sistema de configuração avançado e um sistema de testes e por adotar a tecnologia MVT (model,view, template), que é uma boa prática de arquitetura de software (ABREU, 2016).

Essa arquitetura representa as principais áreas do Django onde o model, estão os modelos de dados, vistos no banco de dados; view, onde está presente os códigos de controle sobre as operações de criar, listar e apagar questões, responder questões, avaliar resposta, controle de acesso por períodos e distribuição de questões; Templates que é as páginas html que são processadas por um template engine que realiza algumas operações como processar uma lista de dados em python convertendo para uma lista em html (MULLER, 2017).

Além de todas essas características positivas ele é desenvolvido modularmente. Sendo assim, um módulo não interfere no outro, com futuras implementações nos sistemas livre de incompatibilidades com as atuais existentes. A utilização do framework Django facilita o desenvolvimento e simplifica a criação de aplicações web. Sua manutenção, é de fácil alteração por ter a doutrina de não repetir código em diversos arquivos. A manutenção em um bloco de código será facilmente dissipada a outros arquivos do sistema.

### 4.2.2 gTTS

Google *Text-to-Speech* ou gTTS, e uma biblioteca Python de liçenca gratuita para interagir com a API de conversão de texto em fala do Google Tradutor. A fermenta grava dados .mp3 falados em um arquivo, um objeto semelhante a um arquivo *bytestring* para manipulação de áudio adicional ou *stdout*. A biblioteca também possui pré-processamento que podem, por exemplo, fornece correções de pronúncia e *tokenizing* flexíveis que permite a leitura ilimitada de textos, mantendo a entoação, abreviações, decimais e muito mais, bem como recuperação automática de idiomas suportados (DURETTE, 2014).

Atualmente (versão 2.0.3), o Google *Text-to-Speech* incluí o suporte para mais de 40 idiomas (inclusive o português), que podem ser facilmente listados utilizando o comando "gtts-cli –all".

A API de conversão de texto em voz do Google Tradutor aceita no máximo 100 caracteres. Se, após a tokenização, qualquer um dos tokens for maior que 100 caracteres, ele será dividido em dois onde no último caractere de espaço mais próximo, mas antes do 100º caractere ou entre o 100º e o 101º caracteres, se não houver espaço (DURETTE, 2014).

#### 4.2.2.1 Pré-processamento e tokenizing

O gTTS possui um módulo chamado gtts.tokenizer que alimenta os recursos padrão de pré-processamento e tokenizing além de fornecer ferramentas para expandi-los facilmente. A seguir é apresentado os detalhes que cada etapa realiza:

- Pré-processador: Função que pega texto e retorna texto. Seu objetivo é modificar o texto (por exemplo, corrigindo a pronúncia) e/ou preparar o texto para uma tokenização adequada (por exemplo, garantindo o espaçamento após certos caracteres).

- tokenizing: função que recebe texto e retorna-lo em uma lista de tokens *strings*. No contexto do gTTS, seu objetivo é cortar o texto em segmentos menores que não excedam o tamanho máximo de caracteres permitido para cada solicitação, ao mesmo tempo em que torna o som da fala natural e contínuo. Ele faz isso dividindo o texto onde a fala pausaria naturalmente (por exemplo, em "."ou ",") durante o manuseio onde não deveria (por exemplo, em "10.5"ou "U.S.A."). Essas regras são chamadas de casos de tokenizing, aos quais é necessária uma lista.

- Caso tokenizing: Função que retorna um objeto regex que descreve o que procurar para um caso específico.

### 4.2.3 Speech Recognition

O *Speech Recognition* é uma Biblioteca para realização de reconhecimento de fala, para Python. Ela atua como um wrapper ou seja, ela atua como uma "ponte"de ligação da função de reconhecimento de fala do sistema para as várias APIs de fala populares, com isso, ela torna o processo extremamente flexível e prático (SPEECHRECOGNITION, 2017).

Cada instância Recognizer tem sete métodos para reconhecer fala de uma fonte de áudio usando várias APIs. Cada instância vem com uma variedade de configurações e funcionalidades para reconhecer a fala de uma fonte de áudio. Esses são:

- recognize_bing(): Utiliza API Microsoft Bing Speech.
- recognize_google(): Utiliza API Google Web Speech, suporta uma chave de API padrão que é codificada na biblioteca SpeechRecognition, contudo essa chave está limitado a apenas 50 solicitações por dia e não há como aumentar essa cota.
- recognize_google-cloud(): Utiliza API Google Cloud Speech - requer a instalação do pacote google-cloud-speech.
- recognize_houndify(): Utiliza API API Houndify da SoundHound Inc.
- recognize_ibm(): Utiliza API IBM Speech to Text pertencente a plataforma de serviços cognitivos o IBM Watson.
- recognize_sphinx(): CMU Sphinx - requer a instalação do PocketSphinx - biblioteca reconhecedora escrita em C. único dos sete que funciona offline.
- recognize_wit(): Utiliza a api Wit.ai.

Para realizarmos uma conversão de fala-para-texto inicialmente devemos criar um arquivo de áudio, o Speech Recognition suporta os formatos de arquivo de áudio WAV, FLAC, AIFF-C e AIFF, com o arquivo de áudio pronto, basta chamar uma das sete instâncias que desejar. isso torna o sistema flexível, pois caso os limites de cota de uma instância ocorrerem, ou servidor não estiver disponível, o tratamento desse problema é mais simples e rápido (AMOS, 2018).

## 4.3 TREINAMENTO

Para realizarmos o treinamento, inicialmente devemos estipular as áreas de conhecimento que o sistema terá, que nesse caso será os conhecimentos técnicos e conhecimentos extras. A área de conhecimento técnicos, contém as informações da base de ensino em si, que no caso é o conhecimento sobre Arquitetura de Computadores onde toda essa base de informações foi extraída da internet. Já na área de conhecimentos extras, contém as informações mais relacionadas a saudação, despedida, piadas, hesitação, ou seja, interações que "humaniza" o sistema e o torna mais interativo.

Uma vez que determinamos um conteúdo a ser treinado, podemos iniciar a inserção das informações no sistema. Para isso, foi montado uma interface que facilita esse treinamento (conforme a figura 5). Nela, contém campos de preenchimento onde cada uma delas tem uma função especifica no treinamento:


Figura 5 – Interface de Treinamento Bel

Fonte: Do autor

- Classificação: a classificação é a parte que engloba o tópico/classe do assunto em questão. ela por si só tem o objetivo de ligar a classe da pergunta com suas objetivas respostas. como a imagem abaixo, a classificação de um assunto deve seguir em um esquema de camadas com seus respectivos nós, que vão desde o tópico geral até suas camadas mais internas. Para uma melhor separação dos assuntos, a descrição da classificação deve conter a identificação de cada nó que iremos treinar; exemplos: ‘memoriasRamFunção’ classe que irá ser inseridas as informações sobre a função da memória Ram, ‘placamaeSoqueteModelos’ classe que irá ser inseridas as informações sobre os modelos de soquete da placa-mãe, ‘processadoresArquitetura’ classe que irá ser inseridas as informações sobre a arquitetura de Processadores, e assim por diante.

