UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

MATHEUS XAVIER SAMPAIO

TRANSCRIÇÃO DE VOZ PARA TEXTO E SÍNTESE DE VOZ EM CHATBOTS

QUIXADÁ

2020

MATHEUS XAVIER SAMPAIO

TRANSCRIÇÃO DE VOZ PARA TEXTO E SÍNTESE DE VOZ EM CHATBOTS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Dr. Regis Pires Magalhães

QUIXADÁ

2020

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S184t Sampaio, Matheus Xavier.
Transcrição de voz para texto e síntese de voz em chatbots / Matheus Xavier Sampaio. – 2021.
63 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Ciência da Computação, Quixadá, 2021.
Orientação: Prof. Dr. Regis Pires Magalhães.

1. Chatterbot. 2. Reconhecimento Automático da voz. 3. Fala-Síntese. I. Título.

CDD 004

MATHEUS XAVIER SAMPAIO

TRANSCRIÇÃO DE VOZ PARA TEXTO E SÍNTESE DE VOZ EM CHATBOTS

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus Quixadá da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: ____/_____/_____.

BANCA EXAMINADORA

Prof. Dr. Regis Pires Magalhães (Orientador)
Universidade Federal do Ceará (UFC)

Profa. Dra. Ticiana Linhares Coelho da Silva
Universidade Federal do Ceará (UFC)

Prof. Dr. Davi Romero de Vasconcelos
Universidade Federal do Ceará (UFC)

Aos meus pais, minha irmã, minha avó e meu padrinho.

AGRADECIMENTOS

Primeiramente agradeço a minha mãe, Elizabeth Florentino Xavier, meu pai, Laécio Andrade Sampaio, minha irmã, Laymara Xavier Sampaio, minha avó, Maria Xavier Sampaio, e meu padrinho, Manoel Matias Felipe da Silva, por todo o apoio e carinho que me deram, e os sacrifícios que fizeram durante esta jornada.

Aos meus grandes amigos de infância Marcus Vinícius Nunes Guimarães, Mateus Parente Ripardo Oliveira, Lynda Veríssimo Teixeira Barroso e José Carlos Alves Filho, por todo o companheirismo, motivação e força que me deram.

Agradeço aos amigos que fiz durante estes quatro anos, pelos momentos de diversão e de estudo que compartilhamos, nossas experiências de morar longe de casa, e a tristeza de não poder criar mais momentos juntos em nosso ultimo ano de graduação devido a pandemia. Em especial, agradeço ao Francisco Marcelo da Silva Lima, João Marcelo Ravache Fernandes da Silva e Carlos Henrique Severo, por terem me aceitado quando pedi para morar com eles, sem nem exitar.

Agradeço ao Professor Regis Pires Magalhães pelos anos de trabalho como bolsista e orientando. Obrigado por todos os conselhos, ensinamentos, oportunidades, por ter acreditado em mim e pela ótima orientação neste e em outros trabalhos. Este aprendizado foi muito importante para a minha vida pessoal e acadêmica.

Aos professores Davi Romero de Vasconcelo e Ticiana Linhares Coelho da Silva, pela disponibilidade de participar da banca avaliadora, e por todas as dicas e conselhos passados em nossas várias reuniões semanais durante o desenvolvimento deste trabalho.

Agradeço imensamente à toda comunidade acadêmica da UFC Quixadá que me proporcionou um ambiente de aprendizagem incrível, no qual pude obter experiências, conhecimentos e lições que estarão comigo para sempre.

Agradeço ao Insight Lab por ter me acolhido com bolsista, pelo ambiente de trabalho que me foi proporcionado, e pela oportunidade de trabalhar em diferentes projetos, o que me trouxe experiência e grande aprendizagem. E à Fundação Cearense de Apoio ao Desenvolvimento (Funcap), pelo financiamento da pesquisa via bolsa de estudos.

“In War, Victory. In Peace, Vigilance. In Death, Sacrifice.“

(Dragon Age: Gray Wardens Moto)

RESUMO

A popularização de aplicativos de troca de mensagem mudou a forma em que as pessoas esperam interagir com a tenologia. Devido a isso, o uso de *chatbots* vem ganhando grande destaque por apresentar uma interface semelhante a estes aplicativos. O Governo do Estado do Ceará busca disponibilizar informações sobre seus serviços a população através da Carta de Serviços, e o uso de *chatbots* apresenta como uma boa interface para para tornar essas informações mais facilmente acessíveis. No entanto, enquanto a troca de mensagens de voz em aplicativos de troca de mensagem é bastante comum e permite a inclusão de usuários com dificuldades de comunicar por texto, a maioria dos *chatbots* não apresenta esta funcionalidade, o que acaba excluindo uma parcela de usuários. Visto isso, este trabalho busca a criação de uma interface capaz de receber e enviar mensagens por voz, criação de um conjunto de dados e treinamento de um modelo de rede neural para produzir uma voz cearense, para ser integrada aos *chatbots* desenvolvidos no projeto do Governo do Estado do Ceará: Governo Digital e Plataforma Big Data para acelerar a Transformação Digital, que serão disponibilizados em diferentes plataformas, incluindo aplicativos de troca de mensagem.

Keywords: Chatterbot. Reconhecimento Automático da voz. Fala-Síntese.

ABSTRACT

The popularization of message exchange applications has changed the way people expect to interact with technology. Because of this, the use of chatbots has been gaining great prominence for presenting an interface similar to these applications. The Government of the State of Ceará seeks to make information about its services available to the population through the Service Charter, and the use of chatbots is a good interface to make this information more easily accessible. However, while the exchange of voice messages in message exchange applications is quite common and allows the inclusion of users with difficulties in communicating by text, most chatbots do not have this functionality, which ends up excluding a share of users. In view of this, this work seeks to create an interface capable of receiving and sending messages by voice, creating a data set and training a neural network model to produce a Ceará voice, to be integrated with the developed chatbots in the project of the Government of the State of Ceará: Digital Government and Big Data Platform to accelerate the Digital Transformation, which will be made available on different platforms, including message exchange applications.

Keywords: Chatterbot. Automatic voice recognition. Speech Synthesis.

LISTA DE ILUSTRAÇÕES

Figura 1 – Exemplo de conversa com o *chatbot* ELIZA 18
Figura 2 – Exemplo de um arquivo *A.I.M.L.* 19
Figura 3 – Popularidade de termos de pesquisa pelo assunto *Chatterbot* no *Google* . . . 20
Figura 4 – Arquitetura do modelo wav2vec 25
Figura 5 – Arquitetura Tacotron 2 28
Figura 6 – Matriz de custo de alinhamento entre duas séries temporais 32
Figura 7 – Fluxo de atividades 37
Figura 8 – Quantidade de exemplos por tamanho de texto 48
Figura 9 – *Tacotron 2 Loss* 50
Figura 10 – *Multi-band MelGAN Loss* 50
Figura 11 – Progressão do alinhamento da camada de atenção 51
Figura 12 – Progressão dos espectrogramas gerados 52
Figura 13 – Conversa com o *chatbot* pelo *Whatsapp* e *Telegram* 54

LISTA DE TABELAS

Tabela 1 – Comparação de caracteristicas das *APIs* de *ASR* 40
Tabela 2 – Comparação de caracteristicas das *APIs* de *TTS* 41
Tabela 3 – Características dos conjuntos de dados 42
Tabela 4 – Métricas de avaliação por Conjunto de Dados 44
Tabela 5 – Uso de recursos e tempo de inferência dos modelos 45
Tabela 6 – Comparação de Espectrogramas utilizando DTW e Similaridade Cosseno . . 46
Tabela 7 – Avaliação *MOS* 52

LISTA DE QUADROS

Quadro 1 – Comparação com trabalhos relacionados 36
Quadro 2 – Máquina utilizada para testes de *ASR* 44
Quadro 3 – Máquina utilizada para testes de *TTS* 45
Quadro 4 – Descrição estatística dos dados 48

LISTA DE ABREVIATURAS E SIGLAS

ASR	Sistemas de reconhecimento de voz ou *Automatic Speech Recognition*
TTS	Sistemas de Síntese de Voz ou *Text to Speech*
BLEU	Avaliação Bilíngue de Substituição ou *Bilingual Evaluation Understudy*
METEOR	Métrica para Avaliação de Tradução com Ordenação Explícita ou *Metric for Evaluation of Translation with Explicit ORdering*
IA	Inteligência Artificial
NLP	Processamento de Linguagem Natural ou *Natural Language Processing*
A.I.M.L.	*Artificial Inteligence Markup Language*
DL	Aprendizagem Profunda ou *Deep Learning*
CNN	Rede Neural Convolucional ou *Convolutional Neural Networks*
GAN	Redes Geradoras Adversárias ou *Generative Adversarial Networks*
MFCC	Componentes Mel Cepstrais ou *Mel Frequency Cepstral Coefficient*
HMM	Modelo Oculto de Markov ou *Hidden Markov Model*
CTC	Classificação Temporal Conexionista ou *Connectionist Temporal Classification*
Vocoder	Modelo Gerador de Formas de Onda
WER	Taxa de Erro de Palavras ou *Word Error Rate*
MOS	Pontuação Média de Opinião ou *Mean Opinion Score*
DTW	Distorção Temporal Dinâmica ou *Dynamic Time Warping*
API	*Application Programming Interface*
NLU	Compreensão de Linguagem Natural ou *Natural Language Understanding*

SUMÁRIO

1 INTRODUÇÃO 15
1.1 Objetivos 16
2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 *Chatbots* 17
2.1.1 Ascensão dos Chatbots 19
2.2 Redes Neurais Profundas 20
2.2.1 Redes Neurais Convolucionais 21
2.2.2 Codificador-Decodificador com camada de Atenção 21
2.2.3 Redes Geradoras Adversárias 22
2.3 Sistemas de Reconhecimento de Voz 22
2.3.1 Sistemas tradicionais de ASR 23
2.3.2 Evolução em ASR 24
2.4 Sistemas de Síntese de Voz 24
2.4.1 Técnicas tradicionais de TTS 25
2.4.2 Nova Geração em TTS 26
2.5 Métricas de Avaliação 28
2.5.1 Avaliação de Reconhecimento de Voz 28
2.5.1.1 Taxa de Erro de Palavras *29*
2.5.1.2 Distância de Jaccard *29*
2.5.1.3 BLEU e METEOR *29*
2.5.1.4 Similaridade Cosseno *31*
2.5.2 Avaliação de Síntese de Voz 31
2.5.2.1 Similaridade Cosseno *32*
2.5.2.2 Distorção Temporal Dinâmica *32*
2.5.2.3 Pontuação Média de Opinião *33*
3 TRABALHOS RELACIONADOS 34
3.1 *Android Based Educational Chatbot for Visually Impaired People* 34
3.2 *BookBuddy: Turning Digital Materials Into Interactive Foreign Language Lessons Through a Voice Chatbot* 34
3.3 *Solução Mobile – Voice Assisted Chatbots* 35

3.4 **Análise Comparativa** 35
4 **METODOLOGIA** 37
4.1 **Revisão bibliográfica sobre métodos de transcrição e síntese de voz** . . . 37
4.2 **Obtenção de conjunto de dados** 37
4.3 **Avaliação dos métodos de transcrição de voz** 38
4.4 **Correção de problemas nas transcrições** 38
4.5 **Avaliação dos métodos de síntese de voz** 38
4.6 **Integração da transcrição e síntese ao *chatbot*** 38
4.7 **Criação de conjunto de dados e treinamento do modelo** 39
4.8 **Integração do *chatbot* a canais** 39
5 **EXPERIMENTOS E RESULTADOS** 40
5.1 **Identificação de ferramentas para transcrição e síntese de voz** 40
5.2 **Conjuntos de dados para experimentos** 41
5.3 **Experimentos *ASR*** 43
5.4 **Experimentos *TTS*** 45
5.5 **Integração de *ASR* e *TTS* ao chatbot** 47
5.6 **Criação do conjunto de dados customizado** 47
5.7 **Treinamento do modelo** 48
5.8 **Performance do modelo treinado** 50
5.9 **Integração do *Chatbot* a canais de comunicação** 52
6 **CONCLUSÕES E TRABALHOS FUTUROS** 55
REFERÊNCIAS 57
APÊNDICE A–LINKS PARA APIS 62
A.0.1 ***APIs para ASR*** 62
A.0.2 ***APIs para TTS*** 62
A.0.2.1 *TTS com voz customizada* *62*
APÊNDICE B–ESTRUTURA DE PASTAS DO CORPUS DE *ASR* . . . 63

1 INTRODUÇÃO

Desde 2016, tem-se observado um crescimento no interesse e popularidade de *chatbots*, o que se dá devido a popularização de serviços de troca de mensagem, como *Facebook Messenger*, *Telegram*, *WhatsApp*, pelos avanços no campo de Inteligência Artificial (IA), e devido a interação por conversação ser mais fácil e intuitiva. O uso de mensagens de texto para se comunicar é comum mesmo entre aqueles com dificuldade em usar aplicativos ou sites, por ser uma forma mais natural de interagir (ACCENTURE INTERACTIVE, 2016).

Em 2016, os aplicativos de mensagens já eram usados por mais de 2,1 bilhões de pessoas: 49% das pessoas entre 18 e 29 anos, 37% de pessoas entre 30 e 49 anos e 25% daqueles com mais de 50 anos (DALE, 2016). O *Facebook Messenger*, por exemplo, teve mais de 1,2 bilhão de usuários ativos por mês em 2017, e em 2018 já tem mais de 200 mil *chatbots* (BRANDTZÆG; FØLSTAD, 2018). Em 2017 o *WhatsApp*, aplicativo de mensagem mais popular no Brasil, já havia chegado a marca de 1.5 bilhões de usuários ativos por mês, com cerca de 60 bilhões de mensagens enviadas todos os dias em todo mundo, e em 2020 atingiu a marca de 2 bilhões de usuários [1]. Segundo uma pesquisa realizada pela *Mobile Time/Opinon Box*[2], o *WhatsApp* está instalado no smartphone de 99% dos brasileiros, e 93% usam o aplicativo todo dia. 90% dos brasileiros usam o *WhatsApp* para enviar mensagens de texto, e 81% se comunicam por mensagens de áudio. Já o *Telegram* chegou a 45% dos celulares no país[3].

Com a popularidade dos aplicativos de mensagens, os usuários esperam poder utilizar a mesma forma de comunicação em outros serviços. Assim, é comum ao utilizar web-sites ou aplicativos que oferecem algum tipo de serviço ou suporte, ser cumprimentado por uma caixa de texto desses *chatbots* oferecendo ajuda ao usuário. No entanto, oferecer uma interação apenas por entrada e saída de texto pode ser um impeditivo para o usuário utilizar o *chatbot*, seja por que o usuário está com as mãos e olhos ocupados no momento, por ter alguma deficiência motora ou visual (COHEN; OVIATT, 1995), ou possuir baixo letramento. Uma interface de iteração por voz é capaz de mitigar esta limitação.

Outra vantagem que a iteração por voz traz é um maior engajamento do usuário com o *chatbot*, pois como demonstrado no estudo realizado em Grigore *et al.* (2016), embora a utilização dessa forma de iteração seja mais sujeita a erros e menos robusta do que apenas por texto, os usuários percebem a entidade como mais relacionável e presente, passando assim

[1] https://blog.whatsapp.com/two-billion-users-connecting-the-world-privately
[2] https://panoramamobiletime.com.br/pesquisa-mensageria-no-brasil-fevereiro-de-2020/
[3] https://olhardigital.com.br/2021/03/05/internet-e-redes-sociais/telegram-base-usuarios-celulares-brasileiros/

mais confiança ao usuário durante a interação. Isso destaca a importância de empregar uma forma de comunicação mais natural, mesmo que represente uma escolha menos robusta. Uma interface de linguagem natural bem projetada deve oferecer suporte à aceitação de tecnologias e serviços digitais em grupos com menos capacidade ou conhecimento de tecnologia (FØLSTAD; BRANDTZÆG, 2017).

Permitir o acesso destes grupos aos serviços oferecidos de forma digital pelo Governo do Estado do Ceará é um dos principais objetivos do projeto Governo Digital e Plataforma Big Data para acelerar a Transformação Digital do Estado do Ceará. Trazer para estes serviços uma forma de interação com a qual as pessoas estão mais habituadas é essencial para que elas possam utilizar os serviços independente de sua idade, escolaridade ou contexto social. Assim, este trabalho tem como objetivo adicionar ao *chatbot* da Carta de Serviços do Estado do Ceará[4], que fornece um banco de dados de perguntas frequentes referentes a diversos serviços, uma interface de comunicação por voz e integrar esta interface aos canais de comunicação e informação do Governo do Estado do Ceará, como o Portal Ceará Digital, Ceará App, e aplicativos de troca de mensagem.

1.1 Objetivos

O objetivo principal deste trabalho é utilizar técnicas para permitir a interação através de conversação por voz com *chatbots*.

