UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ALEXANDRE RIBEIRO FERREIRA

INVESTIGAÇÕES DO USO DE *DEEPFAKE AUDIO* COMO UMA TÉCNICA DE AUMENTO DE DADOS UTILIZADOS NO TREINAMENTO DE TRANSCRITORES AUTOMÁTICOS

CAMPINA GRANDE - PB

2022

ALEXANDRE RIBEIRO FERREIRA

INVESTIGAÇÕES DO USO DE *DEEPFAKE AUDIO* COMO UMA TÉCNICA DE AUMENTO DE DADOS UTILIZADOS NO TREINAMENTO DE TRANSCRITORES AUTOMÁTICOS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. Cláudio Elízio Calazans Campelo

CAMPINA GRANDE - PB

2022

ALEXANDRE RIBEIRO FERREIRA

INVESTIGAÇÕES DO USO DE *DEEPFAKE AUDIO* COMO UMA TÉCNICA DE AUMENTO DE DADOS UTILIZADOS NO TREINAMENTO DE TRANSCRITORES AUTOMÁTICOS

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Cláudio Elízio Calazans Campelo

Orientador – UASC/CEEI/UFCG

Carlos Eduardo Santos Pires

Examinador – UASC/CEEI/UFCG

Francisco Vilar Brasileiro

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 02 de Setembro de 2022.

CAMPINA GRANDE - PB

RESUMO

Para o treinamento de modelos transcritores que produzam resultados robustos, são necessários dados rotulados em grande quantidade e diversificados. Encontrar tais dados com as características necessárias é uma tarefa difícil, principalmente em idiomas menos populares do que o inglês. Além disso, produzir tais dados requer bastante esforço, tempo e, quase sempre, dinheiro. Logo, uma estratégia para mitigar esse problema é a utilização de técnicas de aumento de dados. Nesse trabalho, foi investigada a utilização de *deepfake audio* para o aumento de dados, utilizando um clonador de voz capaz de gerar novos áudios mantendo características da voz do falante original, como, por exemplo, o sotaque. Para tanto, foi selecionado um pequeno conjunto de dados produzido por indianos no idioma inglês, garantindo a presença de apenas um sotaque no conjunto. Para a realização das investigações, experimentos foram conduzidos utilizando o clonador para o aumento de dados. Em seguida, os dados aumentados foram utilizados no treinamento dos transcritores, em diversos cenários. Surpreendentemente, a estratégia não teve um impacto positivo após a realização dos treinamentos, tendo como possível causa a qualidade dos áudios gerados pelos clonadores atuais.

Investigações do uso de *deepfake audio* como uma técnica de aumento de dados utilizados no treinamento de transcritores automáticos

Trabalho de Conclusão de Curso

Alexandre Ribeiro Ferreira (Aluno), Cláudio Campelo (Orientador)
Departamento de Sistemas e Computação
Universidade Federal de Campina Grande
Campina Grande, Paraíba - Brasil

RESUMO

Para o treinamento de modelos transcritores que produzam resultados robustos, são necessários dados rotulados em grande quantidade e diversificados. Encontrar tais dados com as características necessárias é uma tarefa difícil, principalmente em idiomas menos populares do que o inglês. Além disso, produzir tais dados requer bastante esforço, tempo e, quase sempre, dinheiro. Logo, uma estratégia para mitigar esse problema é a utilização de técnicas de aumento de dados. Nesse trabalho, foi investigada a utilização de *deepfake audio* para o aumento de dados, utilizando um clonador de voz capaz de gerar novos áudios mantendo características da voz do falante original, como, por exemplo, o sotaque. Para tanto, foi selecionado um pequeno conjunto de dados produzido por indianos no idioma inglês, garantindo a presença de apenas um sotaque no conjunto. Para a realização das investigações, experimentos foram conduzidos utilizando o clonador para o aumento de dados. Em seguida, os dados aumentados foram utilizados no treinamento dos transcritores, em diversos cenários. Surpreendentemente, a estratégia não teve um impacto positivo após a realização dos treinamentos, tendo como possível causa a qualidade dos áudios gerados pelos clonadores atuais.

PALAVRAS-CHAVE

Aumento de Dados, Deepfake Audio, Clonagem de Voz, Transcritores

1 INTRODUÇÃO

A inteligência artificial teve seu crescimento impulsionado nos últimos anos devido ao aumento do poder computacional e à ampliação da variedade e do volume dos dados trafegados pela *internet*. A busca por modelos gerados a partir do aprendizado de máquina se expandiu por meio de diversas aplicações ao redor do mundo, como é o caso dos modelos transcritores fala-para-texto (*speech-to-text*), os quais são utilizados, por exemplo, em tradutores, assistentes virtuais, pesquisas por voz e análises de sentimentos em áudio [4].

No treinamento desses modelos transcritores são necessários dados rotulados, ou seja, pares de áudios com suas respectivas transcrições. Essas transcrições devem ser feitas por humanos a fim de evitar o enviesamento dos resultados causado por transcrições produzidas a partir de outro modelo. Além disso, o aumento no uso dos transcritores provocou a necessidade de que os mesmos produzissem resultados robustos, ou seja, os mesmos devem conseguir compor um resultado independentemente das variações de uma determinada linguagem. Por exemplo, eles devem produzir transcrições que se mantenham consistentes para diferentes sotaques de um mesmo idioma.

Para que os transcritores consigam produzir esses resultados, são necessárias mais etapas de treinamento, além da utilização de mais dados, dados esses, diversificados e em abundância. Dados diversos são importantes no treinamento, pois ajudam o transcritor a preservar a qualidade dos seus resultados, independentemente das variações nos dados de entrada. No entanto, conseguir conjuntos de dados com essas características é uma tarefa desafiadora, principalmente em idiomas menos populares do que o inglês.

Produzir um grande conjunto de dados que possua essas características é custoso e demanda muito tempo e muito dinheiro, além da estrutura necessária para que a produção ocorra, visto que, várias pessoas capacitadas devem produzir as transcrições manualmente. Além disso, para garantir uma boa qualidade das transcrições, cada áudio deve ter sua transcrição produzida por mais de uma pessoa, dessa forma, viabiliza-se a escolha da transcrição que melhor representa esse áudio. Nesse sentido, o custo desse processo cresce à medida que o conjunto de dados a ser produzido aumenta.

Uma alternativa é encontrar um conjunto de dados que possua os aspectos necessários para o treinamento do transcritor, conforme sua aplicação. Contudo, exceto em alguns casos, essa busca demanda tempo e raramente se encontra um conjunto de dados com todas ou grande parte das características necessárias. Além disso, quando se encontra um conjunto com tais características, é necessário avaliar a qualidade dos seus dados e verificar se a licença permite o seu uso conforme a sua demanda, pois é comum que os conjuntos de dados possuam licenças que impõem restrições de uso em diversos cenários.