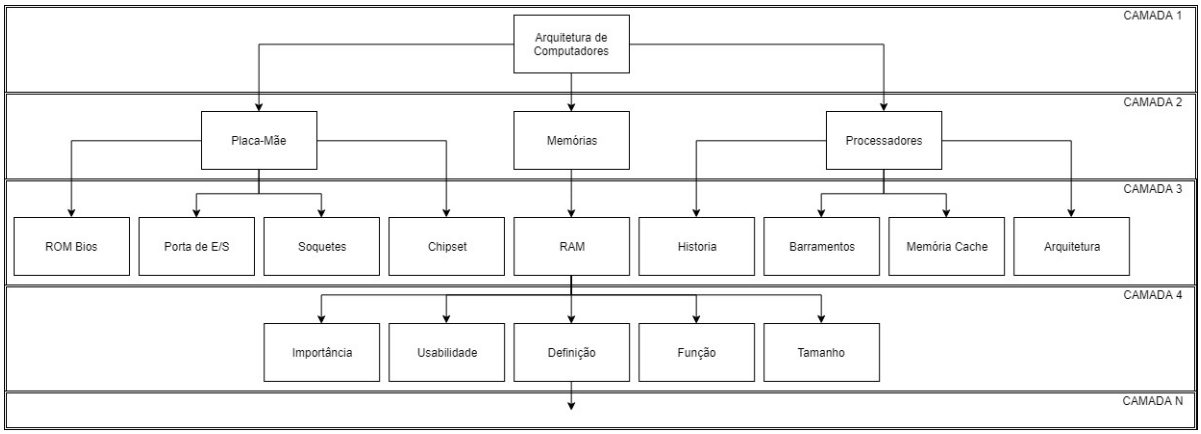


Figura 6 – Camadas de Treinamento

Fonte: Do autor

- Palavras Chaves: Nesse campo são inseridas todas as possíveis perguntas que um usuário pode fazer ao sistema que são relacionado a classe; exemplos: O que é processador?, Qual a importância da Memoria Ram do Computador?, Como funciona o chipset?. Essas perguntas são utilizadas como chave/identificador das ideias ou os temas para as referências de pesquisa.

- Previa de Resposta: é um campo de preenchimento opcional, mas pode ser utilizado para introduzir uma hesitação ou outra fala antes da resposta em si com um ’hmmm’, ’deixa eu lembrar...’ ou até mesmo ’Ótima pergunta!’ o que pode dar um tom de naturalidade na resposta.

- Resposta da Classificação: é um campo de preenchimento obrigatório pois ela é a resposta de retorno da classificação.
- Conclusão da Resposta: tem o mesmo funcionamento da Previa de resposta, contudo, utilizado após da entrega da resposta.

Quando é necessário editar, corrigir ou até mesmo excluir uma classe já registrada, a ferramenta tem à disposição uma interface que permite listar e buscar os conteúdos já gravados.

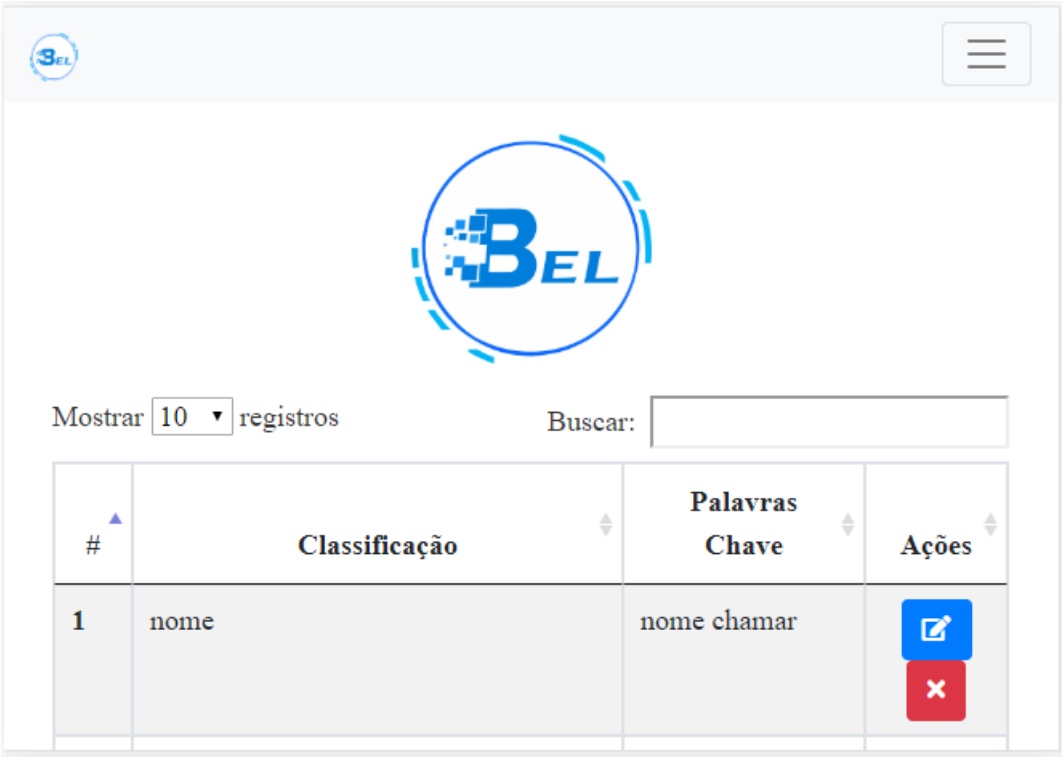


Figura 7 – Tela de listagem e busca

Fonte: Do autor

Todas as informações são inseridas dentro de um banco de dados que nesse caso, utilizamos o PostgreSQL. Conforme o diagrama a seguir, quatro tabelas foram necessárias para o agrupamento das informações; 'base_previus' que armazena as informações do campo prévia de resposta, "base_response"armazena as informações do campo resposta da classificação, 'base_next' que armazena as informações do campo conclusão de resposta, e a ultima 'base_classification' a tabela principal que armazena as informações do campo classificação. Todas elas tem como o termo comum a variável 'classification' que realiza a ligação da tabela principal 'base_classification' a todas as outras.