Para isso, os objetivos específicos são:

a) Interpretar o discurso de voz do usuário transcrevendo em forma de texto.
b) Fornecer a sentença ao motor do *chatbot*, para que possa ser interpretada.
c) Receber do motor do *chatbot* o resultado da solicitação do usuário.
d) Responder ao usuário sintetizando a resposta em voz, caso necessário.
e) Criar um conjunto de dados de texto e voz cearense
f) Treinar uma Rede Neural para gerar voz cearense

Este trabalho está organizado da seguinte forma: o Capítulo 2 traz os conceitos teóricos que fundamentam o trabalho; o Capítulo 3 apresenta os trabalhos relacionados; o Capítulo 4 apresentará os passos estabelecidos para a execução deste trabalho; o Capítulo 5 apresenta os resultados obtidos; o Capítulo 6 conclui o trabalho, resumindo as resultados e apresentando possíveis trabalhos futuros.

[4] https://cartadeservicos.ce.gov.br/ConsultaCesec/pg_cs_servico.aspx

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados os conceitos que fundamentam o desenvolvimento deste trabalho. A Seção 2.1 apresenta os conceitos referentes a *chatbots*, apresentando um pouco da história, fundamentos e aplicações. A Seção 2.2 traz alguns conceitos sobre Redes Neurais. Já as Seções 2.3 e 2.4 trazem as tecnologias que permitem realizar o reconhecimento e síntese automáticos de voz. Na Seção 2.5 serão apresentadas algumas formas de avaliar o reconhecimento e síntese de voz.

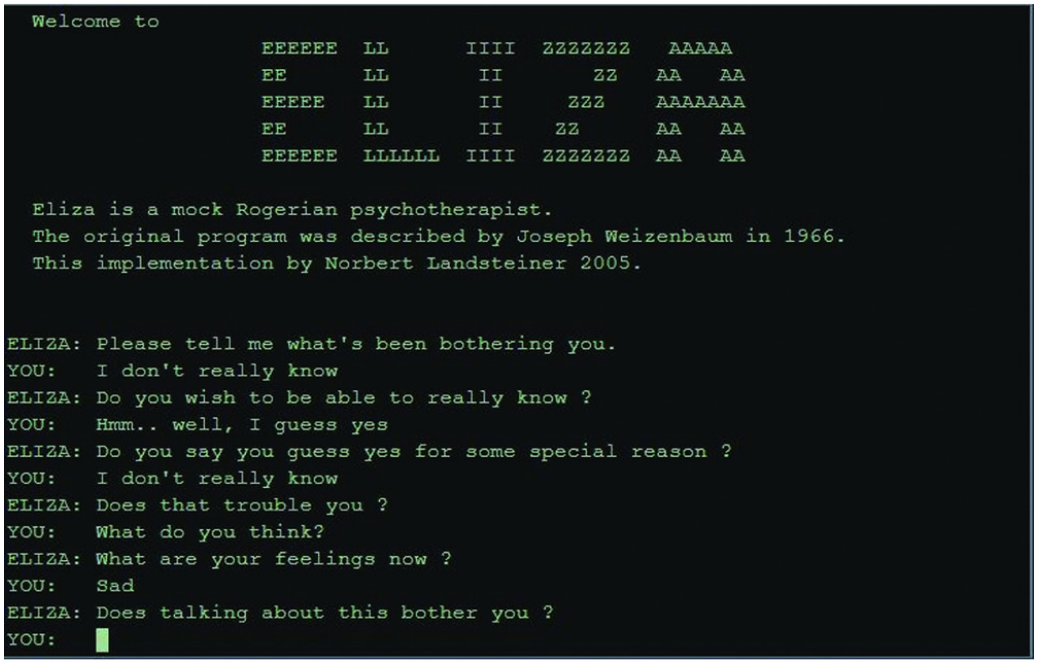
2.1 *Chatbots*

Um *chatbot* é um exemplo típico de um sistema de IA e um dos mais elementares e generalizados exemplos de Interação Humano-Computador inteligente. É um programa de computador capaz de interagir com o usuário por meio de texto ou voz, que responde como uma entidade inteligente através do uso de Processamento de Linguagem Natural ou *Natural Language Processing* (*NLP*) (ADAMOPOULOU; MOUSSIADES, 2020).

O primeiro *chatbot*, ELIZA, foi desenvolvido entre 1964 e 1966 por Joseph Weizenbaum no laboratório de Inteligência Artificial do MIT, com a tarefa de simular um terapeuta Rogeriano. A psicoterapia Rogeriana encoraja o paciente a falar mais sobre si ao invés de se envolver em uma discussão (NURUZZAMAN; HUSSAIN, 2018). A Figura 1 contém um exemplo de conversação com o chatbot. ELIZA utilizava correspondência de palavras chave simples, apenas respondendo aos usuários de acordo com as entidades identificadas no texto. Por exemplo, se a entrada incluir a palavra-chave "mãe", a saída seria "Como é a sua relação com seus familiares?", regra inspirada na teoria de que mãe e família são fundamentais para problemas psicológicos, então um terapeuta deveria encorajar o paciente a falar mais sobre sua família (SHAWAR; ATWELL, 2007).

Inspirado por ELIZA, Richard S. Wallace desenvolveu entre 1995 e 2002 o *chatbot* *Artificial Linguistic Internet Computer Entity* (ALICE) na Lehigh University. Para apoiar seu desenvolvimento, Richard criou a linguagem de marcação *Artificial Inteligence Markup Language* (*A.I.M.L.*), que expande o conceito de correspondência de padrões permitindo definir blocos para a construção do conhecimento do *chatbot* e que consistem em unidades chamadas de tópicos e categorias. O tópico é um elemento opcional de nível superior que possui um atributo de nome e um conjunto de categorias relacionadas a esse tópico. As categorias são a unidade

Figura 1 – Exemplo de conversa com o *chatbot* ELIZA



Fonte: adaptado de Tzouganatou (2018).

básica de conhecimento em *A.I.M.L.* (SHAWAR; ATWELL, 2007). Cada categoria consiste em uma pergunta de entrada, uma resposta de saída e um contexto opcional. A pergunta ou estímulo é chamada de *pattern*. A resposta ou reação é chamada de *template*. A Figura 2 contém o exemplo de como uma categoria é definida. As combinações de tópicos e categorias são utilizadas em conjunto com as regras para gerar a resposta ao usuário. O *chatbot* ALICE ganhou o Prêmio Loebner, um Teste de Turing anual, nos anos de 2000, 2001 e 2004 e foi o primeiro programa de computador a ganhar o posto de "computador mais humano" (ADAMOPOULOU; MOUSSIADES, 2020).

Diferente de ELIZA e ALICE, o *chatbot* Cleverbot, desenvolvido por Rollo Carpenter em 1986 e publicado na internet em 1997, não possui respostas pré-programadas. Em vez disso, ele aumenta sua base de conhecimento a cada interação com o usuário. Quando o usuário insere uma frase, o Cleverbot encontra todas as palavras-chave ou frases correspondentes à entrada e pesquisa em sua base de dados respostas de usuários reais a entradas semelhantes (NURUZZAMAN; HUSSAIN, 2018). Isto permite que o Cleverbot engaje em conversas mais diversificadas e aprenda com seus usuários, mas que acabe gerando respostas imprevisíveis.

Figura 2 – Exemplo de um arquivo *A.I.M.L.*

```
<?xml version="1.0" encoding="UTF-8"?>

  <aiml version="2.0">
    <category>
      <pattern>HI *</pattern>
      <template>Hello world!</template>
    </category>

    <category>
      <pattern>What is a chatbot</pattern>
      <template>
        A chatbot is a computer program designed to respond
        to text or voice inputs in natural language.
      </template>
    </category>
  </aiml>
```

Fonte: adaptado da AIML Foundation (2020).

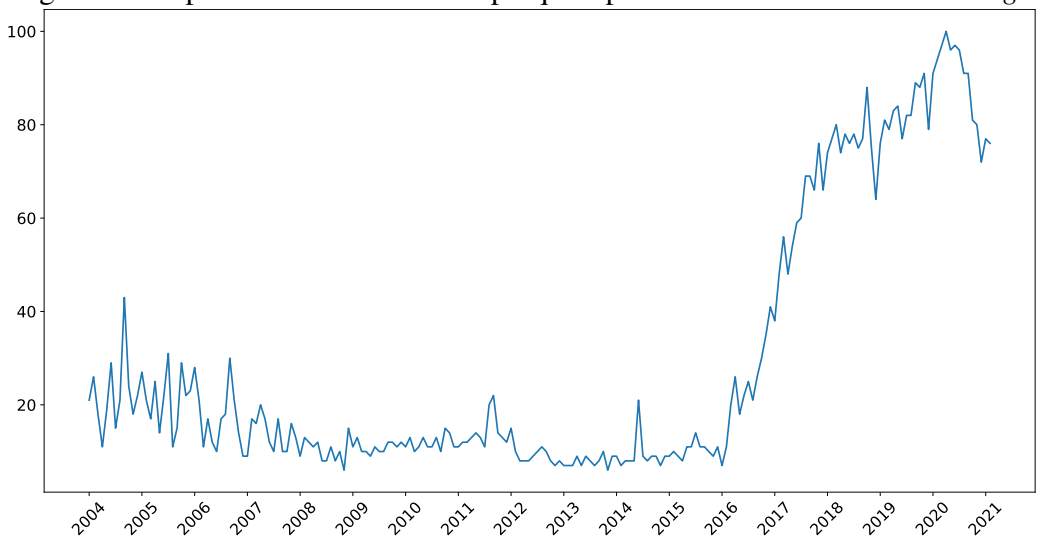
2.1.1 Ascensão dos Chatbots

A disseminação dos *chatbots* no cenário comercial começou a ocorrer em 2016 devido a dois fatores importantes, os avanços na área de IA e a popularização das aplicações móveis de troca de mensagem, como *Facebook Messenger*, *Whatsapp*, *Slack* e *Telegram*. É possível observar na Figura 3 o crescimento pelo interesse no assunto. Naquele ano, o CEO da Microsoft, Satya Nadella, comparou a transição prevista para *chatbots* e interfaces de usuário de linguagem natural a revoluções anteriores, como a introdução da interface gráfica do usuário, a Web e a Internet móvel [1]. O CEO do Facebook, Mark Zuckerberg, proclamou que os *chatbots* são uma solução para o desafio da sobrecarga de aplicativos [2]. Chris Messina, na época Líder de experiência do desenvolvedor da Uber, escreveu um blog declarando 2016 o ano do comércio de conversação [3].

A maioria das implementações dos *chatbots* independe de plataforma e está instantaneamente disponível para os usuários sem instalações necessárias. Os *chatbots* podem ser integrados com conversas em grupo ou compartilhados como qualquer outro contato, enquanto várias conversas podem ser realizadas em paralelo (ADAMOPOULOU; MOUSSIADES, 2020). O uso de Aprendizagem Profunda ou *Deep Learning* (*DL*) permite aos *chatbots* compreender as mensagens do usuário, extraindo sua intenção, entidades e qualquer outra informação estruturada. Assim, é possível decidir qual ação deve tomar de acordo com as informações extraídas, seja

[1] https://www.businessinsider.com/microsoft-ceo-satya-nadella-chatbots-wpc-2016-7
[2] https://www.wsj.com/articles/facebook-hopes-chatbots-can-solve-app-overload-1460930220
[3] https://medium.com/chris-messina/2016-will-be-the-year-of-conversational-commerce-1586e85e3991

Figura 3 – Popularidade de termos de pesquisa pelo assunto *Chatterbot* no *Google*



Fonte: adaptado de Google Trends (2021).

apenas enviar uma mensagem para o usuário ou executar uma função arbitrária (BOCKLISCH *et al.*, 2017).

2.2 Redes Neurais Profundas

Nesta seção será feita uma breve explicação sobre alguns exemplos de arquiteturas de redes neurais profundas e termos importantes utilizados no campo de aprendizado de máquina.

- Treinamento: O treinamento de um modelo é o processo de ajustá-lo a um conjunto de dados de acordo com uma função objetivo. A função objetivo é escolhida a partir do problema tratado.
- Inferência: É o processo de fornecer a um modelo um exemplo no domínio em que este foi treinado, para que o modelo produza uma saída de acordo com o que foi aprendido durante o treinamento.
- *Loss*: O valor calculado pela função objetivo. A função objetivo é utilizada para avaliar uma solução candidata para resolução do problema. Em redes neurais, procuramos minimizar o erro, assim, a função objetivo é função de custo ou uma função de perda.
- *Batch*: É a divisão do conjunto de dados em pequenos pedaços. Estes pedaços são fornecidos um a um a rede até que todo o conjunto de dados tenha passado por esta, o que é chamado de época. A utilização de *batches* é necessária pois muitos conjuntos de dados não cabem por inteiro na memória do computador.
- *Transfer Learning*: A aprendizagem por transferência e a adaptação de domínio referem-se à situação em que o que foi aprendido em um ambiente é explorado para melhorar a

generalização em outro cenário (GOODFELLOW *et al.*, 2014). Esta técnica consiste em reutilizar um modelo base treinado com conhecimento em um domínio e adaptá-lo ao problema específico. Por exemplo, um modelo treinado para identificar cachorros pode ser adaptado para identificar gatos.

2.2.1 Redes Neurais Convolucionais

Rede Neural Convolucional ou *Convolutional Neural Networks* (*CNN*) é um tipo de Rede Neural capaz de aprender padrões e hierarquias espaciais de forma automática e adaptativa utilizando operações de convolução (GOODFELLOW *et al.*, 2014). Uma convolução é uma operação matemática linear entre duas funções ao longo da região em que elas se sobrepõem. Em *CNN*s esta operação ocorre entre uma região da entrada e uma matriz de filtro, que é responsável por extrair um padrão daquela região.

A principal aplicação de *CNN* é no campo de visão computacional. Um modelo capaz de reconhecer objetos pode aplicar várias convoluções em uma imagem, com filtros capazes de reconhecer retas diagonais, horizontais e verticais, semi círculos, espirais, e ao fim utilizar as informações obtidas por todos estes filtros para decidir se a imagem de entrada corresponde a um carro ou a um cachorro.

2.2.2 Codificador-Decodificador com camada de Atenção

Estes modelos são utilizados em tarefas que apresentam entrada e saída sequenciais de tamanhos variados, muitas vezes de domínios diferentes, como reconhecimento de voz, tradução automática ou resposta a perguntas, em que as sequências de entrada e saída no conjunto geralmente não têm o mesmo comprimento (GOODFELLOW *et al.*, 2014).

O codificador é análogo a um modelo de extração de características, transformando a entrada em uma representação intermediária. A camada de atenção atua como um modelo de alinhamento, selecionando partes relevantes da entrada a cada passo. Já o decodificador tem o papel de transformar os dados fornecidos pelo codificador no domínio de saída, selecionando as características de acordo com as informações fornecidas pela camada de atenção.

2.2.3 Redes Geradoras Adversárias

Redes Geradoras Adversárias ou *Generative Adversarial Networks* (*GAN*) são redes treinadas em paralelo de forma competitiva com o objetivo de que uma delas aprenda a gerar dados similares aos dados de treino. Nesta arquitetura, o modelo gerador é treinado para gerar novos exemplos enquanto o modelo discriminador tenta classificar os exemplos como reais (do domínio) ou falsos (gerados). Os dois modelos são treinados juntos em um jogo de soma zero, até que o modelo discriminador seja enganado na metade do tempo, o que significa que o modelo gerador está criando exemplos próximos aos reais (GOODFELLOW *et al.*, 2014).

Cada modelo nesta arquitetura possui uma *loss*, calculada da seguinte forma: o gerador produz alguns exemplos de saída, o discriminador então é apresentado com um conjunto de dados reais, que ele deve classificar como verdadeiros, e os dados criados pelo gerador, que ele deve classificar como falsos. A taxa de erros de classificação total entre os conjuntos real e falso é usada para treinar o discriminador, enquanto os acertos do discriminador no conjunto falso é utilizada para treinar o gerador.

2.3 Sistemas de Reconhecimento de Voz

Sistemas de reconhecimento de voz ou *Automatic Speech Recognition* (*ASR*) são capazes de transcrever um arquivo de áudio em uma sequência de palavras. Durante o reconhecimento do discurso esses sistemas se deparam com diversas dificuldades, como a ausência de limites claros entre fonemas ou palavras, sinais de ruído indesejados do ambiente ao redor do falante e variabilidade do falante, como gênero, estilo de fala, velocidade da fala, fala regional e dialetos (CUTAJAR *et al.*, 2013). Ehsani e Knodt (1998) destaca os quatro componentes de um sistema de reconhecimento de voz automático, sendo estes:

- Pré-processamento: Transforma a fala de sinal para algo legível pelo sistema.
- Extração de características: Extrai uma quantidade predefinida de características do sinal processado.
- Classificação: Responsável por classificar a entrada de acordo com as características extraídas.
- Modelo de Linguagem: Produz representações significativas do sinal de fala.

2.3.1 Sistemas tradicionais de ASR

Desde meados da década de 1980, Componentes Mel Cepstrais ou *Mel Frequency Cepstral Coefficient* (*MFCC*) eram o método de extração de características mais amplamente usado no campo de *ASR*. O *MFCC* tenta imitar o ouvido humano, onde as frequências são resolvidas de forma não linear em todo o espectro de áudio. (CUTAJAR *et al.*, 2013). Este vetor de características gerado pelo *MFCC* consiste de características estáticas obtidas pela análise independente de cada quadro e características dinâmicas a partir das diferenças entre as características estáticas de quadros consecutivos.