Uma opção para mitigar esse problema, diminuindo os custos de tempo e de dinheiro, é utilizar técnicas de aumento de dados (*data augmentation*). Existem diversas técnicas de aumento de dados, no entanto, a maioria delas permite apenas a geração de novos dados que tenham características semelhantes, por exemplo, adicionando novas características como ruído de fundo ou alterando a tonalidade da voz do falante presente no áudio.

Essas técnicas são eficientes, porém, não muito eficazes a depender das necessidades do seu transcritor. Por exemplo, quando

elas são utilizadas em dados que servirão para o treinamento de um modelo transcritor, elas são eficazes em ajudar o transcritor a produzir resultados de qualidade semelhante, independentemente do ruído de fundo ou tonalidade da voz presente em seu áudio de entrada.

No entanto, alguns transcritores precisam manter a qualidade da sua transcrição, mesmo que outras características variem no áudio de entrada. Por exemplo, áudios que possuem a mesma frase, porém, são produzidos por pessoas que possuem sotaques diferentes, devem ter como saída a mesma transcrição, apesar do sotaque. Para isso, o modelo transcritor precisa ser treinado com dados que possuem variações de sotaque entre os falantes, de forma que o mesmo aprenda a transcrever falas nos mais diversos sotaques.

A técnica de aumento de dados investigada nesse trabalho faz o uso de *deepfake audio*. O *deepfake audio* é uma área da inteligência artificial cujo objetivo é produzir áudios que simulem as vozes de determinadas pessoas, ou seja, os áudios soam como se elas próprias tivessem produzido. Existem vários tipos de modelos que cumprem esse objetivo, nesse trabalho, é utilizado um modelo que permite a clonagem da voz a partir de poucos segundos de áudio do falante original. Com isso, a técnica de aumento de dados se beneficia, pois consegue produzir áudios de um mesmo falante com diferentes conteúdos de fala, ao passo que mantém as características da voz presente no áudio, como, por exemplo, o sotaque.

O objetivo desse trabalho é investigar o uso dessa técnica em conjuntos de dados utilizados no treinamento de transcritores automáticos, avaliando o impacto produzido na eficácia dos mesmos. Para tanto, diversos cenários são investigados e um pequeno conjunto de dados é utilizado para aplicação dessa técnica. Com isso, o conjunto de dados aumentados produzido é utilizado no treinamento de um modelo transcritor, isso se dá a partir de um modelo pré-treinado utilizando um processo de *fine-tunning*. Por fim, uma pequena parcela dos dados originais é reservada para testar o transcritor antes e depois do treinamento, identificando se houve melhora, ou não, nas transcrições produzidas.

As principais contribuições desse trabalho são:

- A avaliação de um cenário completamente diferente para o aumento de dados com a utilização de *deepfake audio*. Nesse sentido, não se verificou na literatura trabalhos relacionados. Essa avaliação é importante para que, novas pessoas interessadas no assunto possam analisar as investigações realizadas nesse trabalho e nortear um caminho para o qual suas investigações podem seguir.
- Conceder e implementar uma estratégia para a utilização de *deepfake audio* como uma técnica de aumento de dados. A implementação está pronta para uso e disponível no repositório[1] desse trabalho. Caso alguém queira testar a estratégia com um clonador de voz melhor, basta trocar a parte da geração de novos áudios e utilizar o restante da implementação normalmente.

Dois experimentos foram conduzidos para a realização das investigações. No primeiro, o clonador de voz é utilizado com os modelos pré-treinados disponibilizados pelo autor, com isso, os áudios são gerados e utilizados no treinamento do transcritor em diversos cenários. Por fim, uma avaliação dos resultados é conduzida mostrando que a qualidade das transcrições pioraram e a métrica *Word Error Rate* (WER) aumentou cerca de dois por cento.

Buscando melhores resultados, um segundo experimento foi realizado. Nesse experimento, diferente do anterior, dois dos três modelos utilizados pelo clonador de voz são treinados em diferentes cenários. Em seguida, a melhor combinação de modelos treinados é escolhida para a geração dos áudios e posterior treinamento do modelo transcritor em diversos cenários, conforme o experimento anterior. Por fim, a avaliação dos resultados mostrou uma piora na qualidade das transcrições, com a métrica WER aumentando cerca de seis por cento. A qualidade das transcrições geradas pelos modelos transcritores treinados nos dois experimentos pioraram com relação ao modelo pré-treinado, porém, acredita-se que essa piora nos resultados seja devida à qualidade ruim dos áudios gerados pelo clonador de voz.

O restante desse trabalho está estruturado da seguinte maneira. A próxima Seção apresenta os trabalhos relacionados. Em seguida, a Seção 3 fornece detalhes dos fundamentos teóricos para o entendimento da pesquisa. Logo após, a Seção 4 discorre sobre os procedimentos executados para a realização dos experimentos. Posteriormente, a Seção 5 descreve os experimentos realizados e os resultados alcançados, assim como, as possíveis causas dos resultados obtidos. Por fim, a Seção 6 apresenta conclusões para esse trabalho e aponta para trabalhos futuros.

2 TRABALHOS RELACIONADOS

Devido à necessidade crescente por grandes conjuntos de dados, diversas técnicas de aumento de dados vem sendo desenvolvidas ao passar dos anos. Algumas dessas técnicas são utilizadas no aumento de dados para treinamento de transcritores, criando áudios a partir de modificações dos existentes [9, 11, 14] ou a partir da geração de áudios utilizando modelos texto-para-fala (*text-to-speech*) [15].

A perturbação de velocidade [9] do áudio é utilizada como técnica para aumento dos dados. Para tanto, um valor *alpha* é definido e utilizado para realizar alterações na taxa de amostragem dos áudios. Com isso, novos áudios são criados com modificações em sua taxa de amostragem. Essa técnica é eficiente e se mostrou eficaz nos testes realizados pelos autores.

Uma técnica denominada *SpecAugment* [11] foi criada a partir da utilização de três métodos que alteram diretamente o espectrograma do áudio. O primeiro método comprime ou estica o espectrograma em pontos aleatórios no tempo de duração do áudio. O segundo utiliza uma máscara para cobrir determinados canais de frequência e, semelhantemente, o terceiro usa uma máscara para cobrir determinados passos de tempo. Segundo os autores, o uso combinado desses métodos obteve bons resultados.

Similar ao *SpecAugment*, a técnica denominada *SpecSwap* [14] utiliza dois métodos que alteram diretamente o espectrograma do áudio. O primeiro método conduz a troca de dois blocos de frequência de um espectrograma de forma aleatória. Enquanto o segundo método realiza a troca, aleatoriamente, de dois blocos do espectrograma ao longo do tempo de duração do áudio. Segundo os autores, essa técnica produz bons resultados, porém falta comparação com o *SpecAugment*.

Um método para o aumento de dados a partir da geração de áudio e texto sintéticos foi conduzido por Rodolfo [15] em sua tese

[1] https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados

de mestrado. Ele utilizou um conjunto de dados da língua Quechua, um modelo sequência-para-sequência (*sequence-to-sequence*) para a geração de texto e um modelo texto-para-fala (*text-to-speech*) para a geração de áudios na língua Quechua. Após a realização dos experimentos, ele reportou bons resultados e uma melhora na qualidade das transcrições geradas pelo modelo transcritor, o qual ele treinou com os dados aumentados.