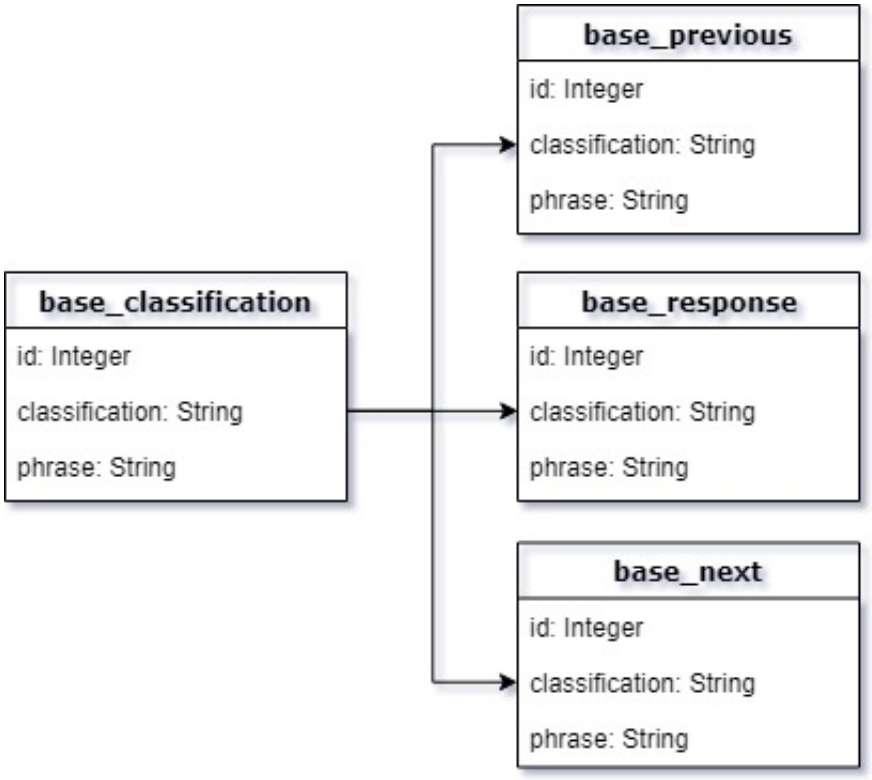


Figura 8 – Estrutura do Banco de dados

Fonte: Do autor

Uma vez que as informações estão presentes no banco de dados, elas são carregadas nas variáveis correspondentes para que possam ser manipuladas. Com isso, a função Learning fica responsável por realizar o treinamento no sistema:

```
1
2  base_class = models.Base_Classification.objects.all()
3  base_previous = models.Base_Previous.objects.all()
4  base_response = models.Base_Response.objects.all()
5  base_next = models.Base_Next.objects.all()
6
7  def Learning(base):
8      corpus_words = {}
9      for data in base:
10         phrase = data.phrase
11         phrase = Tokenize(phrase)
12         phrase = Stemming(phrase)
13         phrase = RemoveStopWords(phrase)
```

```
14          class_name = data.classification
15          if class_name not in list(corpus_words.keys()):
16              corpus_words[class_name] = {}
17          for word in phrase:
18              if word not in list(corpus_words[class_name].keys()):
19                  corpus_words[class_name][word] = 1
20              else:
21                  corpus_words[class_name][word] += 1
22      return corpus_words
```

O parâmetro base_class é o array que contém as informações do banco de dados para treinamento. Inicializamos um vetor auxiliar denominado corpus_words que será responsável por armazenar as classes, para suas respectivas palavras e peso. A função percorre por os elementos dentro do array base_class, realiza os tratamentos de Tokenização, Stemming e Stopwords da frase. A variável class_name armazena o valor correspondente ao nome da classe, a mesma é adicionada ao vetor corpus_words caso ainda não exista. No ultimo passo é feito uma verificação que realiza as pontuações nas palavras, caso a palavra ainda não seja conhecida em no vetor corpus_words, inicializa com o valor 1, e caso ela já seja conhecida adicionamos um valor de +1 na palavra. O diagrama abaixo ilustra o passo-a-passo dessa função;

Essas pontuações que as palavras recebem são utilizadas para localizar a classe de resposta que mais se identifica com a pergunta que foi realizada. Para isso, a função CalculateScore() fica responsável para realizar os cálculos das pontuações e o retorno do resultado:

```
1  def CalculateScore(sentence):
2      high_score = 0
3      class_name = 'default'
4      for classe in dados.keys():
5          pontos = 0
6          pontos = CalculateClassScore(sentence,classe)
7          if pontos > high_score:
8              high_score = pontos
9              class_name = classe
10     return class_name,high_score
```

Na função seu único parâmetro de entrada é a frase (sentence) que será analisada. Nela, se inicializa duas variáveis, a high_score e a class_name, que serão utilizadas para comparar

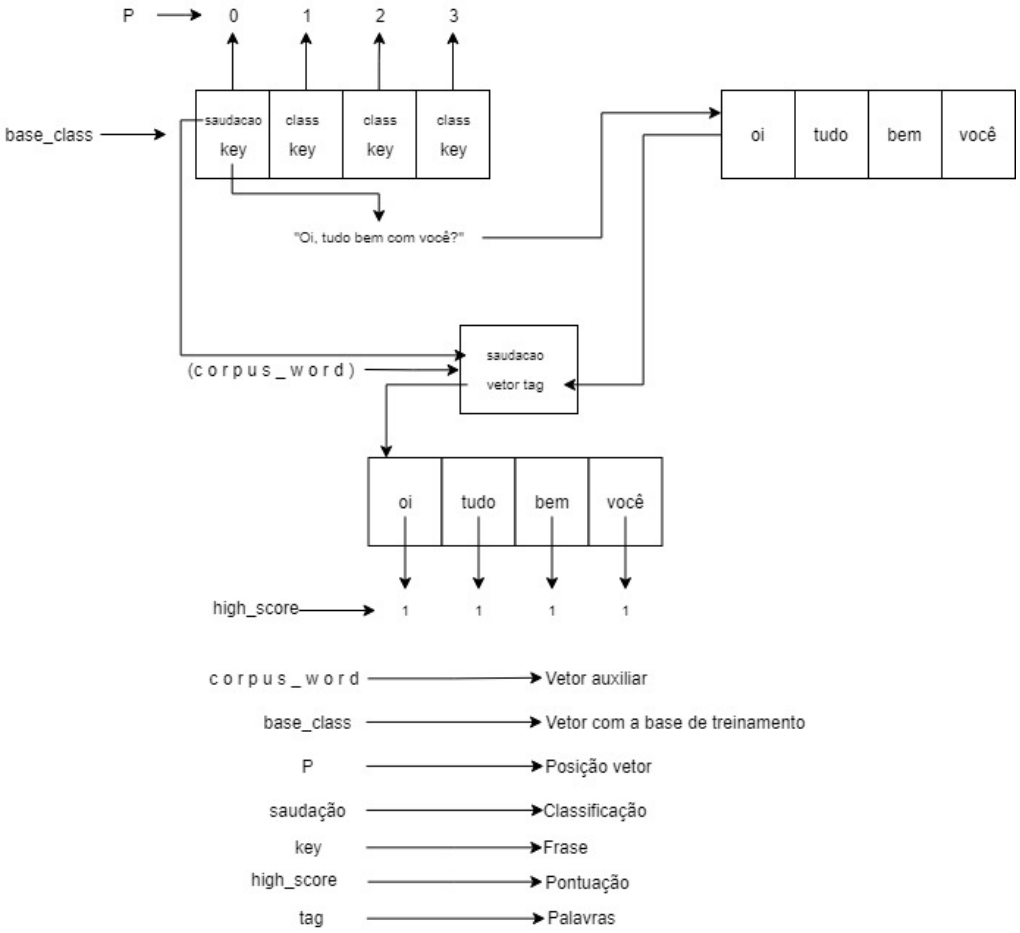


Figura 9 – ilustração função Learning

Fonte: Do autor

com o resultado do classificador. Um laço percorre todas as classes e armazena a quantidade de pontos que aquela frase obteve em relação àquela classe.E isso será repetido até o laço percorrer todas as classes contidas dentro do dicionário. e no final ela retornara o nome da classe que obteve a maior quantidade de pontos.