Modelo Oculto de Markov ou *Hidden Markov Model* (*HMM*) é a abordagem mais comumente utilizada para o estágio de classificação de um sistema *ASR*. Com o *HMM*, é possível encontrar a probabilidade de que um enunciado de fala foi gerado pela pronúncia de um fonema ou palavra em particular. Consequentemente, a representação mais provável para um enunciado de fala pode ser avaliada a partir de uma série de possibilidades (CUTAJAR *et al.*, 2013). O *HMM* é baseado em três fundamentos: a avaliação da probabilidade de uma sequência de enunciados para um determinado *HMM*, a seleção da melhor sequência de estados do modelo e a modificação dos parâmetros do modelo vencedor correspondente para melhor representação dos enunciados de fala apresentados.

O conhecimento da língua falada é necessário para um sistema de linguagem, a fim de produzir representações significativas do sinal de entrada de voz. Esse conhecimento é implementado na forma de um modelo de linguagem e pode ser dividido em vários níveis (LIDDY, 2001):

- Fonologia: Incorpora o conhecimento sobre os sons linguísticos dentro e através das palavras.
- Morfologia: Trata do significado dos componentes que constituem uma palavra.
- Léxica: Concentra na identificação do significado das palavras individualmente.
- Sintática: Palavras são analisadas no contexto de uma frase para determinar a estrutura gramatical da frase.
- Semântica: Todos os significados possíveis de uma frase são determinados dentro deste nível.
- Discurso: O texto mais longo do que uma frase é considerado.
- Pragmática: Dentro deste nível, o conhecimento do mundo e a compreensão das intenções, planos e objetivos do falante são necessários.

Um sistema de *NLP* pode utilizar um ou mais dos níveis acima, mas não necessariamente todos eles. Os níveis mais baixos de análise linguística, como morfologia e léxica, são aqueles amplamente estudados e implementados (CUTAJAR *et al.*, 2013). Após as fases de pré-processamento, extração de características e classificação, o modelo *ASR* produz mais de uma representação possível do sinal de voz, essas são então analisadas com uma *NLP* e a melhor é escolhida.

2.3.2 Evolução em ASR

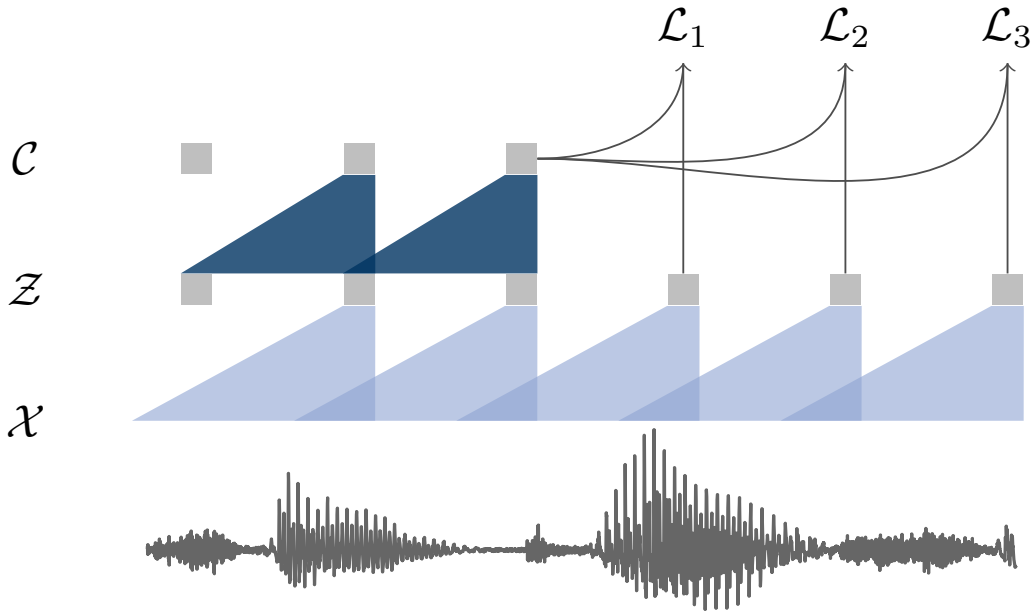
O desenvolvimento da aprendizagem profunda permitiu a criação de sistemas Ponta a Ponta, cujo objetivo é substituir o máximo possível das etapas de problemas de ASR por uma arquitetura de Rede Neural Profunda. A primeira tentativa bem sucedida foi feita por Graves et al. em 2006, que propôs a Classificação Temporal Conexionista ou *Connectionist Temporal Classification* (*CTC*). A *CTC* foi projetada especificamente para tarefas de classificação temporal, ou seja, para problemas de rotulagem de sequência em que o alinhamento entre as entradas e os alvos é desconhecido. Também não requer dados de treinamento pré-segmentados ou pós-processamento externo para extrair a sequência de rótulos da rede (QUINTANILHA, 2017).

O *wav2vec* (SCHNEIDER *et al.*, 2019) utiliza do conceito de pré-treinamento não supervisionado, que busca aprender uma representação geral da fala a partir de exemplos não rotulados. Observando a Figura 4, dado um sinal de áudio X como entrada, o modelo codifica o áudio da fala através de uma *CNN* que tem como saída uma representação de características de baixa frequência Z. Uma segunda rede de contexto, também uma *CNN*, busca mapear Z em um vetor de contexto C. Esta tarefa é realizada para intervalos de tempo, produzindo a cada etapa uma *loss* L. Os modelos pré-treinados são ajustados para reconhecimento de fala, adicionando uma camada de projeção no topo da rede de contexto a fim de prever uma das classes que representam o vocabulário da tarefa, otimizando para minimizar uma perda de *CTC* (BAEVSKI *et al.*, 2020).

2.4 Sistemas de Síntese de Voz

Sistemas de Síntese de Voz ou *Text to Speech* (*TTS*) são capazes de transformar texto em áudio simulando a fala humana. Técnicas tradicionais de *TTS* são compostas de duas etapas principais, análise de texto, em que o texto de entrada é transcrito em representações fonéticas

Figura 4 – Arquitetura do modelo wav2vec



Fonte: adaptado de (SCHNEIDER *et al.*, 2019)

ou linguísticas que representam regras de pronunciação, e processamento acústico, que utiliza a saída da etapa anterior para produzir o sinal de fala.

As técnicas tradicionais de *TTS* apresentam evoluções que podem ser apresentadas em 3 gerações (INDUMATHI; CHANDRA, 2012).

- Técnicas de Primeira Geração incluem Síntese de Formantes e Síntese Articulatória. Requerem uma descrição bastante detalhada e de baixo nível do que deve ser falado.
- Técnicas de Segunda Geração incorporam Síntese Concatenativa e Síntese Sinusoidal. Usam uma abordagem baseada em dados para aumentar a qualidade da fala, reduzindo o tempo de modelagem.
- Técnicas de Terceira Geração incluem *HMM*, a Síntese por Seleção de unidades e o uso de técnicas estatísticas de aprendizado de máquina para inferir o mapeamento de parâmetros a partir dos dados.

2.4.1 Técnicas tradicionais de TTS

O modelo de Síntese de Formantes trabalha no domínio da frequência buscando simular o trato vocal reconstruindo o formato das características a serem reproduzidas. Esta técnica não utiliza exemplos de discurso humano, mas de parâmetros definidos por regras linguísticas. A outra técnica da primeira geração é a Síntese Articulatória, na qual a fala é gerada por meio da modelagem direta do comportamento do articulador humano, utilizando pequenos

tubos auto controlados com os mais diversos parâmetros simulando o sistema vocal humano.

O sistema Concatenativo produz a fala pela concatenação de pequenas unidades pré-gravadas da fala, como fonemas, difonemas e trifonemas para construir o enunciado (TABET; BOUGHAZI, 2011). A base de dados da qual o sistema constrói o discurso deve ser construída a partir da gravação destas unidades, a fim de refletir as características fonológicas da linguagem alvo, com os tamanhos das unidades afetando a qualidade do áudio gerado. A evolução do sistema concatenativo veio na forma do sistema de Seleção de Unidades, que busca lidar com o problema de gerenciar uma grande quantidade de unidades e reduzir as distorções causadas pela concatenação. Nesta técnica, múltiplas instâncias de cada unidade fonética são armazenadas com prosódias variadas, ou seja, as mesmas unidades fonéticas são gravadas com diferentes entonações, ritmo e acentuação. Com isso, é necessário um algorítimo de seleção de unidade, para escolher a melhor unidade de acordo com a sentença a ser sintetizada (TABET; BOUGHAZI, 2011).

A desvantagem dos sistemas concatenativos é que estes são limitados a recriar as unidades fonéticas gravadas. A síntese Sinusoidal usa um modelo harmônico paramétrico e decompõe cada quadro em um conjunto de harmônicos de uma frequência fundamental estimada. A ideia base desta técnica é classificar janelas do texto em porções de harmônicas e ruído, assim como a contribuição de cada janela para o discurso. A síntese por *HMM* busca o treinando de um modelo estatístico paramétrico utilizando características a partir de *MFCC* para inferir o discurso. A síntese *HMM* fornece um meio pelo qual treinar a especificação para o módulo de parâmetro automaticamente, e geralmente consiste em três estados que representam o início, o meio e o fim do fonema (INDUMATHI; CHANDRA, 2012). A fase de síntese consiste em duas etapas: em primeiro lugar, os vetores de recursos para uma determinada sequência de fonemas devem ser estimados. Em segundo lugar, um filtro é implementado para transformar esses vetores de recursos em sinais de áudio. A qualidade da fala gerada pelo *HMM* não é tão boa quanto a qualidade da fala gerada pela síntese da seleção da unidade (TABET; BOUGHAZI, 2011).

2.4.2 Nova Geração em TTS

Assim como no problema de *ASR*, as técnicas tradicionais de *TTS* requerem grande conhecimento linguístico para modelar e selecionar seus parâmeros, e em parte pelos mesmos motivos, o surgimento de modelos de Aprendizado Profundo trouxeram uma enorme evolução também no campo de *TTS*.

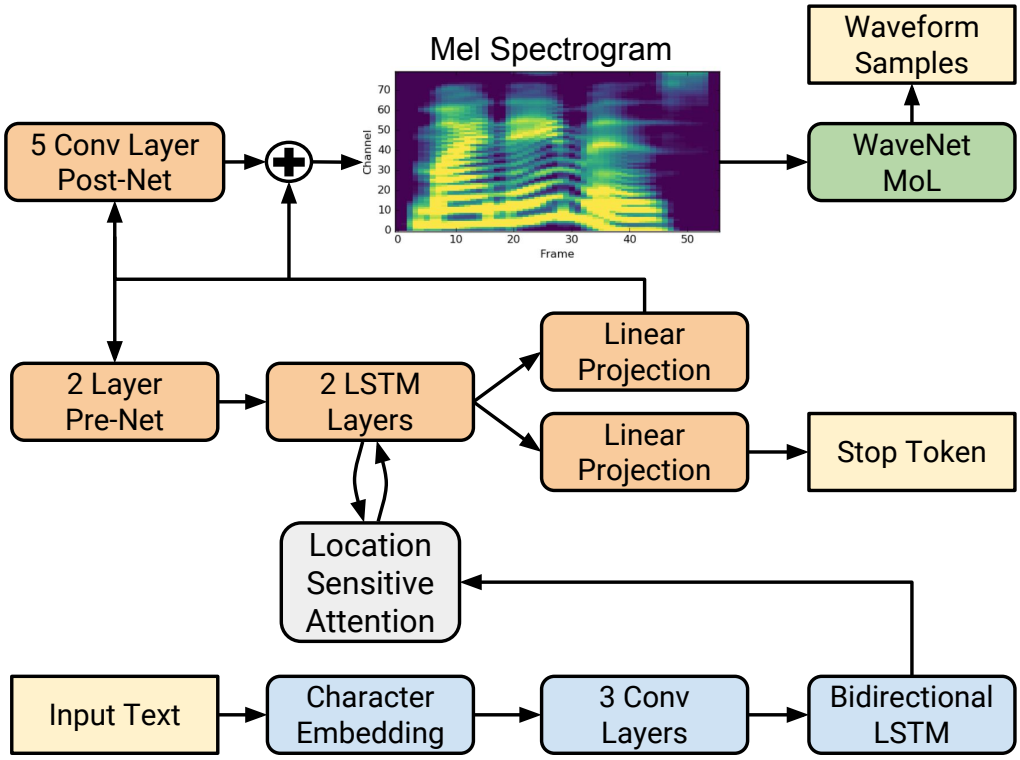
Os modelos de Aprendizado Profundo permitem integrar todas as etapas de processamento em um único modelo e conectá-los diretamente do texto de entrada à saída de áudio sintetizada utilizando aprendizado Ponta a Ponta, e demonstrou ser capaz de estimar espectrogramas a partir do texto de entrada com desempenhos promissores (CASANOVA *et al.*, 2020). Um espectrograma é uma representação visual do espectro de frequências de um sinal, capaz de representar a frequência e amplitude de um sinal de áudio conforme este varia com o tempo.

Estes sistemas podem ser treinados em pares texto-áudio, o que reduz a necessidade de seleção e criação de características a partir do áudio. Estes modelos também permitem mais facilmente um condicionamento rico em vários atributos, como falante ou idioma, ou recursos de alto nível, como sentimento. Isso ocorre porque o condicionamento pode ocorrer bem no início do modelo, e não apenas em certos componentes. Da mesma forma, a adaptação a novos dados também pode ser mais fácil (WANG *et al.*, 2017).

O *WaveNet*, um Modelo Gerador de Formas de Onda (*Vocoder*), produz qualidade de áudio que começa a rivalizar com a fala humana real e já é usado em alguns sistemas *TTS* completos (SHEN *et al.*, 2018), com ouvintes humanos classificando-o como um som significativamente mais natural do que os melhores sistemas paramétricos e concatenativos (OORD *et al.*, 2016). A desvantagem do *WaveNet* é seu extenso tempo de treinamento e inferência, mas algumas alternativas surgiram para superar estes problemas, como *WaveRNN* (KALCHBRENNER *et al.*, 2018), *Parallel WaveGAN* (YAMAMOTO *et al.*, 2020) e *Multi-band MelGAN* (YANG *et al.*, 2021).

O modelo *Tacotron 2* (SHEN *et al.*, 2018), apresentado na Figura 5, utiliza a arquitetura codificador-decodificador com atenção em combinação com um *Vocoder* para produzir apenas com uma entrada de texto um arquivo de áudio *waveform* com a sentença sintetizada. O codificador (azul), atua em uma sequência de entrada, extraindo o contexto de uma representação vetorial dos caracteres, que é interpretada pela camada de atenção (cinza). O decodificador (laranja) projeta um quadro do espectrograma a partir da saída da camada de atenção, e este quadro previsto é alimentado a uma rede neural convolucional responsável por reconstruir o espectrograma que representa a sequência de entrada. Por fim, o *Vocoder* (verde) analisa o espectrograma para produzir um arquivo *waveform*.

Figura 5 – Arquitetura Tacotron 2



Fonte: adaptado de Shen *et al.* (2018)

2.5 Métricas de Avaliação

Neste trabalho, é necessário avaliar as duas etapas de forma separada. Para o *ASR*, é possível utilizar métricas para calcular a similaridade de texto, utilizando estas métricas para comparar o texto original do discurso com a transcrição obtida. A avaliação de *TTS* é um pouco mais complicada devido a seu domínio final, assim, é necessária uma análise subjetiva dos arquivos de áudio. No entanto, a comparação de seus espectrogramas pode permitir uma análise objetiva.

2.5.1 Avaliação de Reconhecimento de Voz

Para esta etapa, foram escolhidas métricas que calculam não apenas a diferença entre duas sentenças, com a Taxa de Erro de Palavras ou *Word Error Rate* (*WER*) e Distância de Jaccard, mas também métricas que avaliem seus contextos, tanto na ordem e posicionamento das palavras, com Avaliação Bilíngue de Substituição ou *Bilingual Evaluation Understudy* (*BLEU*) e Métrica para Avaliação de Tradução com Ordenação Explícita ou *Metric for Evaluation of Translation with Explicit ORdering* (*METEOR*), quanto em suas similaridades, com a Similaridade Cosseno.

2.5.1.1 Taxa de Erro de Palavras

WER (HUNT, 1990) é a métrica mais comum de desempenho de um sistema de reconhecimento de voz ou tradução automática. O *WER* é derivado da distância de Levenshtein, trabalhando no nível da palavra ao invés do nível de caracteres. Para calcular o *WER*, as sentença de referência e a transcrição são alinhadas, e as seguintes informações são computadas:

- O número de substituições (S)
- O número de remoções (R)
- O número de inserções (I)
- O número de palavras corretas (C)

Com isso, a Equação 2.1 é calculada. Um *WER* mais baixo geralmente indica que o software *ASR* é mais preciso no reconhecimento de fala. Consequentemente, um *WER* alto indica menor precisão do *ASR*.

$$WER = \frac{S+R+I}{S+R+C} \tag{2.1}$$

2.5.1.2 Distância de Jaccard

O índice de Jaccard, também conhecido como coeficiente de similaridade de Jaccard (JACCARD, 1912), é uma estatística usada para entender as semelhanças entre conjuntos de amostras. A medição enfatiza a similaridade entre conjuntos de amostra finitos e é definida como o tamanho da interseção dividido pelo tamanho da união dos conjuntos de amostra A e B, como demonstrado na Equação 2.2:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{2.2}$$

A distância de Jaccard mede a dissimilaridade entre os conjuntos de amostras. A distância de Jaccard, descrita na Equação 2.3, é calculada encontrando o índice de Jaccard e subtraindo-o de 1.

$$1 - J(A,B) \tag{2.3}$$

2.5.1.3 BLEU e METEOR

As métricas *BLEU* (PAPINENI *et al.*, 2002) e *METEOR* (LAVIE; AGARWAL, 2007) são utilizadas para avaliar a qualidade da tradução automática entre dois idiomas. Nestas

métricas o texto traduzido é comparado com referências de tradução e suas similaridades são calculadas para chegar a uma estimativa da qualidade geral da tradução.