Assim como nos trabalhos anteriores, nesse trabalho é investigado o aumento de dados utilizados para o treinamento de modelos transcritores. Para tanto, é utilizado um clonador de voz que permite geração de novos áudios utilizando características das vozes presentes no conjunto original, como, por exemplo, o sotaque. Esse é um grande diferencial quando comparado às técnicas mais simples ou que utilizam modelos fala-para-texto (TTS) convencionais. As técnicas mais simples mudam pequenas características ou geram pequenas distorções entre um áudio e outro, fazendo com que o transcritor fique robusto a essas mudanças. Enquanto técnicas que utilizam modelos TTS convencionais tendem a gerar áudios com uma voz padrão, com as mesmas características da voz presente para todos os áudios.

3 FUNDAMENTAÇÃO TEÓRICA

Esta seção fornece detalhes dos fundamentos teóricos necessários para um completo entendimento da pesquisa. Primeiramente, é explicado o funcionamento do clonador de voz escolhido para ser utilizado e, em seguida, é descrito o modelo transcritor escolhido.

3.1 Clonagem de voz

O clonador escolhido para a realização das investigações foi o Real-Time Voice Cloning, fornecido por Corentin Jemine em seu Github [6] e produzido durante sua tese de mestrado. Esse clonador foi escolhido para ser utilizado devido a sua capacidade de gerar novos áudios a partir de poucos segundos de um áudio de referência e sem a necessidade de re-treinamento dos modelos, mesmo que o áudio referência não tenha sido utilizado durante o seu treinamento.

O Real-Time Voice Cloning é uma implementação da arquitetura de aprendizado profundo `SV2TTS` [7], a qual é composta por três componentes treinados independentemente. O primeiro componente é um *encoder* treinado em uma tarefa de verificação de falantes utilizando um conjunto de dados sem a presença de transcrições, ele recebe como entrada poucos segundos de um áudio referência e tem como saída um vetor de *embeddings* de tamanho fixo. O segundo componente é um *synthesizer* baseado no `Tacotron 2` [13] e é responsável por gerar um *mel spectrogram* conforme o vetor de *embeddings* e o texto recebidos como entrada. O terceiro componente é um *vocoder*, responsável por receber o *mel spectrogram* e gerar um áudio como saída, ele foi implementado com base no `WaveRNN` [8] para possibilitar o funcionamento em tempo real.

A Figura 1 ilustra os três componentes com suas respectivas entradas e saídas. No primeiro componente, uma representação digital da voz é criada e, em seguida, no segundo e terceiro componentes, essa representação é utilizada como referência para a geração de fala a partir de um texto arbitrário.

3.2 Transcritor

O transcritor escolhido para ser utilizado nesse trabalho foi o DeepSpeech [5], o qual consiste em um modelo *open-source* fala-para-texto. A arquitetura desse modelo consiste em uma larga Rede Neural Recorrente (`RNN`). Esse modelo é simples, porém, bastante robusto a ruídos de fundo, variação de falantes e reverberação.

O projeto do DeepSpeech fornece modelos pré-treinados, a cada versão, para serem utilizados na realização de inferências ou no treinamento por meio de transferência de aprendizado.

4 METODOLOGIA

Esse trabalho consiste em um estudo qualitativo experimental. As próximas seções detalham os procedimentos executados para a realização dos experimentos e para as análises dos resultados.

4.1 Conjunto de dados

Para a realização das investigações faz-se necessário um conjunto de dados que possua pares de áudios com suas respectivas transcrições. Além disso, os mesmos precisam ter sido gravados em inglês, por falantes que possuam um mesmo sotaque. Em geral, os sotaques das pessoas para um mesmo idioma tendem a variar de região para região. Com isso, para a execução dos experimentos, buscou-se conjuntos de dados em inglês produzidos por indianos, uma vez que tais falantes possuem um sotaque distinto dos falantes estadunidenses e britânicos [1].

O conjunto de dados escolhido para ser utilizado nos experimentos é o NPTEL (NPTEL2020 — Indian English Speech Dataset [2]), produzido a partir da coleta de vídeos no YouTube. Todos os vídeos são em inglês e produzidos por indianos, além disso, a maioria deles é de cunho educacional e possui um sotaque sul-asiático. O áudio de cada vídeo foi extraído em conjunto com sua transcrição, pois, em todos os vídeos coletados, a transcrição estava presente e havia sido carregada manualmente pelo autor.

O conjunto `NPTEL` completo possui 6,2 milhões de trechos de áudio, com a duração média de cada trecho sendo de 3 a 10 segundos. Ele é estruturado no formato do LibriSpeech [10], onde os arquivos de áudio estão no formato `WAV`, as transcrições estão em arquivos de texto e os metadados estão no formato `JSON`.

Como o `NPTEL` não é anotado manualmente pelos autores, não se sabe ao certo se as transcrições de cada vídeo são feitas manualmente ou com o auxílio de algum modelo transcritor. Procurando contornar esse problema, os autores do `NPTEL` resolveram criar uma amostra de mil áudios, onde todos eles são transcritos manualmente pelos próprios autores. Essa amostra é denominada de *Pure-Set*. Por esse motivo, essa porção dos dados foi escolhida para ser utilizada como conjunto de dados para a execução dos experimentos. A Tabela 1 e a Figura 2 mostram algumas informações e metadados importantes acerca desse conjunto.

4.2 Pré-processamento dos dados

4.2.1 Pré-processamento do conjunto de dados. Para realizar o pré-processamento do conjunto de dados foi criado um *script*[2], o qual gera `IDs` únicos e sequenciais para cada um dos arquivos, mantendo a consistência dos `IDs` entre os áudios, transcrições e metadados.

[2]https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/preprocess_nptel-pure.py

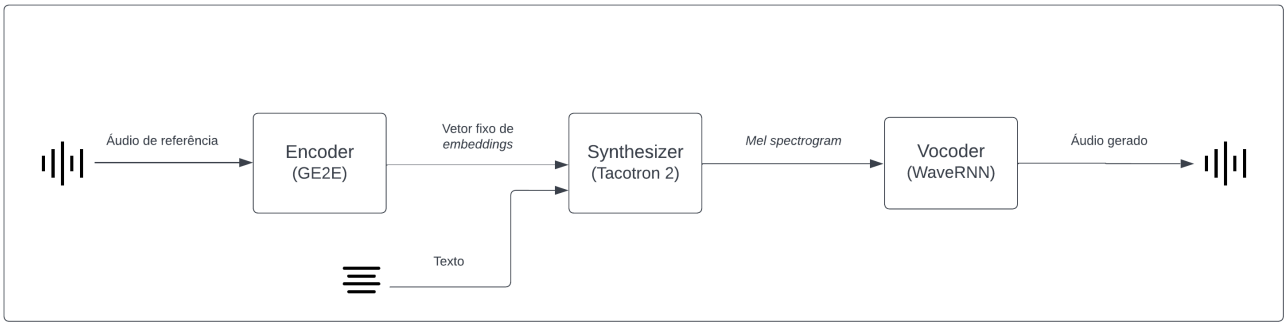

Figura 1: Arquitetura do clonador de voz (Real-Time Voice Cloning)

Tabela 1: Informações sobre o conjunto de dados *Pure-Set*

Metadado	Detalhes
Número de trechos	1000
Duração média dos trechos	7,82 segundos
Total de minutos	130 minutos
Tamanho do conjunto	272 MB

Figura 2: Detalhes da duração dos áudios no conjunto de dados *Pure-Set*

Além disso, ele utiliza a biblioteca `ffmpeg-normalize` [12] para normalizar os áudios, colocá-los na frequência de 16000 Hz e realizar pós-processamentos para remoção de ruído e utilização de filtro passa-alta. Por fim, os áudios que possuem transcrição vazia são removidos do conjunto. Informando, ao final da execução, quais deles foram removidos.