Outra função que auxilia a CalculateScore(), é a CalculateClassScore(), essa função recebe os parâmetros da frase (sentence) e a classe na qual a frase será comparada. Em seguida a função realiza alguns tratamentos na frase (sentence) como o Stemming e Tokenize. Após a frase ser submetida a esses tratamentos, um laço percorre todas as palavras da frase (sentence) e procura dentro da classe informada as frases que correspondentes, calculando assim a quantidade total de pontos e retornando o valor para a variável (pontos):

```
1
2 def CalculateClassScore(sentence,class_name):
3     score = 0
4     sentence = Tokenize(sentence)
5     sentence = Stemming(sentence)
6     for word in sentence:
7         if word in dados[class_name]:
8             score += dados[class_name][word]
9     return score
```

Após as funções acima retornar o nome da classe que obteve a maior quantidade de pontos, utilizamos algumas etapas para processar uma resposta para o usuário dentro da classificação feita pela analise:

```
1     output=''
2         try:
3             previous = random.choice(Previous(class_name))
4             if previous:
5                 output = previous+' '
6         except Exception as e:
7             print('Exception: '+str(e))
8         try:
9             responses = random.choice(Response(class_name))
10            output+=responses
11        except Exception as e:
12            print('Exception: '+str(e))
13        try:
14            nexts = random.choice(Next(class_name))
15            if nexts:
16                output+=', '+nexts
17        except Exception as e:
18            print('Exception: '+str(e))
19        if person:
20            output+= ', '+person
21        print('chat diz:'+output)
22        voice.Voice.SetVoice(output)
```

Após o sistema receber a entrada da informação do usuário, realizar os tratamentos de Tokenização, Stemming e Stopwords da frase e buscar e localizar a questão pelas classes já treinadas, O processamento acima consiste em uma variável que será utilizada para concatenar de modo randômico a previa da resposta, a resposta e a conclusão da resposta justamente nessa ordem. E no final, a variável já estará com a resposta final para ser apresentada. Caso a resposta da questão não for encontrada, uma frase padrão "não tenho resposta"é acionada no lugar e encaminhada para os sintetizadores de voz e para a tela para ser apresentado ao usuário.


Figura 10 – Tela Principal

Fonte: Do autor

Esta é a interface que são realizadas as interações com o usuário. Nela, há o campo onde são inseridas a entrada das informações, o campo onde são exibidas as informações de retorno e o botão cujo a função de ligar/desligar o microfone e o envio das informações para ser processado pela ferramenta.

## 4.4 ANÁLISE DO RESULTADO

Para a análise qualitativa foi utilizado um questionário de opinião. O questionário foi criado com base utilizado por Filho (2018). O mesmo tem o intuito de coletar informações para compreender melhor a necessidade dos usuários em um ambiente que utiliza o sistema, segue anexo:

**AVALIAÇÃO DE UMA FERRAMENTA DIDÁTICA-PEDAGÓGICA PARA ENSINO DE ARQUITETURA DE COMPUTADORES**

1 - Qual seu nível de conhecimento sobre Arquitetura de Computadores?
( ) Passei a conhecer por meio da ferramenta desta pesquisa.
( ) Conheço pouco.
( ) Conheço moderadamente.
( ) Conheço muito.

2 - Ficou claro o significado e a importância da disciplina?

( ) Sim ( ) Não ( ) Em parte

3 - Na sua opinião, os objetivos da disciplina foram alcançados?

( ) Sim ( ) Não ( ) Em parte

4 - O conteúdo da disciplina foi apresentado de forma adequada pela ferramenta?

( ) Sim ( ) Não ( ) Em parte

5 - Ficou claro a utilidade do sistema?

( ) Sim ( ) Não ( ) Em parte

**Perguntas relacionadas à usabilidade da ferramenta. Grau 1 significa discordância total e grau 5 significa concordância total**

6 - Qual foi o grau de facilidade ao utilizar esta ferramenta?
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

7 - Qual foi o grau de satisfação ao utilizar o sistema?
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

8 - Para você, qual é o grau de utilidade da ferramenta proposta para os estudantes e professores?
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

9 - Caso queira, deixe a sua opinião sobre a ferramenta!

Figura 11 – Questionário

Fonte: Do autor

Foram analisados 75 questionários onde 21 deles foram realizados por Jovens cursando o ensino superior e 54 por crianças do 5º e 6º ano do ensino fundamental. A primeira analise apresenta o nível de conhecimento sobre Arquitetura de Computadores, onde 25,33% dos alunos passaram a conhecer o conteúdo por meio da ferramenta, 40% já possuía algum tipo de conhecimento sobre o conteúdo, 30,67% possuía um conhecimento moderado, 2,67% possuía um conhecimento avançado sobre o conteúdo e somente 1,33% não responderam esse tópico. Esse resultado apresentou uma resposta positiva ao que se refere-se o objetivo da ferramenta, segue abaixo o gráfico.

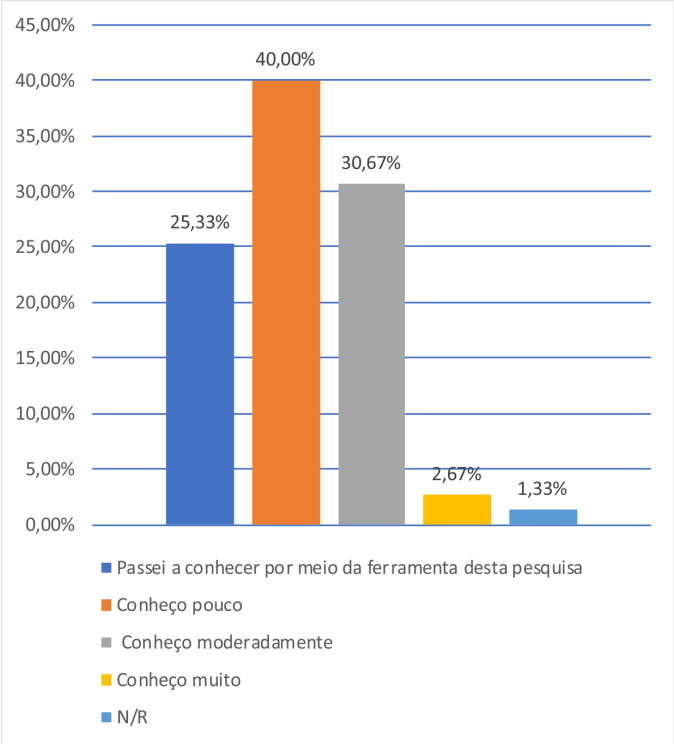


Figura 12 – nível de conhecimento sobre Arquitetura de Computadores

Fonte: Do autor

A segunda analise é referente a clareza sobre a importância e o significado da disciplina, aqui o resultado aponta que 85,33% dos alunos entenderam sim sobre a importância e significado da disciplina, 2,67% não entenderam e 12% entenderam parcialmente o que foi apresentado.