O *BLEU* leva em consideração a precisão modificada de n-gramas, combinando o resultado na média geométrica (VELA *et al.*, 2014). Um n-grama é uma sequência de n itens em um texto. Criando n-gramas de tamanho 2 a nível de palavra para o texto 'O livro está na mesa', temos: `O livro`, `livro está`, `está na` , `na mesa`.

Pela Equação 2.4, p_n é a precisão modificada para n-grama e w_n é o peso entre 0 e 1 para $\log p_n$. *BP* representa a pena de brevidade, para penalizar traduções curtas, representada na Equação 2.5, onde c é o comprimento em todas as traduções candidatas e r é o melhor comprimento de correspondência para cada tradução candidata.

$$\text{BLEU} = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{2.4}$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ \exp\left(1 - \frac{r}{c}\right) & \text{if } c \leq r \end{cases} \tag{2.5}$$

O *METEOR* avalia uma tradução candidata calculando a precisão e revocação no nível do radical de uni-gramas e combinando-os em uma média harmônica parametrizada (VELA *et al.*, 2014). A precisão P é calculada pela Equação 2.6 e a revocação R pela Equação 2.7, onde m é o número de uni-gramas na tradução candidata que também são encontrados na tradução de referência, w_c é o número de uni-gramas na tradução candidata e w_r é o número de uni-gramas na tradução referência.

$$P = \frac{m}{w_c} \tag{2.6}$$

$$R = \frac{m}{w_r} \tag{2.7}$$

Os valores da precisão e revocação são combinados na média harmônia, com um peso maior atribuído a revocação. A Equação 2.8 apresenta este cálculo. Para penalizar uni-gramas que aparecem fora de sequência, este uni-gramas são agrupados no menor número de segmentos possível, com cada segmento formado por uni-gramas adjacentes na tradução e referência. A

partir disto, a Equação 2.9 é calculada, onde c é a quantidade de segmentos e u_m é a quantidade de uni-gramas. Por fim, o valor final é calculado a partir da Equação 2.10.

$$F = \frac{10PR}{R+9P} \tag{2.8}$$

$$p = 0.5\left(\frac{c}{u_m}\right)^3 \tag{2.9}$$

$$\text{METEOR} = F \cdot (1-p) \tag{2.10}$$

2.5.1.4 Similaridade Cosseno

A Similaridade Cosseno é o cosseno do ângulo entre dois vetores. Na análise de texto, cada vetor pode representar um documento. Para representar as sentenças em um espaço vetorial são utilizados *Word Embeddings*, onde palavras ou frases do vocabulário são mapeadas para vetores de números reais (SITIKHU *et al.*, 2019). Conceitualmente, envolve uma incorporação matemática de um espaço com muitas dimensões por palavra a um espaço vetorial contínuo com uma dimensão muito inferior. A partir dessa representação, o cosseno é calculado entre os vetores a e b, como demonstrado na Equação 2.11. Quanto menor o valor de $\cos\theta$, maior o angulo θ e, portanto, menor a similaridade entre dois documentos.

$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\left\|\vec{b}\right\|} \tag{2.11}$$

2.5.2 Avaliação de Síntese de Voz

Para a avaliação de *TTS*, além da análise subjetiva, utilizando Pontuação Média de Opinião ou *Mean Opinion Score* (*MOS*), foi explorada a possibilidade de aplicar métricas para avaliar a distância entre os espectrogramas dos áudios reais e sintéticos para avaliar a qualidade da voz gerada, utilizando as medidas Similaridade Cosseno e Distorção Temporal Dinâmica ou *Dynamic Time Warping* (*DTW*).

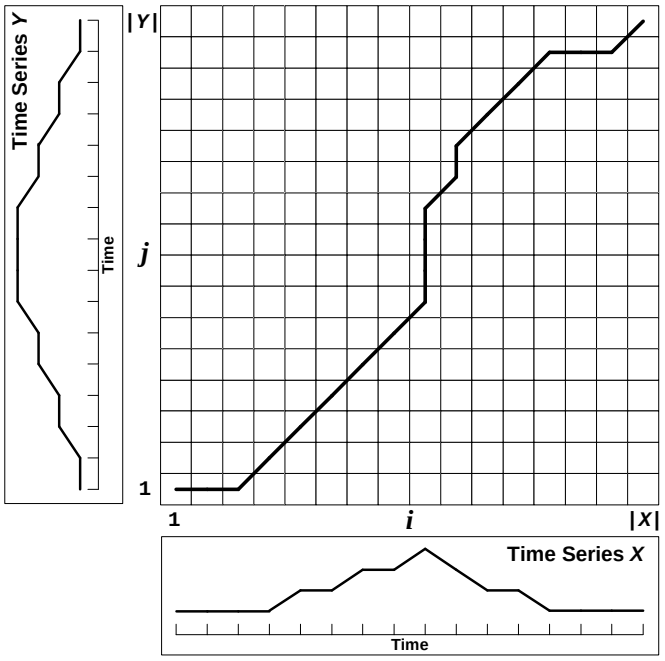
2.5.2.1 Similaridade Cosseno

Uma alternativa a avaliação subjetiva é avaliar as vozes real e sintetizada comparando seus espectrogramas. Assim, a similaridade cosseno pode ser utilizada considerando os espectrogramas como um espaço vetorial. Similar a avaliação realizada na etapa de *ASR*, aplicar a medida cosseno entre dois espectrogramas pode nos informar a similaridade entre estes.

2.5.2.2 Distorção Temporal Dinâmica

DTW é uma métrica para a análise de series temporais, que é utilizada para encontrar o alinhamento não-linear ótimo entre duas sequências de valores numéricos. Como um espectrograma é uma representação da frequência e amplitude no tempo, podemos utilizar do mesmo conceito para comparar suas distancias. Para calcular a *DTW* entre duas séries temporais X e Y é construída uma matriz de custo D, como demonstrada na Figura 6. Os eixos de D representam o tempo nas séries, o valor em $D(i, j)$ é o custo mínimo do caminho de distorção que pode ser construído a partir das duas séries naquele instante, e o valor em $D(|X|, |Y|)$ conterá o custo mínimo para o alinhamento entre as séries X e Y (SALVADOR; CHAN, 2007). Portando, um menor *DTW* significa uma maior similaridade entre os eventos.

Figura 6 – Matriz de custo de alinhamento entre duas séries temporais



Fonte: adaptado de Salvador e Chan (2007)

2.5.2.3 Pontuação Média de Opinião

A *MOS* é a métrica mais comumente usada na avaliação de *TTS*. É um método subjetivo de teste de qualidade, em que humanos julgam a qualidade da transmissão da voz ao ouvirem amostras (RIBEIRO *et al.*, 2011). Após estes testes, os sujeitos irão qualificar a qualidade da voz de acordo com uma escala entre 1 e 5, com as seguintes correspondências:

- 5: Excelente
- 4: Bom
- 3: Razoável
- 2: Pobre
- 1: Ruim

Por ser um teste subjetivo que envolva a avaliação de humanos, pode exigir um consumo de um tempo excessivo para administrar.

3 TRABALHOS RELACIONADOS

Foram identificados estudos com propostas similares ao trabalho proposto. Estes trabalhos são apresentados neste capítulo, enquanto a seção final do capítulo apresenta uma comparação entre os trabalhos relacionados e este trabalho, com base em 6 características específicas.

3.1 *Android Based Educational Chatbot for Visually Impaired People*

Kumar *et al.* (2016) propõem uma aplicação de *chatbot* educacional feita com uma aplicação para a plataforma *Android*. Este aplicativo fornece dois recursos, um *chatbot* no estilo perguntas e respostas desenvolvido em *A.I.M.L.* e uma ferramenta de busca por artigos da *Wikipedia*. O usuário escolhe o recurso que deseja utilizar e dá como entrada texto ou voz. A entrada de voz é transcrita pela classe de reconhecimento de voz no *Android*. A depender do recurso escolhido, a entrada é tratada pela *Application Programming Interface* (*API*) específica.

Para a busca na *Wikipedia* o usuário deve pesquisar com uma única palavra-chave, que é utilizada para gerar um URL enviado à *MediaWiki API* para buscar o conteúdo. Se o usuário mudar o recurso de perguntas e respostas, a biblioteca *AB Library* manipula e entrada e os arquivos *A.I.M.L.* para obter uma resposta. Ao processar o resultado do recurso, a saída será dada como fala pelo sistema de *TTS* do *Android*, que converte o texto em fala.

3.2 *BookBuddy: Turning Digital Materials Into Interactive Foreign Language Lessons Through a Voice Chatbot*

Em Ruan *et al.* (2019) foi desenvolvida um aplicação web para auxiliar no ensino de inglês para crianças. A aplicação é composta de 3 etapas independentes, mas transparentes ao usuário. A primeira etapa é um sistema de recomendação de livros que coleta informações sobre o usuário e suas preferências, e utiliza destas informações para recomendar um livro.

Para auxiliar a criança na leitura do livro um *chatbot* responde perguntas sobre personagens, acontecimentos e sobre vocabulário em geral. Um segundo *chatbot* é responsável por avaliar o entendimento da criança sobre o livro através de um banco de perguntas. Os autores utilizam da *Web Speech API*[1] para o *ASR*. Para um melhor identificação das crianças com a

[1] https://wicg.github.io/speech-api/

aplicação, foi utilizada a voz sintetizada de uma criança, através da plataforma *Acapela Group*[2].

3.3 *Solução Mobile – Voice Assisted Chatbots*

Palma (2019) desenvolveu uma assistente virtual na forma de *chatbot* que suporta comunicação através de voz para auxiliar aos usuários de seus clientes com assistência técnica, marcação de visitas e pagamento de faturas. Para o desenvolvimento do *chatbot* foram utilizadas as ferramentas *Microsoft Bot Framework*[3] para o desenvolvimento e *Azure Bot Service*[4] para hospedagem.

O desenvolvimento foi voltado a uma interface web para dispositivos móveis, e permite ao usuário comunicar com o *chatbot* através de texto e fala. Para tratar do *ASR* foram utilizadas as ferramentas inerentes dos sistemas operacionais *Android* e *iOS*, e para o *TTS* foi utilizada a *Web Speech API*. A comunicação por voz pode ser habilitada e desabilitada durante a iteração.

3.4 Análise Comparativa

Esta seção apresenta uma análise comparativa dos trabalhos descritos neste capítulo. O Quadro 1 sumariza os principais diferenciais entre os trabalhos.

O ‘propósito‘ representa o objetivo geral do chatbot desenvolvido para acompanhar o trabalho. Todos os trabalhos abordam temas diferentes, mas com um fluxo de interação semelhante. Um dos desafios para as tarefas de *ASR* e *TTS* é o ‘idioma‘, que afeta a disponibilidade e qualidade das ferramentas disponíveis.

A ‘plataforma‘ na qual o *chatbot* será utilizado é importante não apenas no desenvolvimento da interface mas também na escolha de ferramentas para executar as tarefas de reconhecimento e síntese de Voz. Um grande diferencial deste trabalho é que diferentemente dos trabalhos apresentados, que foram desenvolvidos com uma única plataforma em específico, este trabalho foi desenvolvido para ser conectado não apenas dispositivos *Web* e *Mobile*, mas também as populares aplicações de troca de mensagem.

As características de ‘reconhecimento de voz‘ e ‘síntese de voz‘ levam em conta a plataforma em que serão utilizadas assim como seus propósitos. Kumar *et al.* (2016) e Palma

[2] https://www.acapela-group.com/
[3] https://dev.botframework.com/
[4] https://azure.microsoft.com/pt-br/services/bot-service/

(2019) utilizaram de ferramentas inerentes de suas plataformas. Ruan *et al.* (2019) levou em consideração o propósito de seu *chatbot* na escolha da ferramenta de *TTS*. Por ser multiplataforma, este trabalho utilizou de uma *API* externa para a realização da tarefa de *ASR*, e o uso de um modelo customizado para *TTS* permite o uso de uma voz específica para o domínio.

Por fim, o 'controle da forma de interação' indica como será o fluxo de interação com o *chatbot* a depender da entrada do usuário. Kumar *et al.* (2016) e Ruan *et al.* (2019) apresentam a saída sempre em voz, o que pode levar a uma frustração do usuário que não deseja utilizar esta opção. Já Palma (2019) permite o controle da forma de saída a partir de comandos do usuário. Este trabalho assume que o usuário espera uma resposta do mesmo tipo de sua entrada, sendo assim a saída de voz só ocorre quando o usuário comunica com o *chatbot* através de voz.

Quadro 1 – Comparação com trabalhos relacionados

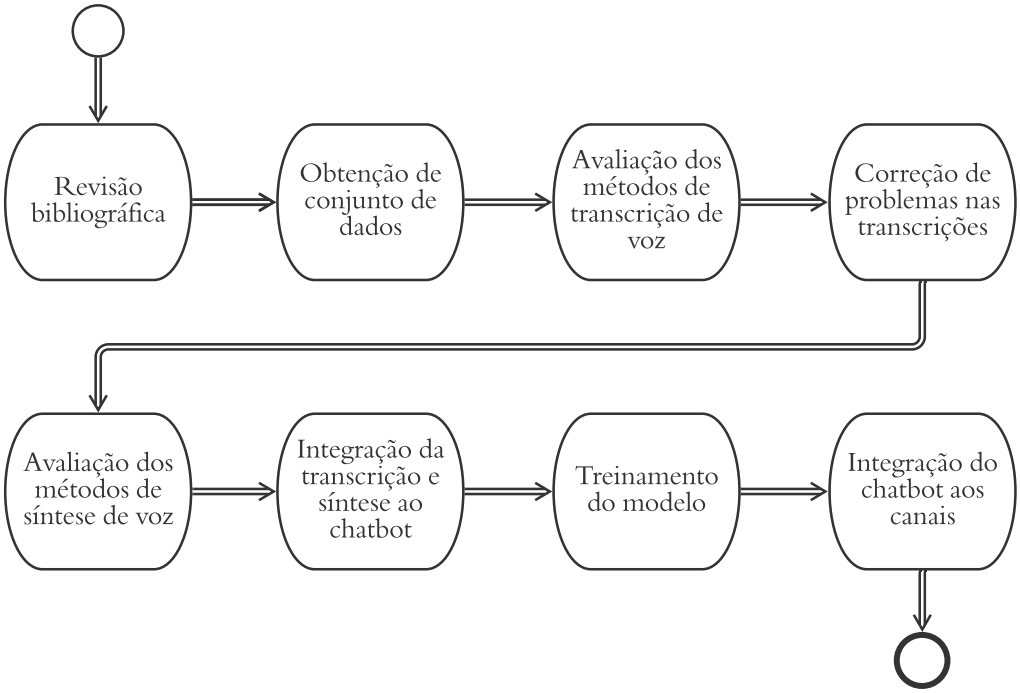
Características	**Kumar *et al.* (2016)**	**Ruan *et al.* (2019)**	**Palma (2019)**	**Este Trabalho**
Propósito	Perguntas e respostas geral e Busca na Wikipédia	Acompanhante de leitura virtual para crianças	Assistência técnica e auxílio pagamento de faturas	Perguntas frequentes da Carta de Serviços do Governo do Estado do Ceará
Idioma	Inglês	Inglês	Portugês de Portugal	Português do Brasil
Plataforma	Aplicativo *Android*	Aplicação *Web*	Aplicação *Web Mobile*	Aplicação *Mobile*, *Web* e de troca de mensagem
Reconhecimento de Voz (*ASR*)	*Android Speech Recognizer*	*Web Speech API*	*Android Speech Recognizer* e *iOS Speech Framework*	Wit.ai
Síntese de Voz (*TTS*)	*Android Text to Speech*	*Acapela Group*	*Web Speech API*	Modelos *Tacotron 2* e *Multi-Band MelGAN* treinados em *dataset* próprio
Controle da Forma de Interação	Saída sempre em voz	Saída sempre em voz	Possível desabilitar saída em voz	Saída de acordo com a entrada do usuário

Fonte: elaborado pelo autor

4 METODOLOGIA

Neste capítulo são apresentados os passos a serem realizados para execução deste trabalho. A Figura 7 apresenta o fluxo dos passos da metodologia. Informações detalhadas sobre cada etapa são descritas nas próximas seções deste capítulo.

Figura 7 – Fluxo de atividades



Fonte: elaborado pelo autor

4.1 Revisão bibliográfica sobre métodos de transcrição e síntese de voz

Nesta etapa são estudados métodos e ferramentas que executam a tarefa *ASR* e *TTS*. É importante avaliar o Estado da Arte destas técnicas e como estes são utilizadas em outros trabalhos e aplicações. É importante também explorar ferramentas e *API*s que executam as tarefas de *ASR* e *TTS*. Para a escolha das ferramentas, além da revisão bibliográfica e do estudo das arquiteturas, devem ser considerados o custo de utilização, capacidade e tempo de execução como critérios.