Com esse *script* também é possível criar porções a partir dos dados do conjunto, podendo escolher a quantidade de porções e a quantidade de áudios para cada porção. Os áudios que farão parte de cada porção são sorteados sem repetição. Ao final, todos os áudios do conjunto de dados são separados nas porções desejadas e arquivos de texto são gerados contendo os `IDs` dos áudios para cada porção. Caso algum deles seja removido durante o processo, por possuir a transcrição vazia, a última porção dos dados fica com uma quantidade menor de áudios.

4.2.2 Pré-processamento de dados para treinamento do clonador. Para realizar o treinamento dos modelos *synthesizer* ou *vocoder*, do clonador de voz, é necessário realizar um pré-processamento adicional nos dados. Para tanto, foi criado um *script*[3] cujo objetivo é organizar os áudios que serão utilizados, colocando-os na estrutura de arquivos esperada pelos *scripts* de treinamento do clonador. Ele possui como entrada um arquivo de texto com os `IDs` dos áudios e realiza a cópia dos mesmos, montando a estrutura esperada pelo clonador.

4.2.3 Pré-processamento de dados para uso no transcritor. Dois *scripts* foram criados para realizar o pré-processamento dos dados utilizados na inferência e no treinamento do transcritor. O primeiro[4] é responsável por gerar arquivos no formato `CSV`, conforme o modelo esperado pelo DeepSpeech, para serem utilizados durante o seu treinamento. Para tanto, ele recebe como entrada uma pasta de áudios e a quantidade de áudios que serão separados para validação, processa eles e, ao final, produz os arquivos no formato `CSV` de treino e de validação. Espera-se, nesse primeiro *script*, que a pasta de áudios recebida como entrada contenha áudios gerados pelo clonador. Com isso, esses áudios são analisados e comparados com os áudios originais de suas respectivas transcrições.

Essa comparação serve para descartar áudios gerados com uma qualidade ruim, pois, durante análises manuais, observou-se que os áudios gerados com longa duração, quando comparado aos áudios originais de suas transcrições, tendem a possuir uma qualidade ruim. Os áudios gerados possuem pausas de curta duração durante a fala, enquanto os áudios originais possuem pausas maiores. Além disso, quando o clonador de voz não consegue gerar uma determinada palavra de um áudio, ele tenta gerá-la intermitentemente, produzindo ruídos até chegar na duração máxima estabelecida. Com

[3] https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/dataset_from_ids.py
[4] https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/train-deepspeech/generate_csv_files.py

isso, se um áudio gerado possuir duração maior do que o áudio original, é possível entender isso como indicativo de que o mesmo não foi gerado corretamente. Para realizar essa comparação foram definidos dois atributos.

O primeiro dos atributos se chama `gap_size_percentage` e representa a porcentagem da duração que o áudio gerado precisa ter a mais do que o áudio original para ser descartado. Por exemplo, utilizando 50% como valor desse atributo e considerando que o áudio original tem cinco segundos, o áudio gerado precisa ter 7,5 segundos, ou mais, para ser descartado. No entanto, durante alguns testes utilizando esse atributo, percebeu-se que, quando os áudios originais eram pequenos e os gerados um pouco maior do que eles, os mesmos estavam sendo descartados quando, na verdade, não deveriam ser. Por exemplo, considerando 50% como valor do atributo e um áudio original com duração de dois segundos, áudios gerados a partir da transcrição desse áudio original, com durações de três segundos ou mais, eram descartados. Porém, analisando-os, foi percebido que a diferença de apenas um segundo entre os áudios gerados e o original descartava áudios que não possuíam qualidade ruim.

Buscando mitigar esse problema, foi adicionado um segundo atributo para ser utilizado durante a comparação, esse atributo se chama `gap_size`. Ele indica a duração que o áudio gerado precisa ter a mais que o áudio original para ser descartado. Por exemplo, considerando que o áudio original tem sete segundos e utilizando cinco como valor do atributo, o áudio gerado precisa ter 12 segundos ou mais para poder ser descartado.

O descarte do áudio gerado só é realizado quando ele é maior que o áudio original de sua transcrição, considerando os dois atributos na comparação. Dessa forma, os áudios descartados possuem grandes probabilidades de realmente serem áudios gerados com qualidade ruim. Por fim, um arquivo de texto é gerado com as informações acerca dos áudios descartados, mostrando os valores dos atributos utilizados durante a comparação, os áudios descartados com suas respectivas durações e as durações dos áudios originais de cada transcrição e, no final, a quantidade de áudios descartada.

Após o descarte, são sorteados, sem repetição, os áudios que farão parte da porção de validação e, em seguida, o restante fará parte da porção de treino. Posteriormente, cada transcrição passa por um pré-processamento, transformando as letras do texto em minúsculas e os números em palavras por extenso, por exemplo, o número **1** é transformado em **one**. Por fim, cada arquivo de validação e treino é criado no formato `CSV` e no modelo esperado pelo DeepSpeech, com seus respectivos áudios e transcrições.

O outro *script*[5] possui o funcionamento similar ao descrito anteriormente. Ele é responsável por gerar arquivos no formato `CSV` no modelo esperado pelo DeepSpeech, para áudios que não foram gerados pelo clonador. Por exemplo, ele pode ser utilizado para gerar o arquivo de teste no formato desejado, com os áudios e as transcrições que serão utilizadas para testar o transcritor. Esse *script* realiza o mesmo pré-processamento da transcrição que o anterior, criando o arquivo no formato `CSV` ao final.

[5]https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/train-deepspeech/create_csv_file.py

4.3 Treinamento do clonador de voz

Após o pré-processamento realizado pelo *script* detalhado na seção 4.2.2, é possível utilizar os dados pré-processados para o treinamento dos modelos utilizados pelo clonador de voz. No entanto, o conjunto de dados escolhido para ser utilizado nesse trabalho possui apenas os áudios e suas respectivas transcrições, dessa forma, como explicado na seção 3.1, só é possível treinar os modelos *synthesizer* e *vocoder* do clonador.