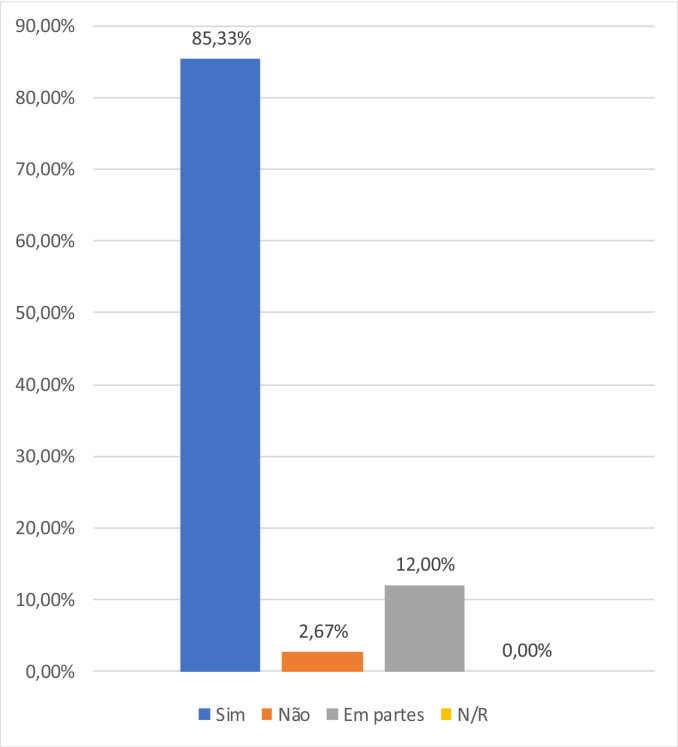


Figura 13 – Clareza, importância e o significado da disciplina

Fonte: Do autor

A terceira analise aponta se os objetivos da disciplina foram alcançados, aqui o resultado aponta que 66,67% dos alunos disseram que o objetivo foi alcançado sim, 6,67% disseram que não e 26,67% apontam que o objetivo foi alcançado parcialmente.

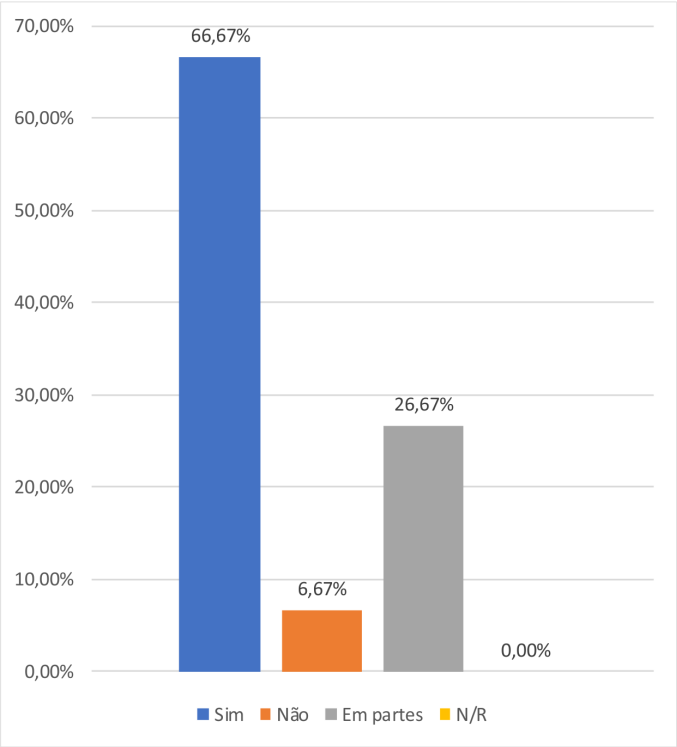


Figura 14 – Objetivos da disciplina

Fonte: Do autor

A quarta analise avalia se o conteúdo da disciplina foi apresentado de forma adequada pela ferramenta, aqui o resultado aponta que 81,33% dos alunos disseram que o objetivo foi alcançado sim, 2,67% disseram que não, 14,67% apontam que o objetivo foi alcançado parcialmente e somente 1,33% não responderam a esse tópico.

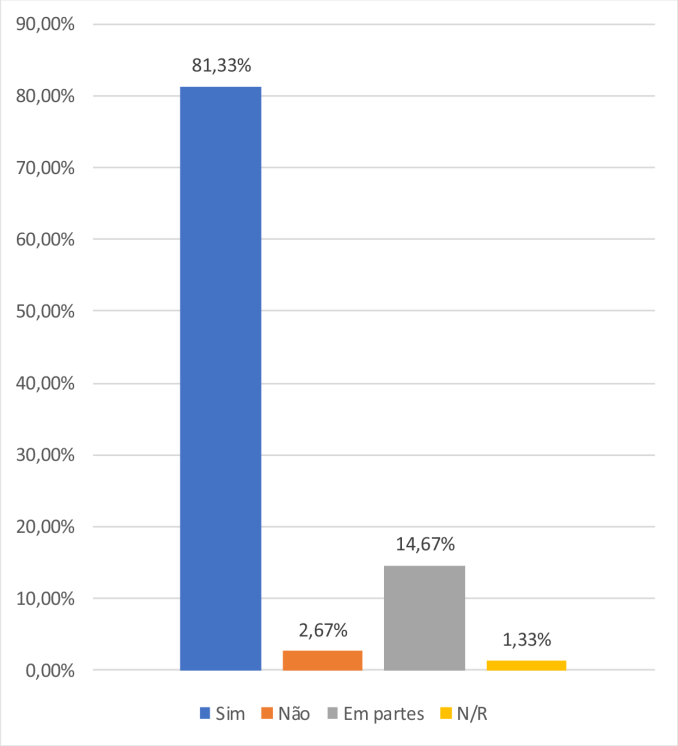


Figura 15 – Avaliação da disciplina apresentada pela ferramenta

Fonte: Do autor

A quinta analise avalia clareza sobre a utilidade da ferramenta, aqui o resultado aponta que 89,33% dos alunos disseram que sim, 4% disseram que não, 5,33% apontam que o objetivo foi alcançado parcialmente e somente 1,33% não responderam a esse tópico.

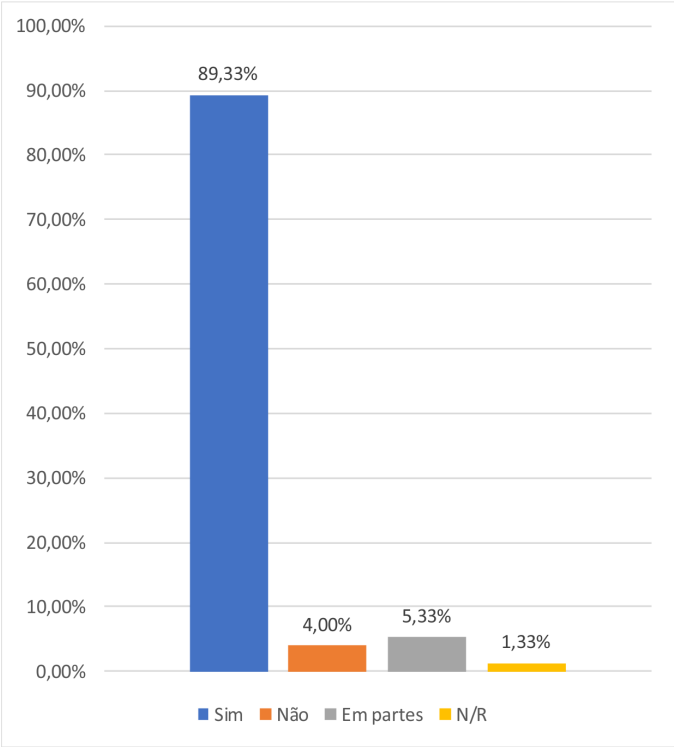


Figura 16 – Avaliação sobre a utilidade da ferramenta

Fonte: Do autor

Os próximos tópicos são relacionados a usabilidade da ferramenta. começando pela sexta analise que avalia o grau de facilidade ao utilizar esta ferramenta em uma escala de 1 para discordância total a 5 para concordância total. O resultado apontou uma igualdade na escala de 4 e 5 de 33,33%, na escala 3 de 16%, já a escala 2 de 10,67%, a escala 1 de 5,33% e 1,33% não responderam a esse tópico.