4.2 Obtenção de conjunto de dados

Para uma avaliação das ferramentas de *ASR* e *TTS* é necessário adquirir um conjunto de dados com uma quantidade expressiva de combinações áudio e texto, em Português do Brasil.

É importante utilizar dados com falas em diferentes contextos, com vocabulário casual ou formal, em ambientes ideais e com ruídos e faladas por pessoas de gênero e idades diversificados.

4.3 Avaliação dos métodos de transcrição de voz

Para avaliar as transcrições, métricas de similaridade de texto serão utilizadas para comparar as transcrições com a sentença real para cada áudio obtido em 4.2. Esta etapa é importante para identificar o desempenho do *ASR* nos diferentes cenários. Tal avaliação permite identificar se a ferramenta varia de desempenho dependendo do vocabulário ditado e também pode ajudar a identificar conjuntos de palavras específicas nas quais encontra mais dificuldade em transcrever.

4.4 Correção de problemas nas transcrições

Com os resultados obtidos em 4.3 são avaliadas maneiras de identificar e corrigir palavras transcritas erroneamente. O vocabulário específico do domínio poderá ser considerado ao identificar e corrigir estas palavras. Para este processo é preciso levar em consideração o tempo e recurso que uma correção levaria e o impacto que palavras transcritas erradas teriam no *chatbot*, pois sua capacidade de interpretar o contexto, com o uso de *NLP*, pode mitigar o impacto destes erros.

4.5 Avaliação dos métodos de síntese de voz

Para a avaliação do *TTS* as possíveis respostas do *chatbot* em conjunto com sentenças generalizadas podem ser utilizadas para a geração dos áudios. Devido a natureza do problema é necessário que os áudios sejam analisados de forma subjetiva pois métricas que permitem comparar este tipo de formato não necessariamente conseguem capturar a naturalidade da fala e sua inteligibilidade. Ainda assim, serão avaliadas alternativas para avaliar a qualidade áudio sintetizado de forma objetiva.

4.6 Integração da transcrição e síntese ao *chatbot*

Com as ferramentas devidamente escolhidas uma arquitetura de aplicação será desenvolvida e acoplada a arquitetura do *chatbot* para que os serviços de *ASR* e *TTS* sejam

integrados. É necessário decidir como lidar com as mensagens recebidas em forma de texto ou voz e como estas serão processadas e repassadas ao chatbot. Também é preciso avaliar o modelo de resposta do *chatbot* para que esta resposta seja devidamente tratada e repassada ao usuário no formato adequado.

4.7 Criação de conjunto de dados e treinamento do modelo

Para proporcionar maior identificação com o *chatbot* é importante que o usuário se sinta familiarizado com suas respostas na forma de voz. Para isso será necessário criar um conjunto de dados que permita o treinamento de um modelo customizado de *TTS*. O estudo realizado em 4.1 e os testes realizados em 4.5 serão importantes para definir a arquitetura de modelo que será utilizada.

A criação deste conjunto de dados deve tomar como exemplo dados já existentes, o que inclui quantidade de dados, uma distribuição semelhante a gaussiana na duração de áudios e tamanho das sentenças, boa cobertura dos fonemas da linguagem, sentenças e áudios sem erros, e gravação dos áudios com naturalidade e sem ruído.

4.8 Integração do *chatbot* a canais

Ao termino do trabalho, pretende-se integrar o *chatbot* a canais de comunicação com o cidadão, como o Portal Ceará Digital e Ceará App, que são plataformas que agregam os serviços digitais oferecidos pelo Governo do Estado do Ceará, e algumas aplicações de troca de mensagem, como *Whatsapp* e *Telegram*. Será necessário analisar como os canais aos quais o *chatbot* será integrado lidam com suas mensagens e como acontece seu suporte a mensagens de voz. A arquitetura da interface de *ASR* e *TTS* do *chatbot* deve ser adaptada para lidar com os diferentes tipos de mensagem e como cada serviço envia e recebe mensagens.

5 EXPERIMENTOS E RESULTADOS

Este capítulo apresenta os resultados obtidos na execução procedimentos metodológicos. Em relação às ferramentas utilizadas, a linguagem de programação *Python* foi empregada para o desenvolvimento de todo o trabalho. Todo o código-fonte está disponível em um repositório no *GitHub* [1].

5.1 Identificação de ferramentas para transcrição e síntese de voz

Durante esta etapa foram identificadas e avaliadas *API*s capazes de executar as tarefas de *ASR* e *TTS*. Foi feita uma avaliação dos custos, modo de cobrança e limites de cada *API*. Para *ASR*, foram identificadas as *API*s *Azure Speech Services*, *Google Cloud Speech-to-Text*, *IBM Watson Speeh to Text*, *Wit.ai* e *Amazon Transcribe*. A Tabela 1 apresenta uma comparação entre as características destas *API*s. Todas as *API*s apresentaram desempenho na transcrição de áudios similar. Como a *API Wit.ai* não cobra por sua utilização, esta *API* foi escolhida para a realização da tarefa de *ASR*.

Tabela 1 – Comparação de caracteristicas das *APIs* de *ASR*

Serviço	**Preco**	**Cobrança mensal**	**Limites**
Azure Speech Services	USD 0,016/minuto	Por minuto	20 solicitações simultâneas 900 solicitações/minuto, 480 horas de áudio por dia
Google Cloud Speech-to-Text	USD 0,006/15 segundos	Por incrementos de 15 segundos	
IBM Watson Speech to Text	USD 0.02/minuto	Agrega todos os usos para o mês e arredonda para o minuto mais próximo.	Não encontrados
Wit.ai	Gratuito	Não existe	240 solicitações/minuto por usuário, 60 solicitações/minuto por aplicação 100 solicitações/segundo, 1 milhão de minutos por mês
Amazon Transcribe	USD 0,0004/segundo	Por segundo, minimo 15 segundos	

Fonte: elaborado pelo autor

Para a etapa de *TTS*, as seguintes *API*s foram avaliadas: *Azure Speech Services*, *Google Cloud Text-to-Speech*, *IBM Watson Text to Speech* e *Amazon Polly*. A Tabela 2 apresenta uma comparação entre as características destas *API*s. Apesar da existência destes serviços, foi escolhido explorar modelos de Aprendizado Profundo identificados na revisão bibliográfica, pois isso permite utilizar um *dataset* customizado, proporcionando melhor identificação do usuário com a voz. Os serviços *Azure Speech Services*, *Google Cloud Text-to-Speech* e *Amazon Polly* oferecem a opção de criar uma voz personalizada, no entanto, em abril de 2021, estes recursos ainda estão em fase de testes e com acesso limitado. O Apêndice A apresenta os links para as

[1] http://bit.ly/tcc-xavier-github

*API*s descritas nesta seção.

Tabela 2 – Comparação de caracteristicas das *APIs* de *TTS*

Serviço	Preco	Cobrança mensal	Limites
Azure Speech Services	USD 16,00 / milhão de caracteres	Após 500 mil caracteres	20 solicitações simultâneas
Google Cloud Text-to-Speech	USD 16,00 / milhão de caracteres	Após 1 milhões de caracteres	150 mil caracteres / minuto, 5 mil caracteres / solicitação
IBM Watson Text to Speech	USD 20,00 / milhão de caracteres	Após 10 mil caracteres	8 mil caracteres / solicitação
Amazon Polly	USD 16,00 / milhão de caracteres	Após 5 milhões de caracteres, quando no nível gratuito	8 solicitações / segundo, 6 mil caracteres / solicitação

Fonte: elaborado pelo autor

5.2 Conjuntos de dados para experimentos

Estes são *datasets* disponíveis abertamente, compostos por sentenças em áudio e seus respectivos textos, o que permite a análise da qualidade de sistemas de *ASR*, transcrevendo os áudios e comparando com seu texto original. Como os dados foram retirados de diferentes fontes, foi necessário padronizar sua estrutura. Todos os arquivos de áudio foram convertidos para o formato `wav`, com frequência `48000Hz`. Para cada corpus foi criada uma pasta para onde os arquivos de áudio foram movidos. Um arquivo `sentences.txt` foi criado para armazenar em cada linha o par `caminho_para_arquivo,sentença`. O Apêndice B apresenta uma visualização da estrutura de organização do conjuntos de dados para teste de *ASR*.

O *Mozilla Common Voice*[2] é um *dataset* em que qualquer pessoa pode contribuir tanto com seu áudio como avaliando a qualidade dos dados e apresenta 5.671 horas de áudio validadas em 54 idiomas. O *dataset* em português possui 48 horas de áudio validadas com 744 vozes diferentes distribuídas da seguinte forma: 37% entre 19 e 29 anos, 35% entre 30 e 39 anos, 11% entre 40 e 49 anos, 4% com mais de 50 anos, 3% com menos de 19 anos e 10% com faixa etária não informada, 85% do sexo masculino, 3% do sexo feminino e 12% com sexo não informado. Para os experimentos, foram selecionados 8014 exemplos validados, com as seguintes distribuições: 32% entre 30 e 39 anos, 22% entre 19 e 29 anos, 20% entre 40 e 49 anos, 14% com mais de 50 anos e 12% com menos de 19 anos, 75 % do sexo masculino, 15% do sexo feminino e 10% com sexo não informado.

Assim como o *Mozilla Common Voice*, o conjunto *Voxforge*[3] é alimentado pela contribuição de usuários e possui sentenças retiradas de *audiobooks* de domínio público. Este conjunto possui 4115 arquivos de áudio, mas não apresenta informações de idade e gênero dos

[2] https://commonvoice.mozilla.org/pt
[3] http://www.voxforge.org/pt

Tabela 3 – Características dos conjuntos de dados

Corpus	**Quantidade de sentenças**	**Quantidade de palavras no vocabulário**	**Duração média dos áudios em segundos**	**Tamanho médio das sentenças em caracteres**	**Formato e frequência do arquivo original**
Código de Defesa do Consumidor	252	1997	20.190	274.472	wav 16KHz
Constituição Brasileira	1251	5319	25.743	354.150	wav 16KHz
Voxforge	4115	566	3.681	27.420	wav 48KHz
Mozilla Common Voice	8014	8596	4.417	42.581	mp3 48KHz

Fonte: elaborado pelo autor

contribuidores. A partir de uma análise manual dos nomes dos arquivos, foram identificados 399 contribuidores diferentes, sendo 281 do sexo masculino, 18 do sexo feminino, e 100 com sexo não identificado.

O Grupo FalaBrasil[4] possui repositórios de citações da Constituição Brasileira e Código de Defesa do Consumidor, ambos compostos por um único locutor do sexo masculino e arquivos com aproximadamente 30 segundos de duração cada. Os arquivos referentes a Constituição Brasileira apresentam em torno de 9 horas de áudio, enquanto o Código de Defesa do Consumidor apresentam em torno de 1 hora e 40 minutos.

A Tabela 3 apresenta as características dos dados utilizados. Os conjuntos de dados podem ser separados em dois grupos. O Grupo 1, formado pelo Código de Defesa do Consumidor e Constituição Brasileira, possui áudios mais longos com média de 23 segundos e vocabulário mais formal e restrito, enquanto o Grupo 2, formado por *Voxforge* e *Mozilla Common Voice*, apresenta áudios curtos com média de 4 segundos e vocabulário mais livre e informal.

Para os testes de *TTS*, foram extraídas 100 sentenças diferentes, de 7 serviços distintos da Carta de Serviços do Estado do Ceará. O tamanho das sentenças possui em média 132 caracteres com tamanho mínimo de 4, máximo de 1019 e desvio padrão de 149 caracteres.

[4] https://ufpafalaBrasil.gitlab.io/

O trabalho de Casanova *et al.* (2020) consiste na criação de recursos publicamente disponíveis para Português do Brasil na forma de um conjunto de dados e modelos para síntese de voz[5]. O *dataset* consiste de 10.5 horas de áudio por um único locutor. Os arquivos de áudio totalizam 3626 exemplos, apresentam duração média de 10 segundos, desvio padrão de 6.5 segundos e durações mínimas e máxima de 1 e 50 segundos. Dos modelos apresentados no trabalho, foram selecionados o modelo *Tacotron 1* com *vocoders Griffin-Lim* e *WaveRNN*, e o modelo *DCTTS*, ambos utilizando transcrição de fonemas, redução de ruído e transferência de conhecimento.

5.3 Experimentos *ASR*

Para os experimentos realizados nessa seção, foi utilizada a plataforma *Google Colab*[6]. O Quadro 2 apresenta as características da máquina utilizada. O primeiro passo para a análise de *ASR* é transcrever os textos. Os 4 conjuntos de dados utilizados, descritos na Seção 5.2, apresentam um total de 13632 sentenças e 25 horas de áudio. A *API Wit.ai* possui limitações referentes a quantidade de solicitações por minuto, como apresentado na Tabela 1, assim, o trabalho de transcrição foi paralelizado em 4 *threads*, com cada *thread* limitada a enviar 1 solicitação por segundo. O tempo total de execução para todas as transcrições foi de 8 horas e 2 minutos.

Após o termino das transcrições, os resultados foram comparados utilizando as métricas *WER*, Distância de Jaccard, *BLEU*, *METEOR* e Similaridade Cosseno, representados na Tabela 4. Esta é uma legenda utilizada para identificar cada *dataset* na tabela:

- Código de Desefa do Consumidor: cod_def_cons
- Constituição Brasileira: constituicao
- Voxforge: voxforge
- Mozilla Common Voice: mozilla
- Resultado utilizando a substituição de palavras erradas: sufixo _oov

Para a métrica de Similaridade Cosseno, foram utilizados os vetores de *Word Embedding* produzido no trabalho de Hartmann *et al.* (2017), que está disponível no Repositório de Word Embeddings do NILC[7]. Foram escolhidos os modelos `Word2Vec` e `Wang2Vec` de 50 dimensões, nas variações *Continuous Bag of Words* (CBOW) e *Skip-Gram*. Como os modelos de *Word Embeddings* foram treinados utilizando vocabulários em português, este vocabulário

[5] https://github.com/Edresson/TTS-Portuguese-Corpus
[6] https://colab.research.google.com/
[7] http://www.nilc.icmc.usp.br/embeddings

foi utilizado para substituir palavras transcritas de forma incorreta, adotando a estratégia de substituir uma palavra transcrita incorretamente pela palavra que possui a menor distância de Levenshtein no vocabulário do *Word Embedding*. As métricas foram novamente calculadas a fim de comparar o impacto desta estratégia de substituição nos resultados. O tempo de execução para o calculo de todas as métricas em todo o corpus foi de 6 horas e 21 minutos.

Quadro 2 – Máquina utilizada para testes de *ASR*

Sistema Operacional	Ubuntu 18.04.5 LTS x86_64
Modelo *CPU*	Intel(R) Xeon(R) CPU
***CPU* GHz**	2.2
Número de *Cores*	1
Número de *Threads*	2
Tamanho da memória *RAM* do Sistema	13021MiB

Fonte: elaborado pelo autor

Foi possível observar que a divisão entre as características dos dados se mostra novamente nos resultados, com o Grupo 1 obtendo valores melhores que o Grupo 2 em todas as métricas. Isso se deve a variedade de vocabulário menor apresentada nos dados do Grupo 1. Apesar dessa diferença, os 4 corpus obtiveram bons resultados. Considerando que as técnicas Estado da Arte em *ASR* apresentam um *WER* em torno de 8.5% para a lingua inglesa, os resultados obtidos foram bastante satisfatórios. Os valores apresentados pelas métricas *BLEU* e *METEOR* demonstram uma boa tradução do texto transcrito em relação ao original, mesmo considerando as sequencias das palavras. Por fim, os resultados obtidos para os *Word Embeddings* demonstam uma grande similaridade entre as sentenças originais e suas transcrições. Como é possível observar, substituir as palavras transcritas de forma errada pelas similares no vocabulário dos *Word Embeddings* não representou diferenças significativas no resultado final para nenhum dos conjuntos de dados.

Tabela 4 – Métricas de avaliação por Conjunto de Dados

Corpus	WER	Jaccard Distance	BLEU	METEOR	Word2Vec CBOW	Word2Vec Skip	Wang2Vec CBOW	Wang2Vec Skip
cod_def_cons	**0.0660**	0.0906	0.8892	0.9438	0.9702	0.9757	0.9716	0.9753
cod_def_cons_oov	0.0682	**0.0936**	0.8849	0.9414	0.9691	0.9748	0.9706	0.9744
constituicao	0.0632	0.0851	**0.8982**	**0.9482**	0.9755	0.9811	**0.9768**	**0.9806**
constituicao_oov	0.0641	0.0853	0.8984	0.9480	**0.9758**	**0.9813**	0.9761	0.9805
voxforge	0.1146	0.1396	0.7653	0.8722	0.9191	0.9201	0.9213	0.9235
voxforge_oov	0.1154	0.1397	0.7655	0.8721	0.9194	0.9200	0.9211	0.9233
mozilla	0.1229	0.1546	0.7724	0.8805	0.9109	0.9139	0.9127	0.9184
mozilla_oov	0.1226	0.1548	0.7723	0.8805	0.9109	0.9137	0.9128	0.9182

Fonte: elaborado pelo autor

5.4 Experimentos *TTS*

Como citado na Seção 5.2, a avaliação do *TTS* foi realizada utilizando o *dataset* e modelos disponibilizados no trabalho de Casanova *et al.* (2020), assim como algumas frases da Carta de Serviços do Estado do Ceará, para avaliar o desempenho da síntese de voz no domínio para qual esta será utilizada.