Para realizar os treinamentos desses modelos são utilizados os *scripts* disponíveis no repositório do Real-Time Voice Cloning [6]. Além disso, é utilizado um passo-a-passo[6], também disponibilizado no mesmo repositório, onde é ensinado quais *scripts* utilizar e em qual ordem. Para o treinamento do modelo *synthesizer* é realizado um pré-processamento dos dados utilizando os *scripts* cujo prefixo é `synthesizer_preprocess`, por fim, é utilizado o `synthesizer_train.py` para o treinamento do mesmo. Ademais, para o treinamento do modelo *vocoder* é realizado o mesmo pré-processamento do modelo *synthesizer* e, em seguida, realizado um pré-processamento específico do *vocoder*, onde o *script* executado é o `vocoder_preprocess.py`, por fim, o treinamento é realizado por meio do `vocoder_train.py`.

4.4 Geração de áudios

Para a geração de áudios utilizando o clonador de voz, foi necessário a criação de dois *scripts*, um principal[7] e o outro auxiliar[8]. O principal recebe como entrada um arquivo de texto com os `IDs` dos áudios utilizados como áudios de referência, assim como a quantidade limite de áudios que serão gerados a partir de cada áudio de referência. Em seguida, para cada áudio de referência, é sorteado a quantidade limite de outros áudios de referência que terão suas transcrições utilizadas na geração dos novos áudios.

Por exemplo, considerando que o *script* principal receba oito áudios de referência e a quantidade limite com valor cinco, são gerados cinco novos áudios para cada um dos oito áudios de referência. O texto desses novos áudios são as transcrições sorteadas, sem repetição, dos outros áudios de referência, sem incluir o atual. A Figura 3 ilustra um passo desse exemplo, onde o áudio 3 é o áudio de referência e as transcrições destacadas são as que foram sorteadas para a geração de novos áudios, utilizando como referência para a clonagem a voz presente no áudio 3. Portanto, a quantidade limite de áudios gerados a partir de cada áudio de referência deve ser menor que a quantidade total de áudios, considerando que a transcrição do próprio áudio de referência não é utilizada na geração dos novos áudios.

Possuindo os áudios referências e suas respectivas transcrições que serão usadas na geração dos novos áudios, é utilizado o *script* auxiliar para a aplicação do clonador de voz. Ele foi criado tendo como base o *script* denominado `demo_cli.py`[9] do repositório do Real-Time Voice Cloning [6], com ele é possível realizar as inferências aos três modelos do clonador na ordem certa, permitindo a clonagem de voz. Durante o processo, alguns áudios que seriam

[6]https://github.com/CorentinJ/Real-Time-Voice-Cloning/wiki/Training
[7]https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/generate_audios.py
[8]https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/voice_cloning_inferences.py
[9]https://github.com/CorentinJ/Real-Time-Voice-Cloning/blob/master/demo_cli.py

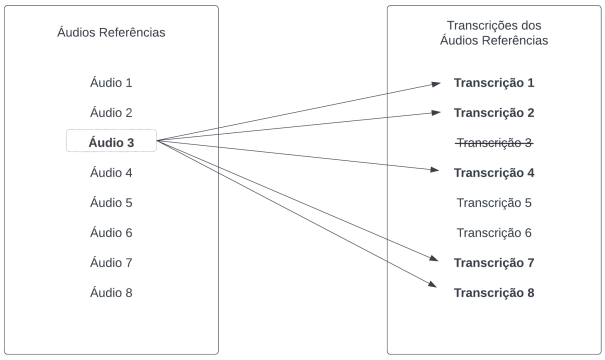

Figura 3: Ilustração de um passo do processo de geração de novos áudios

gerados podem ser descartados caso aconteça algum erro ou caso o modelo *synthesizer* tenha gerado um *mel spectrogram* muito pequeno.

4.5 Treinamento do transcritor

Para o treinamento do modelo de transcrição do DeepSpeech é necessário realizar o pré-processamento dos dados conforme descrito na seção 4.2.3. Após o pré-processamento e a geração dos arquivos necessários no formato `CSV`, o treinamento é realizado a partir do repositório[10] e dos modelos pré-treinados[11] fornecidos pelo DeepSpeech. Os comandos utilizados para o treinamento do transcritor estão em um arquivo de texto chamado `training-commands`[12]. Por padrão, quando o treinamento é conduzido por inúmeras épocas, o DeepSpeech, ao final de cada época, utiliza o conjunto de validação e calcula uma métrica de *loss*, salvando o modelo que possuir o menor valor. Dessa forma, ao final do treinamento, o modelo com o menor *loss* ao longo das épocas é o modelo salvo.

4.6 Inferências no transcritor

Para realizar inferências no transcritor foi produzido um *script*[13] que facilita esse processo. Esse *script* recebe como entrada o modelo, o *scorer* e um arquivo no formato `CSV`, conforme esperado pelo DeepSpeech, com os áudios que são utilizados para realizar as inferências e as transcrições originais de cada áudio.

Para a avaliação das transcrições produzidas pelo mesmo, foi escolhida a métrica *Word Error Rate* [3] (`WER`), pois ela é comumente utilizada nas avaliações de desempenho de transcritores e considera, no seu cálculo, possíveis palavras deletadas ou adicionadas, além de palavras que foram substituídas por outras. Com isso, após as inferências, as transcrições originais e as transcrições geradas pelo transcritor são utilizadas para calcular o `WER` médio de todas as inferências realizadas.

[10] https://github.com/mozilla/DeepSpeech
[11] https://github.com/mozilla/DeepSpeech/releases/tag/v0.9.3
[12] https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/train-deepspeech/training-commands.txt
[13] https://github.com/alexandrerf3/tcc-investigacao-deepfake-aumento-de-dados/blob/main/deepspeech/inferences_deepspeech.py

5 RESULTADOS E DISCUSSÃO

Nesta seção são mostrados os experimentos realizados, os resultados obtidos e as discussões acerca dos mesmos. Para tanto, é utilizado o pré-processamento dos dados descritos na seção 4.2.1 para a criação de dois experimentos utilizando o mesmo conjunto de dados. Os experimentos consistem na geração de áudios, no treinamento do transcritor com os áudios gerados e na avaliação do mesmo antes e depois do treinamento. O segundo experimento, diferente do primeiro, realiza o treinamento dos modelos do clonador buscando uma melhora nos resultados.

5.1 Experimento 1

Para esse experimento, o conjunto de dados é pré-processado e dividido em duas porções de 500 e 498 áudios cada (a porção número 2 teve dois áudios a menos devido ao descarte efetuado durante o pré-processamento). Com isso, a porção número 1 é utilizada para a avaliação das transcrições geradas antes e depois do treinamento e a porção número 2 para a geração de novos áudios, utilizados no treinamento do transcritor. A Figura 4 ilustra todo o passo a passo desse experimento.