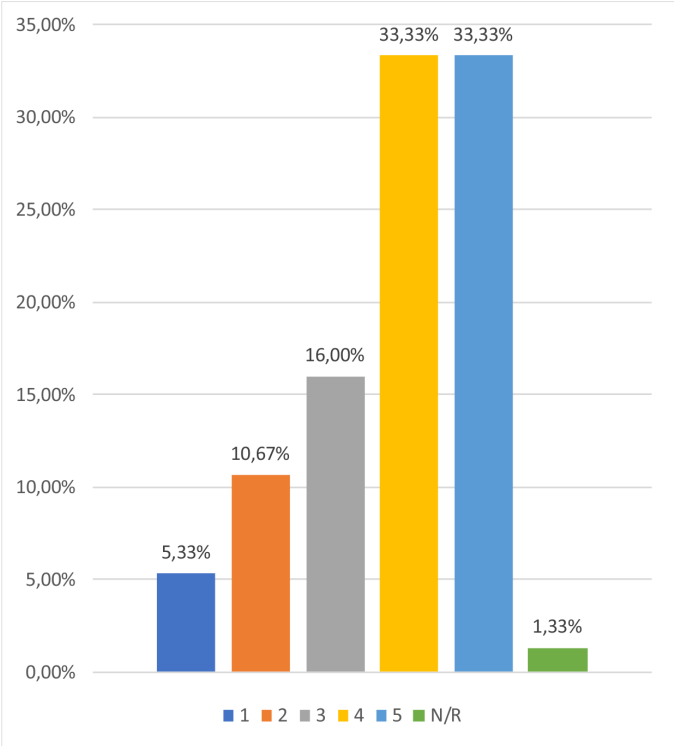


Figura 17 – Avaliação sobre a usabilidade da ferramenta

Fonte: Do autor

A sétima analise que avalia o grau de satisfação ao utilizar o sistema. O resultado demonstra que na escala 5 44% na escala 4 22,67%, na escala 3 de 18,67%, já a escala 2 de 6,67%, a escala 1 de 5,33% e 2,67% não responderam a esse tópico.

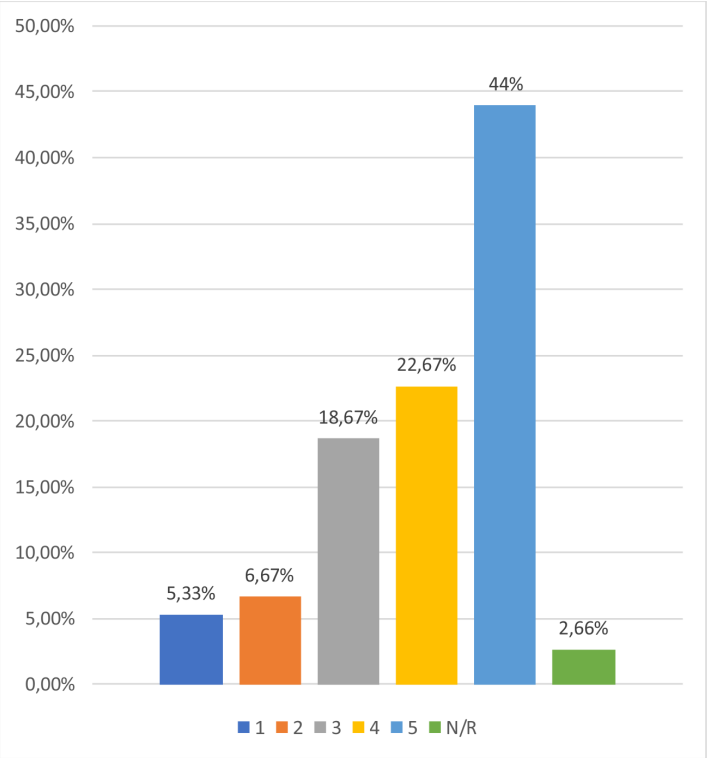


Figura 18 – Grau de satisfação do sistema

Fonte: Do autor

A oitava analise avalia o grau de utilidade da ferramenta proposta para os estudantes e professores. O resultado demonstra que na escala 5 54,67% na escala 4 17,33%, na escala 3 de 16%, já a escala 2 de 9,33%, a escala 1 de 1,33% e 1,33% não responderam a esse tópico.

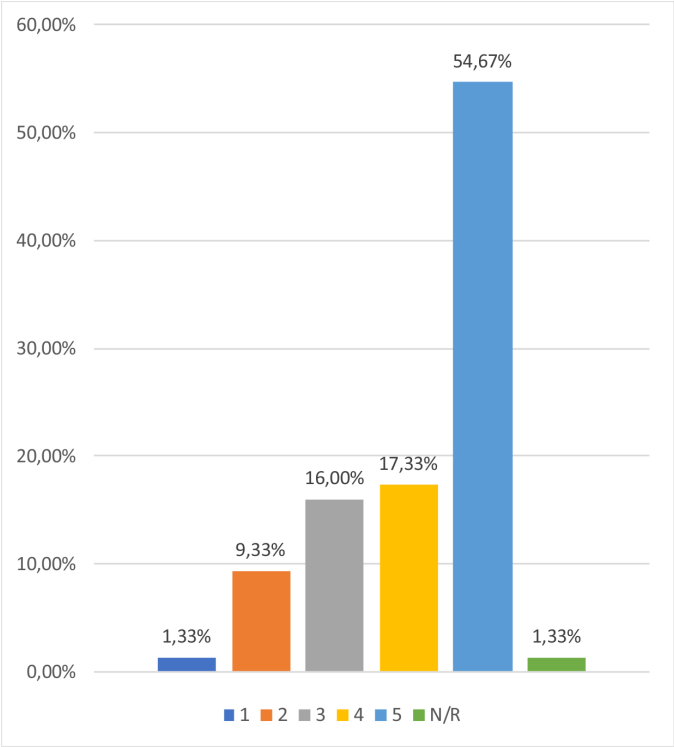


Figura 19 – Avaliação sobre a utilidade da ferramenta

Fonte: Do autor

# 5 CONCLUSÃO

Este trabalho de conclusão mostrou e proporcionou o desenvolvimento e avaliação de um protótipo de um Sistema de Perguntas e Respostas inteligente capaz de auxiliar no ensino didático-pedagógico focada no ensino de arquitetura de computadores.

Durante o desenvolvimento deste projeto diversas técnicas foram estudadas, implementadas e criadas a fim de alcançar o objetivo. Durante a implementação realizada o que mais se aproximou da realização do objetivo deste trabalho foi a união do uso de recursos como processamento de linguagem natural, inteligência artificial, Recuperação da Informação, métodos de captura e conversão de áudio para texto e texto para áudio, e recuperação de informação.

A utilização de Linguagem Natural em conjunto com outras aplicações pode trazer vantagens para implementações de sistemas inteligentes, uma delas consequentemente é a facilidade de interação, deste modo, o protótipo permite que o usuário não se sinta constrangido ao manusear e interagir com ele. Os métodos de entrada e saída de informações também facilitam esse processo, dando ao usuário duas opções de interação que é tanto por áudio como por texto.

Outro ponto a ser destacado está no processo de treinamento, protótipo contém uma interface pensada para usuários que não possua um amplo domínio da linguagem de programação. Contendo os campos de preenchimento específicos para o treinamento, o sistema otimiza a realização dessa tarefa além de proporcionar ferramentas de busca e edição de informações já inseridas.