No total, foram sintetizadas 120 sentenças, 100 da Carta de Serviços e as 20 sentenças utilizadas por Casanova *et al.* (2020) para a avaliação *MOS* dos modelos apresentadas em seu trabalho, somando 15 minutos de áudio para cada um dos 3 modelos, com estes áudios apresentando duração média de 7.5 segundos, desvio padrão de 9 segundos, e durações máxima e mínima de 69 segundos e 0.5 segundo. Para a geração dos áudios por inferência dos modelos, foi utilizada novamente a plataforma *Google Colab*, mas para estes experimentos foi utilizado o ambiênte com *GPU*. O Quadro 3 apresenta as especificações da máquina. Para cada modelo foi realizada a inferência utilizando *CPU* e *GPU*, a fim de calcular os recursos utilizados. A Tabela 5 apresenta a comparação de uso de recursos e tempo de inferência de cada modelo de acordo com o dispositivo.

Tabela 5 – Uso de recursos e tempo de inferência dos modelos

Modelo	**Uso de memória**		**Tempo de inferência / 10 segundos de áudio**	
	CPU	**GPU**	**CPU**	**GPU**
Tacotron 1 Griffin-Lim	380 MiB	1800 MiB	20 segundos	15 segundos
Tacotron 1 WaveRNN	340 MiB	1780 MiB	200 segundos	45 segundos
DCTTS	490 MiB	1260 MiB	84 segundos	12 segundos

Fonte: elaborado pelo autor

Quadro 3 – Máquina utilizada para testes de *TTS*

Sistema Operacional	Ubuntu 18.04.5 LTS x86_64
Modelo *CPU*	Intel(R) Xeon(R) CPU
CPU GHz	2.3
Número de *Cores*	1
Número de *Threads*	2
Tamanho da memória RAM do Sistema	13021MiB
Modelo *GPU*	Nvidia Tesla K80
Versão do Driver	418.67
Versão CUDA	10.1
Tamanho da memória *RAM* da *GPU*	11441MiB

Fonte: elaborado pelo autor

Para a comparação entre os espectrogramas, foram utilizadas as métricas de Similaridade Cosseno e *DTW*. Como possuímos os modelos e os dados utilizados para treino e teste, foi possível comparar os espectrogramas dos áudios gerados pelos modelos com o áudio original como uma forma de medir a similaridade destes espectrogramas, e se esta similaridade é traduzida na qualidade da síntese de voz. A Tabela 6 apresenta os resultados obtidos neste teste. Utilizando estas métricas, não foi possível observar uma reflexão entre o valor obtido e a qualidade da voz sintetizada.

Tabela 6 – Comparação de Espectrogramas utilizando DTW e Similaridade Cosseno

Modelo	DTW	Similaridade Cosseno
Tacotron 1 Griffin-Lim	**43221.28**	0.8338
Tacotron 1 WaveRNN	44577.14	0.8163
DCTTS	45087.10	**0.8440**

Fonte: elaborado pelo autor

A partir da observação de todos os arquivos de áudio gerados, o autor chegou as seguintes conclusões, levando em conta o conjunto de dados e modelos utilizados:

- O modelo *DCTTS* (TACHIBANA *et al.*, 2017) apresentou o melhor balanço entre tempo de inferência e qualidade da voz gerada, podendo sintetizar sentenças com até 128 caracteres, mas não foi capaz pronunciar números ou siglas. Sua arquitetura e implementação é a mais complicada, o que gera a maior dificuldade em sua manutenção.
- O modelo *Tacotron 1* (WANG *et al.*, 2017), utilizando o *vocoder Griffin-Lim* apresentou o melhor tempo de inferência, apesar da qualidade da voz sintetizada ser um pouco inferior aos demais, e foi capaz de lidar com números e siglas. O modelo só foi capaz de sintetizar sentenças com até 30 caracteres, o que acaba gerando algumas quebras na tonalidade da voz ao concatenar sentenças mais longas.
- O modelo *Tacotron 1*, utilizando o *vocoder WaveRNN* (KALCHBRENNER *et al.*, 2018) apresentou os resultados mais promissores, com a voz sintetizada mais natural entre os modelos, sendo capaz de sintetizar sentenças com até 200 caracteres e de lidar com números e siglas. Apesar de produzir a melhor qualidade de voz, o modelo produz alguns artefatos nos áudios, como ruídos, longos trechos em silêncio ou travando e repetindo fonemas. Seu tempo de inferência também é o maior entre os 3, que se dá devido ao *vocoder*.

5.5 Integração de *ASR* e *TTS* ao chatbot

O chatbot utilizado neste trabalho foi desenvolvido em outro contexto do projeto, e foi criado utilizando o *Rasa*[8], um framework desenvolvido na linguagem de programação *Python* e que utiliza de Compreensão de Linguagem Natural ou *Natural Language Understanding* (*NLU*) para interpretar a mensagem do usuário e respondê-la de forma adequada. O *chatbot* criado foi baseado na seção da Carteira Nacional de Habilitação (CNH) da Carta de Serviços do Estado do Ceará.

Para realizar a comunicação de voz com o chatbot, foi necessário implementar um canal de comunicação customizado, pois o *Rasa* não é capaz de lidar com este formato de entrada. Antes disso, foi necessário refatorar os códigos para *ASR* e *TTS* utilizados nos experimentos para que possam ser utilizados como simples chamadas de função pela nova interface.

Nesta nova interface, caso um arquivo de áudio seja identificado, a mensagem de voz é repassada ao módulo de *ASR*, que responde com a transcrição, e essa transcrição substitui a mensagem de áudio recebida, assim, o fluxo normal é seguido pelo *chatbot*. Ao receber a resposta do *chatbot*, a interface verifica se a mensagem do usuário foi de áudio, e caso verdade, o texto é enviado ao módulo de *TTS*, que retorna a voz sintetizada. Esta interface de voz será chamada de *Rasa Voice*.

5.6 Criação do conjunto de dados customizado

Para o treinamento dos modelos foi construído um conjunto de dados desenvolvido para o projeto Governo Digital e Plataforma Big Data para acelerar a Transformação Digital do Estado do Ceará. A gravação dos áudios ocorreu em um estúdio a fim de minimizar ruídos externos, durante um período de 2 semanas.

O conjunto de dados final apresenta 1985 exemplos, selecionados do trabalho de Casanova *et al.* (2020), com duração total dos arquivos de áudio de 5 horas e 4 minutos, e apresenta 10990 palavras diferentes nos textos. É possível observar no Quadro 4 uma descrição estatística dos dados. A Figura 8 apresenta uma visualização da distribuição da quantidade de exemplos por tamanho médio dos textos.

Após a análise dos dados, todos os arquivos de áudio foram convertidos para o formato *WAV* com canal único, pois este é o formato de áudio aceito pelas redes neurais

[8] https://rasa.com/docs/

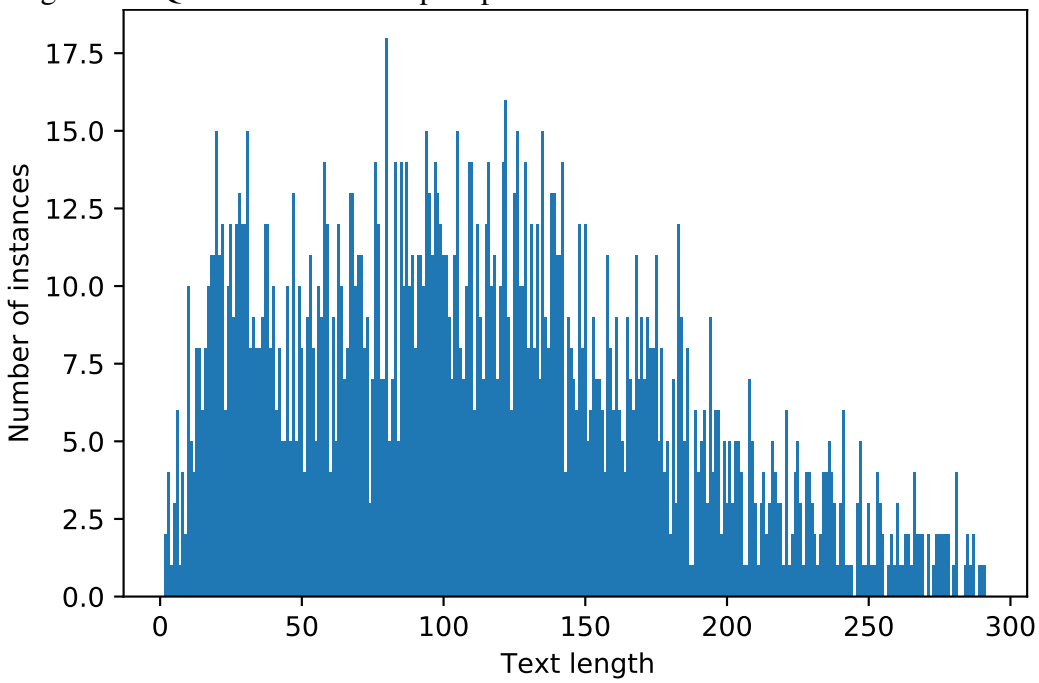
Quadro 4 – Descrição estatística dos dados

	duração dos áudios em segundos	# caracteres nos textos
total	18064.021	237060.000
média	9.100	119.425
desvio padrão	5.676	73.174
mínimo	0.557	2.000
25%	4.736	63.000
50%	8.359	111.000
75%	12.260	163.000
máximo	42.167	432.000

Fonte: elaborado pelo autor

utilizadas, com frequência de 22050Hz, mesma frequência utilizada pelos modelos pre-treinados. O arquivo com os textos foi padronizado para o formato apresentado no *dataset* (ITO; JOHNSON, 2017). Foram separados 1800 exemplos para treino, 145 exemplos para validação e 40 para teste.

Figura 8 – Quantidade de exemplos por tamanho de texto



Fonte: elaborado pelo autor

5.7 Treinamento do modelo

Para o modelo, foram escolhidos o modelo *Tacotron 2* apresentado no trabalho de (SHEN *et al.*, 2018), devido a performance estado da arte, e o modelo *Multi-band MelGAN* apresentado por (YANG *et al.*, 2021) devido a velocidade e baixo recurso utilizado na geração do arquivo de áudio, mesmo utilizando a CPU. A implementação dos modelos foi adaptada do

repositório *Mozilla TTS*[9].

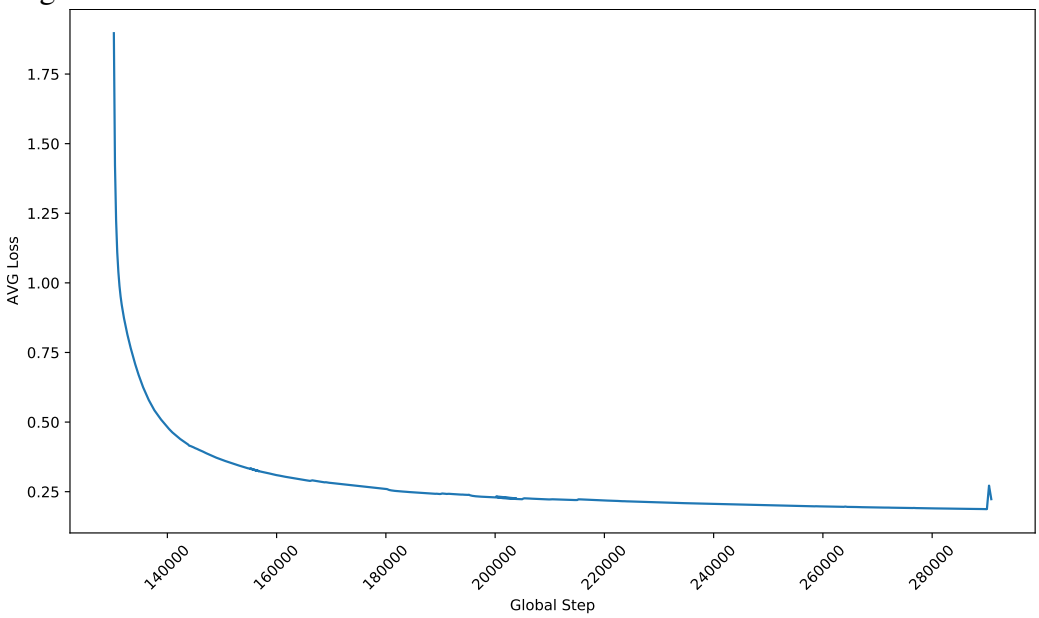
A nível de código, é preciso adaptar alguns arquivos para a língua portuguesa, o que inclui modificar os símbolos para incluir acentuações, adicionar a normalização de números a conversão para português, e excluir da limpeza de texto a remoção de acentos. Para a configuração dos parâmetros dos modelos foram feita modificações específicos para o conjunto de dados, sendo estas, modificar a linguagem dos fonemas para português do Brasil, tamanhos mínimo e máximo dos arquivos de texto, frequência dos arquivos de áudio, e valores de média e desvio padrão dos espectrogramas.

É possível aproveitar da técnica de *transfer learning* para acelerar o treinamento, assim, foram utilizados modelos pre-treinados no conjunto de dados *LJ Speech* (ITO; JOHNSON, 2017), que possui 20 horas de áudio em inglês. O modelo *Tacotron 2* base foi treinado por 130 mil passos, enquanto o *Multi-band MelGAN* foi treinado por 1.45 milhões de passos. Cada passo corresponde a 1 *batch* de dados. Cada modelo utilizou *batches* de tamanho 64 durante seus treinamentos.

Para o treinamento dos modelos, foi utilizada uma máquina com CPU Ryzen 5 3600, GPU Nvidia RTX 2070 Super 8GB VRAM, e 32 GB de memória RAM. O modelo *Tacotron 2* foi treinado por 160 mil passos, utilizando *batch* de tamanho de 32, por um total de 57 horas, enquanto o modelo *Multi-band MelGAN* foi treinado por 350 mil passos, utilizando *batch* de tamanho de 64, por um total de 59 horas. Foi atingido uma utilização máxima de memória da GPU de 6.84 GB durante o treino. A ferramenta *Tensorboard*[10] possibilita o acompanhamento do treinamento, permitindo a visualização da progressão e escutar os áudios gerados. A Figura 9 apresenta a *loss* geral do modelo *Tacotron 2* durante o treinamento, indicando uma evolução no aprendizado até o passo 200 mil, quando o modelo passa a evoluir de forma mais lenta, e a Figura 10 apresenta as *losses* do Discriminador e Gerador no modelo *Multi-band MelGAN* durante o treinamento, que apresenta variações nos valores de perda até uma certa estabilidade a partir do passo 1.6 milhão. Apesar destes gráficos demonstrarem a o aprendizado durante o treinamento, é importante analisar os arquivos de áudio gerados a cada determinada quantidade de épocas para avaliar a evolução dos modelos.