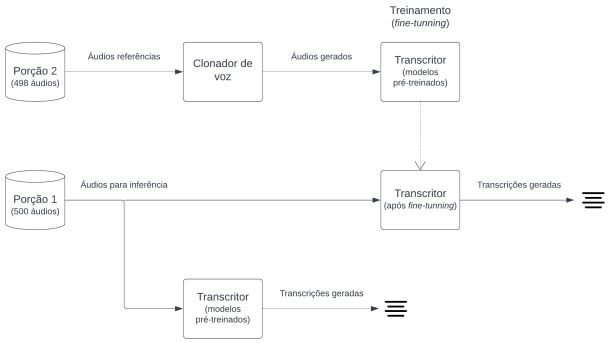

Figura 4: Ilustração do passo a passo realizado no experimento 1

Como visto na Figura 4, a porção número 2 é utilizada para a geração de novos áudios, com isso, os 500 áudios servem como áudios de referência e 21 é o valor da quantidade limite utilizado, tendo, como resultado, a geração de 10.458 áudios. Em seguida, os áudios gerados são utilizados no treinamento do transcritor em diversos cenários, cada um sendo realizado durante 200 épocas. Para cada cenário é alterado um hiper-parâmetro diferente, buscando um treinamento com melhor resultado. O *dropout* é utilizado com o valor padrão e com o valor definido em 0,4. Além disso, o *scorer* também é usado em um dos cenários.

A porção número 1, é utilizada na realização de inferências ao transcritor para avaliá-lo. Primeiramente, são realizadas inferências ao transcritor com o modelo pré-treinado e as transcrições geradas são utilizadas para calcular a métrica `WER`. Após cada treinamento do modelo, a porção é utilizada novamente realizando novas inferências e calculando um novo valor da métrica `WER`.

A Tabela 2 exibe os diferentes cenários de treinamento, com as variações dos hiper-parâmetros e os resultados do `WER` obtidos para

Tabela 2: Treinamento e avaliação do transcritor no experimento 1

Cenário	*Dropout*	*Scorer*	WER
Pré-treinado	-	-	**0,636**
fine-tunning	padrão	não	0,657
fine-tunning	0,4	não	0,709
fine-tunning	padrão	sim	0,681

cada um. Após o *fine-tunning* do modelo transcritor, o resultado do WER piorou em relação ao resultado do modelo pré-treinado, mesmo com as variações dos hiper-parâmetros. Ao analisar os resultados, percebeu-se que os áudios gerados não possuem uma boa qualidade, sendo alguns total ou parcialmente incompreensíveis, sendo essa uma provável causa da piora nos resultados.

5.2 Experimento 2

Nesse experimento é realizado o treinamento dos modelos *synthesizer* e *vocoder* do clonador de voz, buscando uma melhora na qualidade dos áudios gerados. Para tanto, o conjunto de dados é pré-processado e dividido em três porções com 200, 300 e 498 áudios cada (a porção número 3 possui dois áudios a menos devido ao descarte realizado no pré-processamento). Com isso, a porção número 1 é utilizada para a geração de novos áudios, a porção número 2 para a avaliação do transcritor antes e depois do treinamento e a porção número 3 é usada no treinamento dos modelos do clonador de voz, conforme ilustrado na Figura 5.

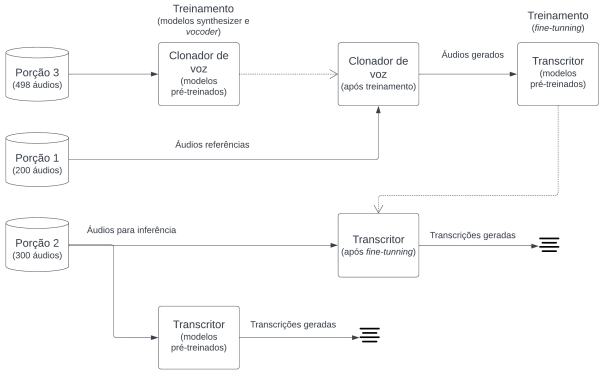

Figura 5: Ilustração do passo a passo realizado no experimento 2

Para a realização do treinamento dos modelos do clonador de voz é necessário um pré-processamento adicional, conforme explicado na seção 4.3. Com isso, durante esse pré-processamento, quatro áudios foram descartados dos 498 da porção número 3, restando 494 áudios para serem utilizados no treinamento.

Foram realizados diversos treinamentos dos modelos *synthesizer* e *vocoder* do clonador de voz, por meio de combinações de *fine-tunning* e de re-treinamento dos mesmos. A Tabela 3 provê detalhes acerca desses treinamentos, mostrando os modelos pré-treinados disponibilizados pelo autor como o padrão e as combinações de treinamento realizadas. Além disso, informa também por quantos passos cada modelo foi treinado, destacando a quantidade do modelo treinado em cada combinação.

Após o treinamento dos modelos nas diversas combinações, houve a necessidade de aferir a qualidade dos áudios gerados por cada combinação. Para tanto, é realizada uma análise qualitativa dos mesmos. Para a realização dessa análise, é separada uma amostra de 10 áudios, onde os mesmos são passados como referências e utilizado nove como quantidade limite, gerando, ao final, 90 áudios. A qualidade dos áudios é avaliada manualmente e os mesmos são classificados dentre três categorias: **ruim**, **razoável** e **bom**. Além disso, uma pontuação para cada combinação de modelos é calculada a partir das classificações recebidas, onde **ruim** equivale a um ponto, **razoável** a dois pontos e **bom** a três pontos.

Em princípio, essas análises são realizadas nas combinações de treinamento onde pelo menos um dos modelos é re-treinado. Conforme observado nas visualizações da Figura 6 e na Tabela 4, as combinações de modelos re-treinados que obtiveram melhores resultados são as denominadas `sys_zero_voc` e `sys_treinado_zero_voc`, os resultados das demais são extremamente ruins.

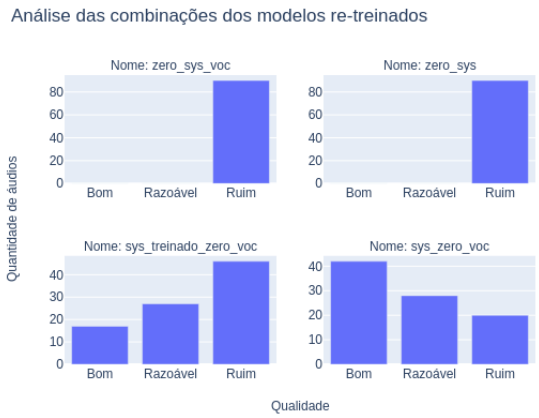

Figura 6: Análise qualitativa dos modelos re-treinados

Na próxima análise, as combinações que obtiveram melhores resultados nessa análise anterior são consideradas em conjunto com as demais combinações que não teve seus modelos re-treinados. Outrossim, durante essa primeira análise observou-se que os áudios gerados com uma longa duração tendem a possuir uma qualidade ruim. Para tanto, a próxima análise classifica a duração do áudio como padrão ou longa, buscando verificar se os áudios com longa duração realmente tendem a serem ruins.