Diante dos resultados analisados, pode-se concluir que o protótipo contribuiu muito para o aprendizado dos alunos referente à disciplina de Arquitetura de Computadores. A sua utilização na área de ensino apresentou ser viável, oportuna, e que desperta o interesse e receptividade pelo protótipo aplicado, auxiliando-os no processo de assimilação do conteúdo e construção do conhecimento.

# REFERÊNCIAS

ABREU, B. G. de. *Desenvolvimento de um sistema Web para utilização e gerenciamento de dados de Cupons Fiscais e Saúde*. 2016. Disponível em: <http://www.monografias.ufop.br/bitstream/35400000/403/1/MONOGRAFIA_DesenvolvimentoSistemaWeb.pdf>. Acesso em: 26 mar. 2019. Citado na página 24.

AGOSTI, C. *Interface em Linguagem Natural para Banco de Dados: uma abordagem prática.* 2003. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/86349/198627.pdf?sequence=1>. Acesso em: 05 jun. 2018. Citado na página 16.

AGOSTI, C.; BRANDAO, P. *A UTILIZAÇÃO DE AUTÔMATOS FINITOS NA TRADUÇÃO DE PORTUGUÊS PARA LIBRAS.* 2010. Disponível em: <http://www.revista.unisal.br/sj/index.php/123/article/view/69/82>. Acesso em: 01 abr. 2018. Citado 2 vezes nas páginas 11 e 13.

AHUJA, D. N. J.; SILLE, R. *A Critical Review of Development of Intelligent Tutoring Systems: Retrospect, Present and Prospect*. 2013. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.401.6425&rep=rep1&type=pdf>. Acesso em: 21 fev. 2019. Citado na página 8.

ALVES, A. F. da C.; PINA, L. E. de O.; GOMES, W. G.; SOUZA, A. P. de; SANTOS, D. S. *INTELIGÊNCIA ARTIFICAL: Conceitos, Aplicações e Linguagens*. 2017. Disponível em: <http://revistaconexao.aems.edu.br/wp-content/plugins/download-attachments/includes/download.php?id=1532>. Acesso em: 20 fev. 2019. Citado na página 5.

AMOS, D. *The Ultimate Guide To Speech Recognition With Python*. 2018. Disponível em: <https://pypi.org/project/SpeechRecognition/>. Acesso em: 23 mai. 2018. Citado na página 26.

ANDRADE RAFAEL COUTO BARROS, M. A. B. d. S. L. M. S. *PROCESSAMENTO DE LINGUAGEM NATURAL (PLN): FERRAMENTAS E DESAFIOS*. 2016. Disponível em: <http://www.editorarealize.com.br/revistas/conidis/trabalhos/TRABALHO_EV064_MD1_SA6_ID2260_09102016215649.pdf>. Acesso em: 28 mai. 2018. Citado 2 vezes nas páginas 19 e 20.

AZEVEDO, E. *Desenvolvimento de Jogos e Aplicações em Realidade Virtual.* [S.l.]: Elsevier, 2005. Citado na página 5.

BARR, A.; FEIGENBAUM, E. A. *The Handbook of artificial intelligence*. [S.l.]: Stanford, Calif.: HeurisTech Press ; Los Altos, Calif. : William Kaufmann, 1981. 3–17 p. Citado na página 4.

BATISTA, G. E. de A.; MONARD, M. C. *Utilizando Métodos Estatísticos de Resampling para Estimar a Performance de Sistemas de Aprendizado*. 1998. Disponível em: <http://conteudo.icmc.usp.br/pessoas/mcmonard/public/icmcwshg1998.pdf>. Acesso em: 20 fev. 2019. Citado na página 6.

BIBER, D.; REPPEN, R.; CONRAD, S. *Corpus-based approaches to issues in applied linguistics*. [S.l.]: Oxford University Press, 1994. 169–189 p. Citado na página 18.

BIRD, S.; KLEIN, E.; LOPER, E. *Natural Language Processing with Python.* 2009. Disponível em: <https://www.nltk.org/book>. Acesso em: 20 abr. 2018. Citado 4 vezes nas páginas 11, 12, 17 e 20.

BITTENCOURT, G. *Inteligência Computacional*. 2011. Disponível em: <http://www.egov.ufsc.br/portal/sites/default/files/anexos/6510-6509-1-PB.pdf>. Acesso em: 19 fev. 2019. Citado 2 vezes nas páginas 4 e 5.

CURADO, J. D. F. *Geração Automática de Metadados para Publicações Eletrônicas*. 2010. Disponível em: <https://bcc.ime.usp.br/tccs/2010/josedavid/capa/monografia.pdf>. Acesso em: 03 mar. 2019. Citado na página 13.

DAMASCENO, F. R. *Concepção e Desenvolvimento do Agente Tutor e Modelo de Aluno no Ambiente inteligente de Aprendizagem PAT2MATH*. 2011. Disponível em: <http://www.repositorio.jesuita.org.br/bitstream/handle/UNISINOS/3940/32.pdf?sequence=1&isAllowed=y>. Acesso em: 21 fev. 2019. Citado na página 8.

DUARTE, V. *Semântica.* 2008. Disponível em: <https://brasilescola.uol.com.br/portugues/a-semantica.htm>. Acesso em: 01 abr. 2018. Citado na página 15.

DUARTE, V. M. do N. *Análise Sintática e Morfológica.* 2009. Disponível em: <https://mundoeducacao.bol.uol.com.br/gramatica/analise-sintatica-morfologica.htm>. Acesso em: 01 abr. 2018. Citado na página 13.

DUQUE, J. L. *INSTANCIAÇÃO, VALIDAÇÃO E EXTENSÃO DE UMA METODOLOGIA DE EXTRAÇÃO DE INFORMAÇÃO PARA OS ASSUNTOS "EFEITOS" E "TRATAMENTOS" NA DOENÇA ANEMIA FALCIFORME*. 2010. Disponível em: <http://gbd.dc.ufscar.br/~julianalduque/files/QualiJulianaDuque.pdf>. Acesso em: 21 fev. 2019. Citado 2 vezes nas páginas 6 e 7.

DURETTE, P. N. *Uma biblioteca Python e uma ferramenta CLI para interagir com a API de texto para voz do Google Tradutor.* 2014. Disponível em: <https://pypi.org/project/gTTS/>. Acesso em: 30 mar. 2018. Citado na página 25.

FADEL, A. A. *GeSTI: UM SISTEMA TUTOR INTELIGENTE APLICADO AO ENSINO DE MECÂNICA*. 2005. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/14/artigos/DF-5-74972022787-1119047141163.pdf>. Acesso em: 21 fev. 2019. Citado na página 8.

FARINON, J. L. *Analise e classificação de conteúdo Textual.* 2015. Disponível em: <https://repositorio.unisc.br/jspui/bitstream/11624/1038/1/Jo%C3%A3o%20Lu%C3%ADs%20Farinon.pdf>. Acesso em: 20 mar. 2018. Citado 2 vezes nas páginas 10 e 17.