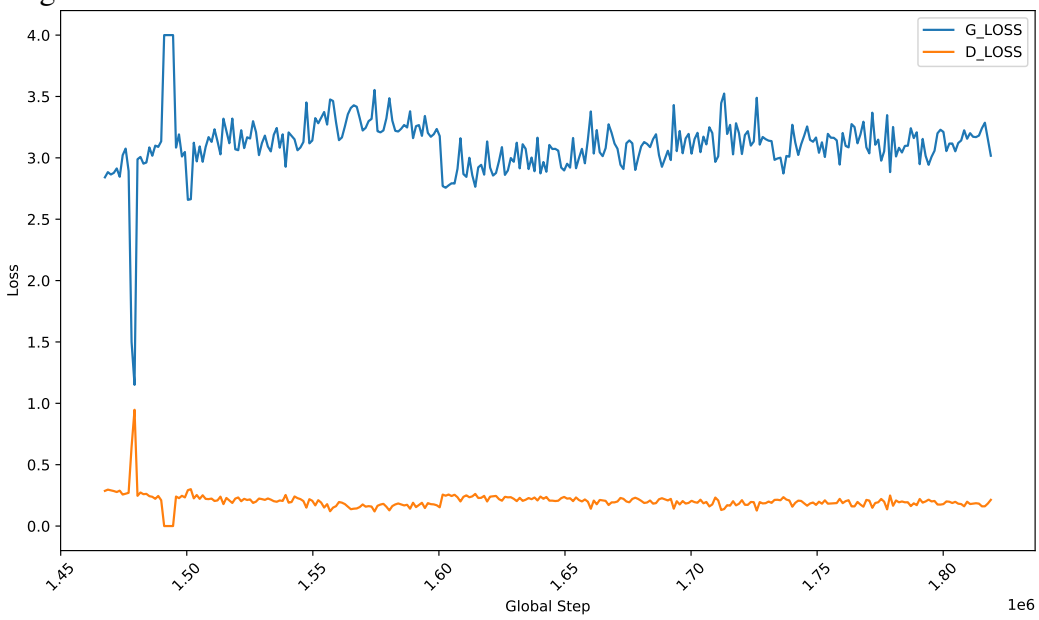
[9] https://github.com/mozilla/TTS
[10] https://www.tensorflow.org/tensorboard?hl=pt-br

Figura 9 – *Tacotron 2 Loss*



Fonte: elaborado pelo autor

Figura 10 – *Multi-band MelGAN Loss*



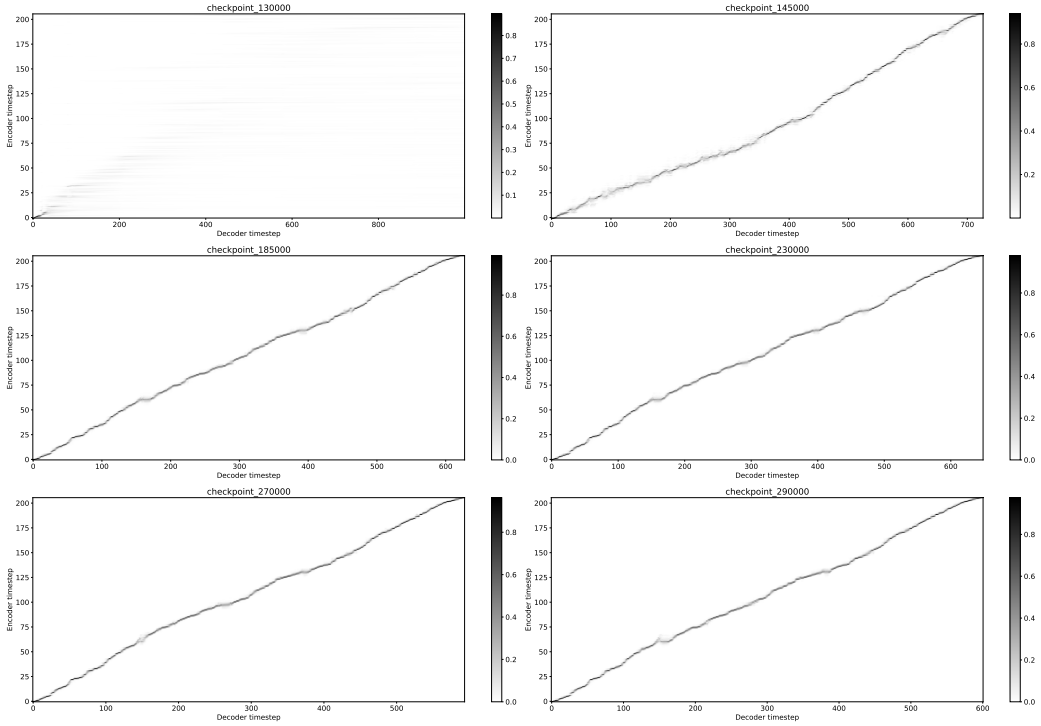
Fonte: elaborado pelo autor

5.8 Performance do modelo treinado

Para avaliar o modelo *Tacotron 2*, podemos observar alinhamento da camada de atenção, que representa o alinhamento entre a entrada de texto e a saída de áudio. O codificador lê os caracteres de entrada passo a passo e gera vetores de estado. O decodificador lê todos os vetores de estado e gera quadros de áudio passo a passo. Um bom alinhamento significa que um som "A"gerado pelo decodificador deve ser o resultado do foco no vetor gerado pelo codificador a partir da leitura do caractere "A". Uma linha diagonal é o resultado de quando os quadros de

áudio são criados focalizando os caracteres de entrada corretos na ordem. A Figura 11 apresenta o alinhamento de um exemplo de teste durante o treinamento, demonstrando como o modelo passa a captar melhor a relação entre entrada e saída com o passar do tempo de treino, enquanto a Figura 12 apresenta os espectrogramas correspondentes, mostrando a criação de espectrogramas mais detalhados a cada passo.

Figura 11 – Progressão do alinhamento da camada de atenção



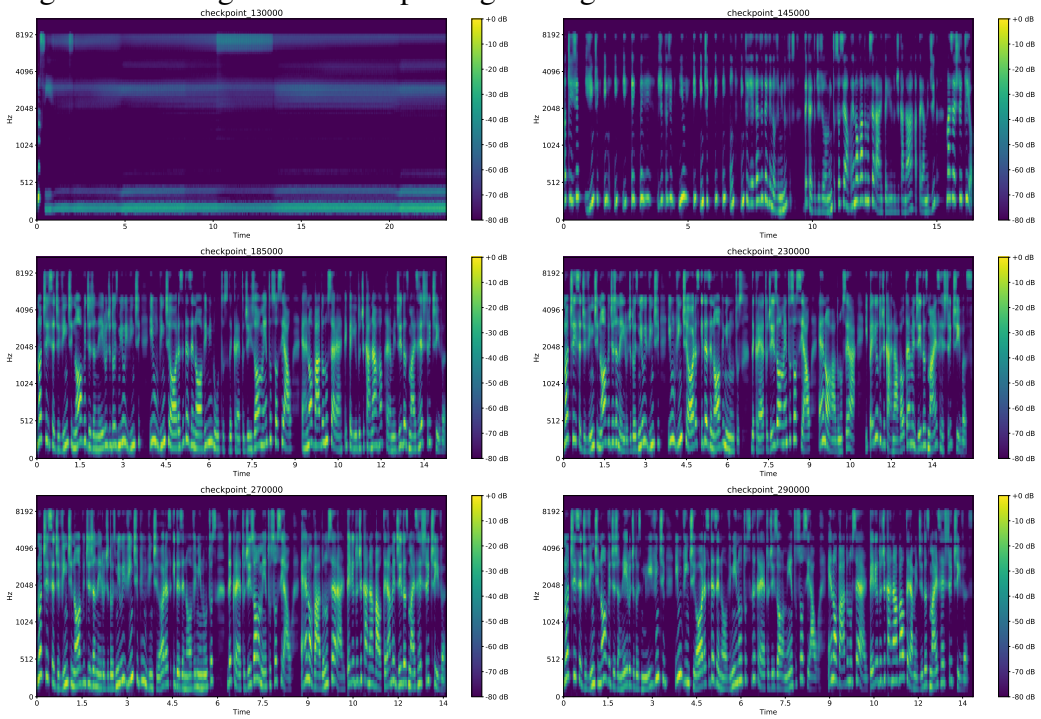
Fonte: elaborado pelo autor

Os modelos são capazes de gerar os arquivos de áudio em 0.5x tempo real, utilizando 1 vCPU a 2.20GHz e 1.3 GB de RAM. Alguns exemplos podem ser escutados em aqui[11].

Apesar deste alinhamento da camada de atenção fornecer uma base para nossa avaliação, como demonstrado anteriormente, o desempenho final do modelo deve ser avaliado de forma subjetiva através de uma avaliação de *MOS*. Devido a complexidade de realização deste tipo de avaliação, foi criado um formulário online utilizando o conjunto de testes, distribuído em alguns grupos de *Whatsapp*, *Telegram* e *Facebook*. Foram selecionados 20 exemplos para serem sintetizados e 20 áudios reais, assim, pode-se ter uma comparação da opinião dos áudios gerados com os áudios reais. O formulário obteve 26 respostas. A Tabela 7 apresenta os valores de *MOS* para os áudios originais e para o modelo treinado, com seus respectivos intervalos de confiança de 95%.

[11] http://bit.ly/tcc-xavier-voice-samples

Figura 12 – Progressão dos espectrogramas gerados



Fonte: elaborado pelo autor

Tabela 7 – Avaliação *MOS*

	MOS
Ground Truth	$4.894(\pm 0.206)$
Tacotron 2 / Multi Band Mel-GAN	$3.842(\pm 0.502)$

Fonte: elaborado pelo autor

5.9 Integração do *Chatbot* a canais de comunicação

O *Rasa* apresenta alguns conectores para facilicar sua integração aplicações e alguns canais de comunicação, porém estes conectores não suportam o recebimento e envio de mensagens de voz. Assim, foi necessário modificar os conectores de *API REST*, *Websocket* e *Telegram*.

Por padrão, os conectores *REST* e *Websocket* aceitam requisições com os campos `sender` e `message`. A função que lida com a entrada das requisições foi modificada para receber um novo campo, `type`, que indica o formato da entrada. O valor `text` indica uma mensagem de texto, enquanto o valor `audio` indica uma mensagem de voz. Com esta informação, podemos simplismente chamar a interface *Rasa Voice*. Caso o *chatbot* deva retornar um arquivo de áudio, é criado um novo campo `audio` na resposta, e este, juntamente com o texto original é retornado ao usuário. O canal do *Telegram* comunica diretamente com o *Telegram Bot API*[12]. Em casos de

[12] https://core.telegram.org/api

mensagens de voz, é recebido apenas um arquivo de áudio, que é tratado e repassado ao *Rasa Voice*. Para responder ao usuário em voz, é necessário converter a resposta do *Rasa Voice* em um formato de áudio aceito pelo *Telegram Bot API* e enviar apenas este arquivo. Para seguir o padrão estabelecido, apos o envio da mensagem de voz é enviada uma mensagem com o texto original.

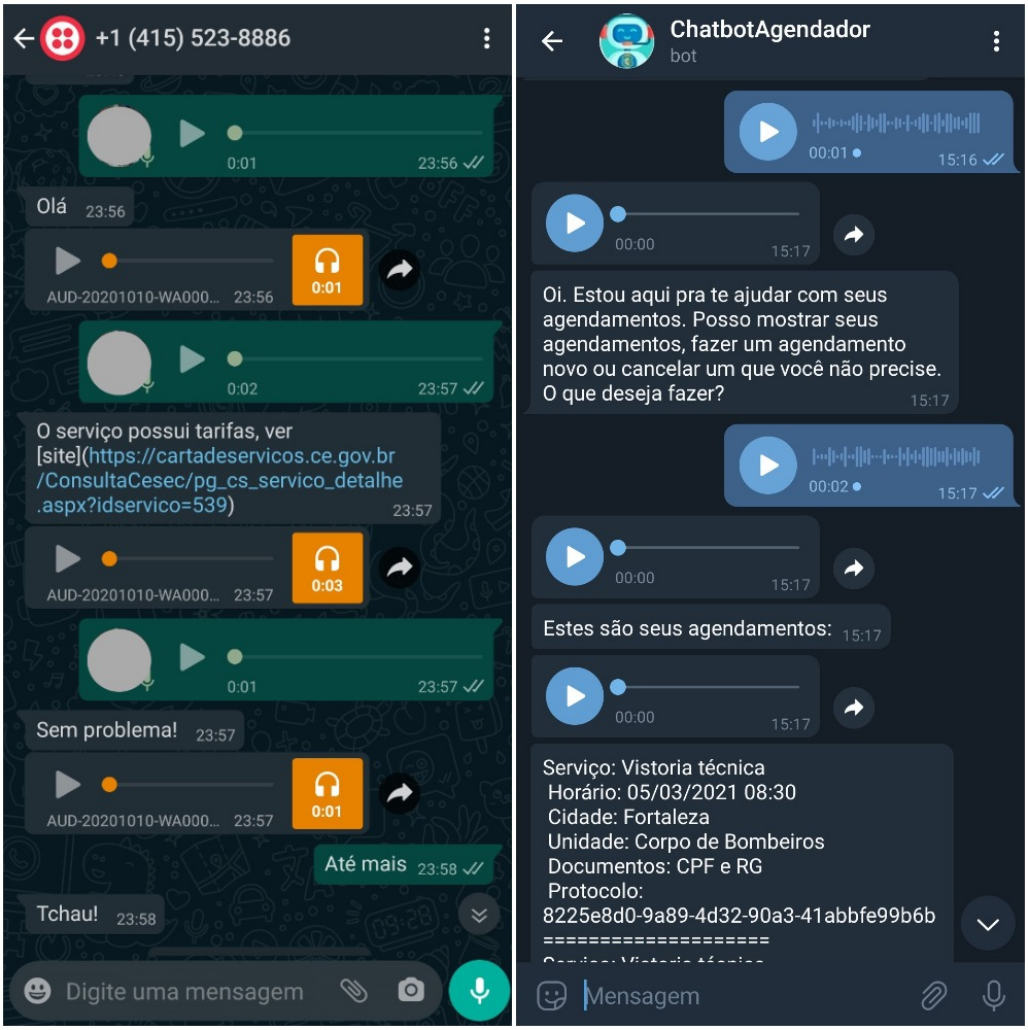
O aplicativo de mensagens mais popular no Brasil é o *Whatsapp*, portanto, descobrir como adicionar um *chatbot* a esta plataforma é essencial. Isso é possível utilizando o *Whatsapp Business*, que permite enviar notificações e automatizar respostas a mensagens. Para criar um perfil no *Whatsapp Business* é necessário criar uma conta do Gerenciador de Negócios do Facebook, verificar a empresa, criar uma conta no *WhatsApp Business* e abrir uma linha de crédito para sua conta comercial [13]. Somente após estes passos é possível utilizar a *API* do *Whatsapp Business*.

Na *API* do *Whatsapp Business* existem dois modos de mensagem, Mensagem de Modelo e Mensagem de Seção. Mensagem de Modelo seguem os padrões providenciados pela *API*, e são enviadas para o usuário em forma de notificação. O *Whatsapp Business* cobra por cada Mensagem de Modelo enviada[14]. As Mensagens de Seção permitem a resposta automática de mensagens. Quando um usuário envia uma mensagem ao *chatbot* do *Whatsapp Business* é iniciada um seção de 24 horas, e dentro deste período o *chatbot* pode trocar as mensagens em qualquer formato com usuário.

Devido aos requisitos necessários para criar um *chatbot* no *Whatsapp Business*, foram buscadas alternativas para que esta integração possa ser testada. Para isso, a plataforma de comunicações *Twillio* permite a criação de um *bot* utilizando a *API* do *Whatsapp Business* no modo *Sandbox*[15]. Com o modo *Sandbox*, foi possível testar a conversação com o *chatbot* através do *Whatsapp*. Para isso, foi necessário redirecionar as mensagens recebidas pela *Sandbox* para o conector *REST* do *Rasa Voice*. A Figura 13 apresenta conversas de voz com o *chatbot* utilizando o *Whatsapp* e o *Telegram*.

[13] https://developers.facebook.com/docs/whatsapp/getting-started
[14] https://developers.facebook.com/docs/whatsapp/pricing/
[15] https://www.twilio.com/docs/whatsapp/sandbox

Figura 13 – Conversa com o *chatbot* pelo *Whatsapp* e *Telegram*



Fonte: elaborado pelo autor

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho propôs o uso de reconhecimento de voz e transformação de texto em voz como formas de interagir com *chatbots* através de voz. Para atingir este objetivo, foram exploradas ferramentas e arquiteturas de Redes Neurais Profundas para realizar as tarefas de *ASR* e *TTS*, obtidos conjuntos de dados e estudadas métricas para realização de testes e experimentos, criado de um conjunto de dados próprio para treinamento de modelo para *TTS* e a integradas estas ferramentas a um *framework* para construção de *chatbots*.

Após a análise das *APIs* de *ASR*, foi escolhida a ferramenta *Wit.ai* por ser gratuita e possuir desempenho semelhante as outras analisadas. Foram realizados experimentos para avaliar a qualidade de transcrição de fala em portugês do Brasil utilizando *WER*, a principal métrica para análise de transcrição de voz para texto, além das métricas calculo de similaridade Distância de Jaccard e Similaridade Cosseno, e métricas de avaliação de tradução de texto *BLEU* e *METEOR*. Para isto foram utilizado os conjuntos de dados *Mozilla Common Voice*, *Voxforge* e do Grupo Fala Brasil. Os resultados mostraram desempenho semelhante a ferramentas estado da arte, mesmo ao comparação com a língua inglesa.

Para a etapa de *TTS* foram explorados modelos de aprendizado profundo, pois isto permitiu a criação de uma voz customizada, sem o alto custo identificado nas *APIs*. O trabalho de (CASANOVA *et al.*, 2020) foi utilizado como base para experimentação e escolha dos modelos. Foi explorada no trabalho a possibilidade de utilizar métricas objetivas para esta avaliação da qualidade da voz gerada por modelos de *TTS* usando *DTW* e Similaridade Cosseno. Os resultados obtidos demonstraram que estas métricas não foram capazes de refletir a qualidade dos áudios.

Foi criado um novo conjunto de dados utilizando uma voz cearense, para criar uma identificação com o público alvo do chatbot, que possui 1985 textos com 5 horas e 4 minutos de áudio. Foram treinados modelos das arquiteturas *Tacotron 2* e *Multi-Band MelGAN*, utilizando a ferramenta *Mozilla TTS* e da técnica de *transfer learning*, tomando como base modelos treinados em inglês no *dataset* de Ito e Johnson (2017). Ao final do treinamento os modelos foram capazes de gerar uma voz de boa qualidade.

Com as ferramentas de *ASR* e *TTS* definidas, foi implementada uma interface de comunicação por voz para o *framework Rasa*, e integrado o chatbot uma aplicação *Web*, ao *Telegram* e ao *Whatsapp*. Esta interface integrada ao *Rasa* permite que qualquer *chatbots* desenvolvidos utilizando o *framework* possam utilizar desta forma de interação mais acessível.