Após a última análise qualitativa dos modelos, é possível observar nas visualizações da Figura 7 e na Tabela 5 que as combinações de modelos que obtiveram melhores resultados são as denominadas `padrão` e `sys_zero_voc`. A combinação de modelos denominada `padrão` já foi utilizada no experimento 1 (5.1) para a geração de áudios, treinamento do transcritor e análise dos resultados. Portanto, nesse experimento é utilizada a combinação de modelos denominada `sys_zero_voc` para a realização das investigações.

Tabela 3: Treinamento dos modelos *synthesizer* e *vocoder* do clonador de voz

Nome	*Synthesizer*	*Vocoder*	Quantidade de passos
padrão	pré-treinado	pré-treinado	295 mil e 1 milhão 159 mil
sys_treinado	***fine-tunning***	pré-treinado	**327 mil** e 1 milhão 159 mil
sys_voc_treinado	treinado (sys_treinado)	***fine-tunning***	327 mil e **1 milhão 160 mil**
sys_treinado_zero_voc	treinado (sys_treinado)	**re-treinado**	327 mil e **18 mil**
zero_sys	**re-treinado**	pré-treinado	**100 mil** e 1 milhão e 159 mil
zero_sys_voc	treinado (zero_sys)	**re-treinado**	100 mil e **13 mil**
sys_zero_voc	pré-treinado	**re-treinado**	295 mil e **48 mil**

Tabela 4: Pontuações da análise qualitativa dos modelos re-treinados

Nome	Pontuação
zero_sys_voc	90 pontos
zero_sys	90 pontos
sys_treinado_zero_voc	**151 pontos**
sys_zero_voc	**202 pontos**

Tabela 5: Pontuações da análise qualitativa dos modelos

Nome	Pontuação
padrão	**207 pontos**
sys_voc_treinado	165 pontos
sys_treinado	148 pontos
sys_treinado_zero_voc	142 pontos
sys_zero_voc	**203 pontos**

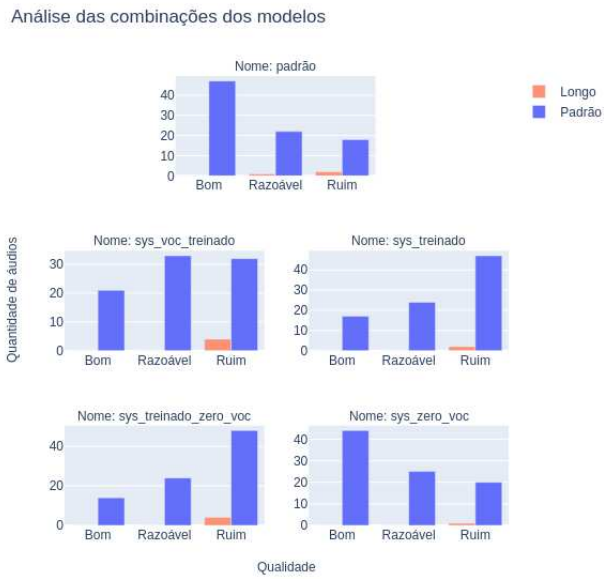

Figura 7: Análise qualitativa dos modelos

Além disso, essa última análise permitiu verificar se os áudios com longa duração tendem a possuir a qualidade ruim. Dessa forma, observando a Figura 7 é possível afirmar que a maioria dos áudios com longa duração realmente possuem a qualidade ruim. Portanto, o descarte dos áudios com longa duração é válido, onde o mesmo é feito durante o pré-processamento dos áudios gerados, antes de serem utilizados no modelo transcritor, conforme descrito na seção 4.2.1.

Os modelos da combinação denominada `sys_zero_voz` são então utilizados no clonador de voz para a geração de novos áudios, a partir dos 200 áudios referências da porção número 2 e a quantidade limite no valor de 52, produzindo, ao total, 10.400 áudios. O valor da quantidade limite é definido em 52 visando gerar uma quantidade de áudios aproximada do que foi gerado e utilizado no experimento 1.

Os áudios gerados são então utilizados no treinamento do transcritor em diferentes cenários, variando os hiper-parâmetros do mesmo conforme foi conduzido no experimento 1 (5.1).

A porção número 3 dos dados é utilizada para avaliar o modelo transcritor antes e depois dos treinamentos, nos diferentes cenários. Os 300 áudios da porção são utilizados para realizar inferências aos diversos modelos, calculando o `WER` para cada um. Na tabela 6 é possível observar os modelos, as variações dos hiper-parâmetros e o valor do `WER` em cada modelo.

Tabela 6: Treinamento e avaliação do transcritor no experimento 2

Cenário	*Dropout*	*Scorer*	`WER`
Pré-treinado	-	-	**0,648**
fine-tunning	padrão	não	0,710
fine-tunning	0,4	não	0,742
fine-tunning	padrão	sim	0,711

Após o *fine-tunning* do modelo transcritor, utilizando os áudios gerados pelo clonador de voz com os novos modelos, as transcrições pioraram significativamente e o valor da métrica `WER` aumentou cerca de seis por cento. Uma provável causa para a piora nos resultados é a qualidade dos áudios gerados, que, mesmo após várias

tentativas de treinamento dos modelos do clonador de voz, continuam com uma qualidade ruim.

Uma opção para a melhora dos resultados é a troca do clonador de voz. No entanto, para a realização das investigações e dos experimentos desse trabalho, é necessário um clonador de voz capaz de clonar uma voz a partir de poucos segundos de um áudio de referência. Com isso, o Real-Time Voice Cloning [6] foi o único encontrado com o código disponível para uso gratuito. Os demais, que relatavam uma maior qualidade na clonagem de voz, não possuem seus códigos disponíveis devido ao possível mal uso dos mesmos. Além disso, alguns relatam que os códigos só serão divulgados após a existência de bons detectores de áudios criados a partir de técnicas de *deepfake audio*.

Além disso, os autores da arquitetura SV2TTS [7], utilizada no Real-Time Voice Cloning, apontam que a maneira mais eficiente e eficaz de melhorar a qualidade dos áudios gerados é treinar o modelo *encoder*, como pode ser observado arquitetura do clonador na Figura 1. No entanto, durante a realização desse trabalho, não foi possível realizar o treinamento do mesmo, pois, para isso, necessita-se de um conjunto de dados onde exista a informação dos falantes para cada áudio, entretanto, o conjunto utilizado nesse trabalho não possui essas informações.

Outro fator, que possivelmente influência na não melhora do clonador após os treinamentos, é que os áudios do conjunto de dados utilizados nesse trabalho são ruidosos, pois são extraídos de vídeos do YouTube[14] gravados em diversos ambientes, com diferentes equipamentos de gravação. Ademais, como a maioria dos vídeos são educativos, grande parte das falas presentes nos áudios possuem um linguajar técnico de acordo com o conteúdo ensinado. Alguns exemplos de transcrições dos áudios, observadas durante a análise manual, podem ser vistas na Tabela 7.