FERRUCCI, D.; BROWN, E.; CHU-CARROLL, J.; FAN, J.; GONDEK, D.; KALYANPUR, A. A.; LALLY, A.; MURDOCK, J. W.; NYBERG, E.; PRAGER, J.; SCHLAEFER, N.; ; WELTY, C. *Projeto "Building Watson: An Overview of the DeepQA Project"*. 2010. Disponível em: <https://aaai.org/ojs/index.php/aimagazine/article/view/2303>. Acesso em: 20 Mar. 2019. Citado 2 vezes nas páginas 23 e 24.

FILHO, E. C. P. *O Uso do Processamento de Linguagem Natural na Construção de Chatterbots*. 2009. Disponível em: <https://dcc.catalao.ufg.br/up/498/o/Eustaquio2009.pdf>. Acesso em: 22 fev. 2019. Citado na página 12.

FILHO, P. C. B. B. A inclusÃo espacial atravÉs da webvr: Um estudo de usabilidade. *CENTRO UNIVERSITÁRIO DA GRANDE DOURADOS*, p. 52–53, 2018. Citado na página 35.

GONZALEZ, M.; LIMA, V. L. S. de. *Recuperação de Informação e Processamento da Linguagem Natural*. 2003. Disponível em: <https://www.researchgate.net/publication/

228608574_Recuperacao_de_Informacao_e_Processamento_da_Linguagem_Natural>. Acesso em: 03 mar. 2019. Citado 4 vezes nas páginas 13, 14, 15 e 16.

GRANATYR, J. *Conversação com ELIZA!* 2016. Disponível em: <http://iaexpert.com.br/index.php/2016/10/18/historico-da-ia>. Acesso em: 28 mar. 2018. Citado na página 10.

LEE, G. G.; SEO, J.; LEE, S.; JUNG, H.; CHO, B.-H.; LEE, C.; KWAK, B.; CHA, J.; KIM, D.; AN, J.; KIM, H.; KIM, K. *Projeto "SiteQ: Engineering High Performance QA system Using LexicoSemantic Pattern Matching and Shallow NLP "*. 2001. Disponível em: <https://trec.nist.gov/pubs/trec10/papers/SiteQ_trec10.pdf>. Acesso em: 20 Mar. 2019. Citado na página 23.

MATOS, P. F.; LOMBARDI, L. de O.; PARDO, P. D. T. A. S.; CIFERRI, P. D. C. D. de A.; VIEIRA, P. D. M. T. P. *Projeto "Um Ambiente para análise de Dados da Doença Anemia Falciforme"*. 2009. Disponível em: <http://conteudo.icmc.usp.br/pessoas/taspardo/TechReportUFSCar2009b-MatosEtAl.pdf>. Acesso em: 20 fev. 2019. Citado na página 6.

MCCARTHY, J.; HAYES, P. J. *SOME PHILOSOPHICAL PROBLEMS FROM THE STANDPOINT OF ARTIFICIAL INTELLIGENCE*. 1969. Disponível em: <http://www.inf.ufsc.br/~mauro.roisenberg/ine6102/leituras/mcchay69.pdf>. Acesso em: 19 fev. 2019. Citado na página 4.

MCCORDUCK, P. *Machines Who Think : A Personal Inquiry into the History and Prospects of Artificial Intelligence*. [S.l.]: A K Peters, Ltd. Natick, Massachusetts, 2004. 111–136 p. Citado na página 4.

MONARD, M. C.; BARANAUSKAS, J. A. *Sistemas Inteligentes: Fundamentos e Aplicações*. [S.l.]: Manole, 2003. 89–114 p. Citado 2 vezes nas páginas 6 e 7.

MULLER, F. B. *Desenvolvimento de Interface em Python/Django para o NEHiLP*. 2017. Disponível em: <https://linux.ime.usp.br/~fabiobm/mac0499/monografia.pdf>. Acesso em: 26 mar. 2019. Citado na página 24.

NETO, J. M. de O.; TONIN, S. D.; PRIETCH, S. S. *Processamento de Linguagem Natural e suas Aplicações Computacionais.* 2010. Disponível em: <https://www.inpa.gov.br/erin2010/Artigo/Artigo9.pdf>. Acesso em: 01 abr. 2018. Citado na página 16.

NYE, J. *Speaking in Tongues.* 2016. Disponível em: <https://www.sciencehistory.org/distillations/magazine/speaking-in-tongues>. Acesso em: 27 mar. 2018. Citado na página 10.

OLIVEIRA, F. A. D. de. *Processamento de linguagem natural: princípios básicos e a implementação de um analisador sintático de sentenças da língua portuguesa.* 2013. Disponível em: <https://docslide.com.br/documents/processamento-de-linguagem-natural-principios-basicos-e-a-implementacao-de.html>. Acesso em: 01 abr. 2018. Citado 5 vezes nas páginas 10, 14, 15, 17 e 18.

RABUSKE, R. A. *Inteligência Artificial*. Brasil: Editora da UFSC, 1995. Citado na página 10.

ROZA, F. S. da. *Aprendizagem de máquina para apoio à tomada de decisão em vendas do varejo utilizando registros de vendas.* 2016. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/171569/PFC_2016-1%20Felippe_Roza.pdf?sequence=1>. Acesso em: 28 mar. 2018. Citado na página 11.

RUSSELL, B.; NORVIG, P. *Inteligência Artificial*. 2004. Disponível em: <https://pt.scribd.com/doc/309173562/Inteligencia-Artificial-3a-Ed-Russell-Stuart-Norvig-Peter-pdf>. Acesso em: 09 mar. 2018. Citado na página 5.

RUSSELL, B.; NORVIG, P. *Inteligência Artificial*. Brasil: Elsevier, 2013. Citado na página 17.

SANTOS, P. P. dos. *Distinção entre Fonética e Fonologia.* 2006. Disponível em: <https://www.infoescola.com/portugues/distincao-entre-fonetica-e-fonologia>. Acesso em: 01 abr. 2018. Citado na página 12.

SAVADOVSKY, P. *Introdução ao projeto de interfaces em linguagem natural*. Brasil: SID Informática, 1988. Citado na página 16.

SCAPINI, I. K. *Relações entre Itens Lexicais*. Brasil: Anais, 1995. Citado na página 13.

SILVA, A. P. C. e. *APLICAÇÕES DE SISTEMAS TUTORES INTELIGENTES NA EDUCAÇÃO A DISTÂNCIA: POSSIBILIDADES E LIMITES*. 2006. Disponível em: <http://www.abed.org.br/seminario2006/pdf/tc056.pdf>. Acesso em: 21 fev. 2019. Citado na página 9.

SILVA, R. R.; LIMA, S. M. B. *Consultas em Bancos de Dados Utilizando Linguagem Natural.* 2016. Disponível em: <http://re.granbery.edu.br/artigos/MjQ0.pdf>. Acesso em: 09 mar. 2018. Citado na página 9.

SPEECHRECOGNITION. *Biblioteca para realização de reconhecimento de fala, com suporte para diversos mecanismos e APIs, online e offline.* 2017. Disponível em: <https://pypi.org/project/SpeechRecognition/>. Acesso em: 16 mar. 2018. Citado na página 26.

TURING, A. M. *Computing Machinery and Intelligence*. 1950. Disponível em: <https://academic.oup.com/mind/article/LIX/236/433/986238>. Acesso em: 09 mar. 2018. Citado 2 vezes nas páginas 4 e 10.

WINOGRAD, T. *SHRDLU*. 2014. Disponível em: <http://hci.stanford.edu/winograd/shrdlu>. Acesso em: 28 mar. 2018. Citado na página 10.