Como propostas de melhorias e trabalhos futuros, remover a dependência de uma

API externa para o *ASR* traria mais flexibilidade e controle a ferramenta. O estudo e adaptação de ferramentas como *fairseq*[1] e *huggingface Wav2Vec2*[2] podem ajudar nesta tarefa. Para os modelos de *TTS*, aumentar o tamanho do conjunto de dados para abranger uma maior quantidade de fonemas traria melhorias no resultado final. Vale também realizar o treinamento dos modelos sem utilizar de *transfer learning*.

[1] https://github.com/pytorch/fairseq
[2] https://huggingface.co/transformers/model_doc/wav2vec2.html

REFERÊNCIAS

ACCENTURE INTERACTIVE. **Chatbots in Customer Service**. [*S. l.*], 2016. Disponível em: https://www.accenture.com/t00010101T000000\%5F\%5Fw\%5F\%5F/br-pt/\%5Facnmedia/PDF-45/Accenture-Chatbots-Customer-Service.pdf. Acesso em: 09 abr. 2021.

ADAMOPOULOU, E.; MOUSSIADES, L. An overview of chatbot technology. In: MAGLOGIANNIS, I.; ILIADIS, L.; PIMENIDIS, E. (Ed.). **Artificial Intelligence Applications and Innovations - 16th IFIP WG 12.5 International Conference, AIAI 2020, June 5-7**. Neos Marmaras, Greece: Springer, 2020. (IFIP Advances in Information and Communication Technology, v. 584), p. 373–383.

AIML FOUNDATION. **AIML Foundation**. [*S. l.*], 2020. Disponível em: http://www.aiml.foundation/. Acesso em: 09 abr. 2021.

BAEVSKI, A.; ZHOU, Y.; MOHAMED, A.; AULI, M. wav2vec 2.0: A framework for self-supervised learning of speech representations. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020**. Virtual: Curran Associates, Inc., 2020. v. 33, p. 12449–12460. Disponível em: https://proceedings.neurips.cc/paper/2020/hash/92d1e1eb1cd6f9fba3227870bb6d7f07-Abstract.html. Acesso em: 09 abr. 2021.

BOCKLISCH, T.; FAULKNER, J.; PAWLOWSKI, N.; NICHOL, A. **Rasa: Open Source Language Understanding and Dialogue Management**. [*S. l.*], 2017. Disponível em: http://arxiv.org/abs/1712.05181. Acesso em: 09 abr. 2021.

BRANDTZÆG, P. B.; FØLSTAD, A. Chatbots: changing user needs and motivations. **Interactions**, Association for Computing Machinery, v. 25, n. 5, p. 38–43, 2018. Disponível em: https://doi.org/10.1145/3236669. Acesso em: 09 abr. 2021.

CASANOVA, E.; CANDIDO JUNIOR, A.; OLIVEIRA, F. S. de; SHULBY, C.; TEIXEIRA, J. P.; PONTI, M. A.; ALUÍSIO, S. M. **End-To-End Speech Synthesis Applied to Brazilian Portuguese**. [*S. l.*], 2020. Disponível em: https://arxiv.org/abs/2005.05144. Acesso em: 09 abr. 2021.

COHEN, P.; OVIATT, S. The role of voice input for human-machine communication. **Proceedings of the National Academy of Sciences**, National Academy of Sciences, v. 92, p. 9921–9927, 1995.

CUTAJAR, M.; GATT, E.; GRECH, I.; CASHA, O.; MICALLEF, J. Comparative study of automatic speech recognition techniques. **IET Signal Process.**, IET, v. 7, n. 1, p. 25–46, 2013. Disponível em: https://doi.org/10.1049/iet-spr.2012.0151. Acesso em: 09 abr. 2021.

DALE, R. The return of the chatbots. **Natural Language Engineering**, Cambridge University Press, v. 22, n. 5, p. 811–817, 2016. Disponível em: https://doi.org/10.1017/S1351324916000243. Acesso em: 09 abr. 2021.

EHSANI, F.; KNODT, E. Speech technology in computer-aided language learning: Strengths and limitations of a new call paradigm. **Language Learning & Technology**, University of Hawaii National Foreign Language Resource Center, v. 2, 09 1998.

FØLSTAD, A.; BRANDTZÆG, P. B. Chatbots and the new world of HCI. **Interactions**, Association for Computing Machinery, v. 24, n. 4, p. 38–42, 2017. Disponível em: https://doi.org/10.1145/3085558. Acesso em: 09 abr. 2021.

GOODFELLOW, I. J.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A. C.; BENGIO, Y. Generative adversarial nets. In: GHAHRAMANI, Z.; WELLING, M.; CORTES, C.; LAWRENCE, N. D.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014**. Montreal, Quebec, Canada: [*S. n.*], 2014. p. 2672–2680. Disponível em: https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html. Acesso em: 09 abr. 2021.

GOOGLE TRENDS. **Chatterbot - Google Trends**. [*S. l.*], 2021. Disponível em: https://trends.google.com.br/trends/explore?date=2004-01-01%202021-01-02&q=chatbot. Acesso em: 09 abr. 2021.

GRIGORE, E. C.; PEREIRA, A.; ZHOU, I.; WANG, D.; SCASSELLATI, B. Talk to me: Verbal communication improves perceptions of friendship and social presence in human-robot interaction. In: TRAUM, D. R.; SWARTOUT, W. R.; KHOOSHABEH, P.; KOPP, S.; SCHERER, S.; LEUSKI, A. (Ed.). **Intelligent Virtual Agents - 16th International Conference, IVA 2016, September 20-23, 2016, Proceedings**. Los Angeles, CA, USA: Springer, 2016. (Lecture Notes in Computer Science, v. 10011), p. 51–63.

HARTMANN, N.; FONSECA, E. R.; SHULBY, C.; TREVISO, M. V.; RODRIGUES, J. S.; ALUÍSIO, S. M. Portuguese word embeddings: Evaluating on word analogies and natural language tasks. In: PAETZOLD, G. H.; PINHEIRO, V. (Ed.). **Proceedings of the 11th Brazilian Symposium in Information and Human Language Technology, STIL 2017, October 2-5, 2017**. Uberlândia, Brazil: Sociedade Brasileira de Computação, 2017. p. 122–131. Disponível em: https://www.aclweb.org/anthology/W17-6615/. Acesso em: 09 abr. 2021.

HUNT, M. J. Figures of merit for assessing connected-word recognisers. **Speech Communication**, v. 9, n. 4, p. 329–336, 1990. ISSN 0167-6393. Disponível em: https://www.sciencedirect.com/science/article/pii/016763939090008W. Acesso em: 09 abr. 2021.

INDUMATHI, A.; CHANDRA, E. Survey on speech synthesis. **Signal Processing: An International Journal (SPIJ)**, Citeseer, v. 6, n. 5, p. 140, 2012.

ITO, K.; JOHNSON, L. **The LJ Speech Dataset**. [*S. l.*], 2017. Disponível em: https://keithito.com/LJ-Speech-Dataset/. Acesso em: 09 abr. 2021.

JACCARD, P. The distribution of the flora in the alpine zone.1. **New Phytologist**, v. 11, n. 2, p. 37–50, 1912. Disponível em: https://nph.onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-8137.1912.tb05611.x. Acesso em: 09 abr. 2021.

KALCHBRENNER, N.; ELSEN, E.; SIMONYAN, K.; NOURY, S.; CASAGRANDE, N.; LOCKHART, E.; STIMBERG, F.; OORD, A. van den; DIELEMAN, S.; KAVUKCUOGLU, K. Efficient neural audio synthesis. In: DY, J. G.; KRAUSE, A. (Ed.). **Proceedings of the 35th International Conference on Machine Learning, ICML 2018, July 10-15, 2018**. Stockholmsmässan, Stockholm, Sweden: Pmlr, 2018. (Proceedings of Machine

Learning Research, v. 80), p. 2415–2424. Disponível em: http://proceedings.mlr.press/v80/kalchbrenner18a.html. Acesso em: 09 abr. 2021.

KUMAR, M. N.; CHANDAR, P. L.; PRASAD, A. V.; SUMANGALI, K. Android based educational chatbot for visually impaired people. In: IEEE. **2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)**. Tamil Nadu, India, 2016. p. 1–4.

LAVIE, A.; AGARWAL, A. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In: **Proceedings of the Second Workshop on Statistical Machine Translation**. Prague, Czech Republic: Association for Computational Linguistics, 2007. (StatMT '07), p. 228–231.

LIDDY, E. D. Natural language processing. In: DECKER, I. M. (Ed.). **Encyclopedia of Library and Information Science**. 2. ed. New York: Marcel Decker, Inc, 2001.

NURUZZAMAN, M.; HUSSAIN, O. K. A survey on chatbot implementation in customer service industry through deep neural networks. In: **15th IEEE International Conference on e-Business Engineering, ICEBE 2018, October 12-14, 2018**. Xi'an, China: IEEE Computer Society, 2018. p. 54–61. Disponível em: https://doi.org/10.1109/ICEBE.2018.00019. Acesso em: 09 abr. 2021.

OORD, A. van den; DIELEMAN, S.; ZEN, H.; SIMONYAN, K.; VINYALS, O.; GRAVES, A.; KALCHBRENNER, N.; SENIOR, A. W.; KAVUKCUOGLU, K. Wavenet: A generative model for raw audio. In: **The 9th ISCA Speech Synthesis Workshop, 13-15 September 2016**. Sunnyvale, CA, USA: ISCA, 2016. p. 125. Disponível em: http://www.isca-speech.org/archive/SSW_2016/abstracts/ssw9_DS-4_van_den_Oord.html.

PALMA, J. D. G. **Solução Mobile–Voice Assisted Chatbots**. Dissertação (Mestrado) – Universidade de Lisboa, Faculdade de Ciências, Departamento de Informática, 2019. Mestrado em Engenharia Informática. Disponível em: http://hdl.handle.net/10451/40279. Acesso em: 09 abr. 2021.

PAPINENI, K.; ROUKOS, S.; WARD, T.; ZHU, W. Bleu: a method for automatic evaluation of machine translation. In: **Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002**. Philadelphia, PA, USA: ACL, 2002. p. 311–318. Disponível em: https://www.aclweb.org/anthology/P02-1040/. Acesso em: 09 abr. 2021.

QUINTANILHA, I. M. **End-to-end speech recognition applied to brazilian portuguese using deep learning**. Dissertação (Mestrado) – Universidade Federal do Rio de Janeiro, Programa de Pós-graduação em Engenharia Elétrica, 2017. Mestrado em Engenharia Elétrica. Disponível em: https://sucupira.capes.gov.br/sucupira/public/consultas/coleta/trabalhoConclusao/viewTrabalhoConclusao.jsf?popup=true&id_trabalho=5002494. Acesso em: 09 abr. 2021.

RIBEIRO, F.; Florêncio, D.; Zhang, C.; Seltzer, M. Crowdmos: An approach for crowdsourcing mean opinion score studies. In: **2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. Prague, Czech Republic: [*S. n.*], 2011. p. 2416–2419.

RUAN, S.; WILLIS, A.; XU, Q.; DAVIS, G. M.; JIANG, L.; BRUNSKILL, E.; LANDAY, J. A. Bookbuddy: Turning digital materials into interactive foreign language lessons through a voice chatbot. In: **Proceedings of the Sixth ACM Conference on Learning Scale, LS**

2019, June 24-25, 2019. Chicago, IL, USA: ACM, 2019. p. 30:1–30:4. Disponível em: https://doi.org/10.1145/3330430.3333643. Acesso em: 09 abr. 2021.

SALVADOR, S.; CHAN, P. Toward accurate dynamic time warping in linear time and space. **Intell. Data Anal.**, IOS Press, Nld, v. 11, n. 5, p. 561–580, out. 2007. ISSN 1088-467x.

SCHNEIDER, S.; BAEVSKI, A.; COLLOBERT, R.; AULI, M. wav2vec: Unsupervised pre-training for speech recognition. In: KUBIN, G.; KACIC, Z. (Ed.). **Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, 15-19 September 2019**. Graz, Austria: ISCA, 2019. p. 3465–3469. Disponível em: https://doi.org/10.21437/Interspeech.2019-1873. Acesso em: 09 abr. 2021.

SHAWAR, B. A.; ATWELL, E. Chatbots: Are they really useful? **LDV Forum**, v. 22, n. 1, p. 29–49, 2007.

SHEN, J.; PANG, R.; WEISS, R. J.; SCHUSTER, M.; JAITLY, N.; YANG, Z.; CHEN, Z.; ZHANG, Y.; WANG, Y.; RYAN, R.; SAUROUS, R. A.; AGIOMYRGIANNAKIS, Y.; WU, Y. Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions. In: **2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018, April 15-20, 2018**. Calgary, AB, Canada: IEEE, 2018. p. 4779–4783. Disponível em: https://doi.org/10.1109/ICASSP.2018.8461368. Acesso em: 09 abr. 2021.

SITIKHU, P.; PAHI, K.; THAPA, P.; SHAKYA, S. A comparison of semantic similarity methods for maximum human interpretability. In: IEEE. **2019 Artificial Intelligence for Transforming Business and Society (AITB)**. Kathmandu, Nepal, 2019. v. 1, p. 1–4. Disponível em: http://arxiv.org/abs/1910.09129. Acesso em: 09 abr. 2021.

TABET, Y.; BOUGHAZI, M. Speech synthesis techniques. a survey. In: IEEE. **International Workshop on Systems, Signal Processing and their Applications, WOSSPA**. [*S. l.*], 2011. p. 67–70.

TACHIBANA, H.; UENOYAMA, K.; AIHARA, S. Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention. In: **2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. Calgary, AB, Canada: IEEE, 2017. p. 4784–4788. Disponível em: https://ieeexplore.ieee.org/abstract/document/8461829. Acesso em: 09 abr. 2021.

TZOUGANATOU, A. Can heritage bots thrive? toward future engagement in cultural heritage. **Advances in Archaeological Practice**, Cambridge University Press, v. 6, n. 4, p. 377–383, 2018.

VELA, M.; SCHUMANN, A.-K.; WURM, A. Human translation evaluation and its coverage by automatic scores. In: **Automatic and Manual Metrics for Operational Translation Evaluation Workshop Programme**. Reykjavik, Iceland: [*S. n.*], 2014. p. 19.

WANG, Y.; SKERRY-RYAN, R. J.; STANTON, D.; WU, Y.; WEISS, R. J.; JAITLY, N.; YANG, Z.; XIAO, Y.; CHEN, Z.; BENGIO, S.; LE, Q. V.; AGIOMYRGIANNAKIS, Y.; CLARK, R.; SAUROUS, R. A. Tacotron: Towards end-to-end speech synthesis. In: LACERDA, F. (Ed.). **Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, August 20-24, 2017**. Stockholm, Sweden: Isca, 2017. p. 4006–4010. Disponível em: https://arxiv.org/abs/1703.10135. Acesso em: 09 abr. 2021.

YAMAMOTO, R.; SONG, E.; KIM, J. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In: **2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, May 4-8, 2020**. Barcelona, Spain: IEEE, 2020. p. 6199–6203. Disponível em: https://doi.org/10.1109/ICASSP40776.2020.9053795. Acesso em: 09 abr. 2021.

YANG, G.; YANG, S.; LIU, K.; FANG, P.; CHEN, W.; XIE, L. Multi-band melgan: Faster waveform generation for high-quality text-to-speech. In: IEEE. **2021 IEEE Spoken Language Technology Workshop (SLT)**. Shenzhen, China: IEEE, 2021. p. 492–498. Disponível em: https://ieeexplore.ieee.org/abstract/document/9383551. Acesso em: 09 abr. 2021.

APÊNDICE A – LINKS PARA APIS

A partir dos links a seguir é possível ver a documentação, preço e quotas das *APIs* para *ASR* e *TTS* que foram analisadas.

A.0.1 APIs para ASR

- *Azure Speech Services*: https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/
- *Google Cloud Speech-to-Text*: https://cloud.google.com/speech-to-text,
- *IBM Watson Speech to Text*: https://www.ibm.com/br-pt/cloud/watson-speech-to-text/,
- *Wit.ai*: https://wit.ai/
- *Amazon Transcribe*: https://aws.amazon.com/pt/transcribe/

A.0.2 APIs para TTS

- *Azure Speech Services*: https://azure.microsoft.com/pt-br/services/cognitive-services/text-to-speech/
- *Google Cloud Text-to-Speech*: https://cloud.google.com/text-to-speech/
- *IBM Watson Text to Speech*: https://www.ibm.com/br-pt/cloud/watson-text-to-speech
- *Amazon Polly*: https://aws.amazon.com/pt/polly/

A.0.2.1 TTS com voz customizada

- *Azure Speech Services*: https://speech.microsoft.com/customvoice
- *Google Cloud Text-to-Speech*: https://cloud.google.com/text-to-speech/custom-voice/docs
- *Amazon Polly*: https://aws.amazon.com/pt/polly/features/#Brand_Voice

APÊNDICE B – ESTRUTURA DE PASTAS DO CORPUS DE *ASR*

Esta é uma representação da árvore de arquivos após a padronização dos *datasets* do Grupo FalaBrasil, *Vorxforge* e *Mozilla Common Voice*.

```
corpus/
    cod_def_cons/
        audios/
            audio1.wav
            audio2.wav
            ...
        sentences.txt
    constituicao/
        audios/
            ...
        sentences.txt
    voxforge/
        audios/
            ...
        sentences.txt
    mozilla/
        audios/
            ...
        sentences.txt
```