Tabela 7: Exemplos de transcrições dos áudios do conjunto de dados utilizado nos experimentos

Transcrição
NOW THIS PREFERENTIAL FLOW OF CURRENT IN A DIODE IS UTILISED TO CONVERT AN AC TO DC SUPPOSE
AND WHAT IS FIRST OF F FIRST OF F IS LEFT PARENTHESIS AND IDEALS SO
Y 1 ONE D X IF THE DIVIDED DIFFERENCE TERM IS ZERO ERROR IS GOING TO BE EQUAL TO ZERO

Esses dados, cujo conteúdo utiliza uma linguagem mais técnica, não são comumente vistos em conjuntos de dados. Portanto, é muito provável que os modelos pré-treinados do clonador de voz e da transcrição não tenham sido treinados com tais palavras mais técnicas.

6 CONCLUSÕES E TRABALHOS FUTUROS

Para a realização das investigações e experimentos desse trabalho, utilizando *deepfake audio* como uma técnica de aumento de dados, procurou-se um conjunto de dados no idioma inglês com a presença única do sotaque indiano, onde o mesmo possuísse pares de áudios com suas respectivas transcrições. Além disso, foi necessário encontrar um clonador de voz capaz de clonar vozes a partir de poucos segundos de um áudio de referência. Com isso, torna-se possível aumentar os dados do conjunto escolhido.

Com o conjunto de dados aumentado, foi necessário verificar se a utilização do mesmo no treinamento de um transcritor resultaria em transcrições geradas com maior qualidade. Para tanto, foi utilizado o modelo transcritor chamado DeepSpeech para a realização das investigações. Selecionando um subconjunto dos dados, foram realizadas inferências ao transcritor e foi medida a qualidade de suas transcrições produzidas a partir do cálculo de uma métrica chamada WER. Em seguida, após o treinamento do modelo transcritor utilizando os dados aumentados, o mesmo subconjunto foi utilizado na realização de novas inferências, buscando medir a qualidade das transcrições produzidas após o treinamento e verificar se houve uma melhora, ou não, nos resultados.

Com os experimentos realizados nesse trabalho, não foram observadas melhoras na qualidade das transcrições geradas após o treinamento do modelo transcritor. Apesar de o transcritor ter sido treinado em diversos cenários, todos apresentaram piora na qualidade da transcrição. Um provável motivo para esses resultados é a qualidade dos áudios gerados pelo clonador, pois, análises manuais do mesmo verificaram uma qualidade ruim nos áudios gerados. Mesmo após o treinamento de alguns dos modelos utilizados pelo clonador de voz, a qualidade dos áudios gerados continuaram insatisfatórias. Com isso, os áudios gerados com uma qualidade ruim devem estar atrapalhando o aprendizado do modelo transcritor.

Na tentativa de obter melhores resultados, um trabalho futuro que poderá ser realizado é a melhora na qualidade dos áudios gerados. Para tanto, pode-se buscar um melhor treinamento dos modelos do clonador de voz, treinando o *synthesizer* e o *vocoder* com mudanças nos seus hiper-parâmetros ou em suas arquiteturas. Além disso, pode-se procurar um conjunto de dados, com os áudios no idioma inglês e com a presença única do sotaque indiano, que possua a identificação dos falantes, para que, com isso, seja possível o treinamento do modelo *encoder* do clonador.

Ademais, pode-se buscar uma solução para o descarte dos áudios gerados com uma qualidade ruim. Esse descarte pode ser feito a partir da análise do espectrograma do áudio gerado ou utilizando modelos treinados com esse propósito. Adicionalmente, é possível a utilização de um novo clonador de voz, encontrando um que seja adequado para a realização dos experimentos ou criando um para essa finalidade.

O conjunto de dados encontrado e utilizado nesse trabalho possui algumas características que dificultam a geração dos áudios e das transcrições, como o ruído de fundo e o linguajar mais técnico. Uma possível melhoria, para a realização dos experimentos, é encontrar ou produzir um conjunto de dados com uma quantidade maior de áudios, onde os mesmos possuam menos ruídos e um linguajar menos técnico.

[14] YouTube — www.youtube.com

AGRADECIMENTOS

Primeiramente, gostaria de agradecer a Deus por me permitir e conceder saúde para vivenciar toda essa experiência da graduação. Agradeço imensamente aos meus pais, Tarcísio Filho e Elaine Cristina, por me ajudarem, me apoiarem, sempre se esforçaram para que eu pudesse ter bons estudos e que estão sempre disponíveis para me amparar. As minhas irmãs, Júllia e Dafne, e a minha família, por todo apoio que tive e tenho. Agradeço também a minha namorada, Sara Ariadne, por todo o apoio, motivação e ajuda nessa jornada, com certeza seria mais difícil sem a sua ajuda, ela é uma pessoa incrível que admiro muito.

Agradeço ao professor Cláudio Campelo, por toda sua paciência, orientação e ajuda durante esse trabalho. Além de todas as oportunidades e ensinamentos ao longo desses últimos quatro anos de graduação. Agradeço aos amigos que fiz durante o curso e que me ajudaram a ter uma graduação mais tranquila e proveitosa, além de terem se tornado amigos para a vida. Agradeço a Universidade Federal de Campina Grande e a todos os professores do curso de Ciência da Computação que fizeram e fazem um ótimo trabalho, mantendo o curso como um dos melhores do Brasil.

REFERÊNCIAS

[1] 2017. Indian accents- just another version of British English? https://www.accentreductionaustin.com/indian-accents-just-another-version-of-british-english/

[2] 2020. NPTEL2020 - Indian English Speech Dataset. https://github.com/AI4Bharat/NPTEL2020-Indian-English-Speech-Dataset

[3] 2020. Word error rate. https://en.wikipedia.org/wiki/Word_error_rate

[4] Cem Dilmegani. 2021. Top 11 speech recognition applications in 2022. https://research.aimultiple.com/voice-recognition-applications/

[5] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567* (2014).

[6] Corentin Jemine. 2022. Real-Time Voice Cloning. https://github.com/CorentinJ/Real-Time-Voice-Cloning

[7] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems* 31 (2018).

[8] Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. 2018. Efficient Neural Audio Synthesis. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Jennifer Dy and Andreas Krause (Eds.), Vol. 80. PMLR, 2410–2419. https://proceedings.mlr.press/v80/kalchbrenner18a.html

[9] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. 2015. Audio augmentation for speech recognition. In *Sixteenth annual conference of the international speech communication association.*

[10] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 5206–5210. https://doi.org/10.1109/ICASSP.2015.7178964

[11] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779* (2019).

[12] Werner Robitza. 2022. ffmpeg-normalize: Audio normalization for python/ffmpeg. https://github.com/slhck/ffmpeg-normalize

[13] Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj Skerrv-Ryan, et al. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 4779–4783.

[14] Xingcheng Song, Zhiyong Wu, Yiheng Huang, Dan Su, and Helen Meng. 2020. SpecSwap: A Simple Data Augmentation Method for End-to-End Speech Recognition.. In *Interspeech*. 581–585.

[15] Rodolfo Zevallos. 2022. Text-To-Speech Data Augmentation for Low Resource Speech Recognition. *arXiv preprint arXiv:2204.00291* (2022).