UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Análise comparativa de redes profundas para reconhecimento automático de fala em português

Antonio Alves Lopes Filho

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Antonio Alves Lopes Filho

Análise comparativa de redes profundas para reconhecimento automático de fala em português

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientadora: Prof. Dr. Fernando Pereira do Santos

Versão original

São Carlos

2022

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Lopes, Antonio
L627a Análise comparativa de redes profundas para reconhecimento automático de fala em português / Antonio Lopes; orientador Fernando Santos. -- São Carlos, 2022.
51 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2022.

1. ASR. 2. Transformers. 3. Wav2Vec2. 4. DeepSpeech2. 5. CoquiSTT. I. Santos, Fernando, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Antonio Alves Lopes Filho

Comparative Analysis of Artificial Neural Networks for Automatic Speech Recognition in Portuguese

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Fernando Pereira

Original version

São Carlos

2022

RESUMO

Lopes, A.A **Análise comparativa de redes profundas para reconhecimento automático de fala em português**. 2022. 51p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2022.

Reconhecimento automático de fala é um processo que visa gerar uma saída em texto a partir de uma entrada em áudio. Mecanismos de inteligência artificial tem sido empregados de forma efetiva como soluções para esta tarefa, com diferentes abordagens baseadas em redes profundas. Wav2Vec2 é uma arquitetura fundamentada nos conceitos de mecanismos de atenção e *Transformers*, enquanto DeepSpeech2/CoquiSTT é construída usando redes recorrentes, LSTM e GRU. Contudo, essas arquiteturas foram projetadas e validadas inicialmente na língua inglesa. Sendo assim, torna-se pertinente a avaliação dessas arquiteturas para dados em língua portuguesa. Para essa avaliação foi utilizado um conjunto de dados público, Common Voice, e outro particular, coletado e catalogado manualmente. De forma geral, Wav2Vec2 superou a performance da arquitetura DeepSpeech2/CoquiSTT em todos os resultados, inclusive quando aplicado ruídos para dificultar a performance. Assim, é notável que o modelo Wav2Vec2 é mais adaptável para sistemas comerciais por ter um melhor desempenho mesmo treinando com poucos dados e por ser menos complexo na quantidade de parâmetros treináveis.

Palavras-chave: Reconhecimento de fala. DeepSpeech2. Wav2Vec2. Transformers. LSTM. CoquiSTT. CommonVoice.

ABSTRACT

Lopes, A.A **Comparative Analysis of Artificial Neural Networks for Automatic Speech Recognition in Portuguese**. 2022. 51p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2022.

Automatic speech recognition is a process that aims to generate a text output from an audio input. Artificial intelligence mechanisms have been used effectively as solutions for this task, with different approaches based on deep neural networks. Wav2Vec2 is an architecture based on the concepts of attention mechanisms and *Transformers*, while DeepSpeech2/CoquiSTT is built using recurrent networks, LSTM and GRU. However, these architectures were initially designed and validated in the English language. Therefore, the evaluation of these architectures for data in Portuguese becomes relevant. For this evaluation, a public dataset, Common Voice, and a private dataset, collected and cataloged manually, were used. Overall, Wav2Vec2 outperformed the DeepSpeech2/CoquiSTT architecture in all results, including when noise was applied to hinder performance. Thus, it is notable that the Wav2Vec2 model is more adaptable to commercial systems because it performs better even when training with few data and because it is less complex in terms of the number of trainable parameters.

KeywordeX: Automatic Speech Recognition. ASR. DeepSpeech2. Wav2Vec2. Transformers. LSTM. CoquiSTT. CommonVoice.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Sistemas de reconhecimento automático de fala com processamento de linguagem natural é um tópico bastante antigo em Ciência da Computação e Engenharia Elétrica (O'SHAUGHNESSY, 2008) e dependem de diversas áreas do conhecimento, como processamento de sinais, acústica, fonologia, sintaxe, semântica e discursos (JURAFSKY; MARTIN, 2008). A primeira máquina de reconhecimento de fala desenvolvida data da década de 1950, mais precisamente em 1952, no *Bell Labs*, em que era capaz de reconhecer quaisquer 10 dígitos de um único vocalizador (JURAFSKY; MARTIN, 2008).

Com o passar do tempo e com o poder computacional atingindo custos financeiros cada vez menores, bem como a alta demanda comercial por clientes empresariais e domésticos, o estudo e disponibilização de ferramentas de reconhecimento de fala se popularizaram, em que podemos citar: geração automática de legendas em reuniões virtuais, identificação de palavras-chave via telefone em empresas de *contact center* e *chatbots* para melhor atendimento de clientes que navegam no seu respectivo sistema de atendimento. Para clientes domésticos, existem vários exemplos, sendo o principal, assistentes virtuais como a Alexa™e Google Assistente™, que possuem, principalmente, como método entrada de dados a fala dos usuários.

Hoje existem várias técnicas para implementar um sistema de reconhecimento de fala, sendo uma técnica bastante tradicional e usada durante um bom tempo, a Cadeia Oculta de Markov (*Hidden Markov Model*) (O'SHAUGHNESSY, 2008), e também técnicas mais contemporâneas, como o aprendizado supervisionado, que usam *Deep Learning*, por meio de Redes Neurais Convolucionais (CNNs) e Redes Neurais Recorrentes (GEORGESCU *et al.*, 2021). Dentre as técnicas mais recentes, *DeepSpeech2* (HANNUN *et al.*, 2014)[1] aplica conceitos de Redes Neurais Recorrentes e *Wav2Vec2* (BAEVSKI *et al.*, 2020) se fundamenta na arquitetura *Transformers*.

Neste projeto de pesquisa foram aplicados dois modelos, *DeepSpeech2* e *Wav2Vec2*, para comparação da qualidade de um sistema de reconhecimento de fala em português. Estruturalmente esses modelos são treinados em língua inglesa e, portanto, o objetivo é medir e estudar a sua influência em outro idioma.

1.1 Questões de Pesquisa

Neste estudo espera-se que, com a comparação do modelo estruturado sobre redes neurais artificiais usando *Wav2Vec2* e o *DeepSpeech2* possamos indicar o que gera melhor precisão na transcrição dos áudios. Diante dos desafios e problemas atualmente enfrentados

[1] https://github.com/mozilla/DeepSpeech

em sistemas de reconhecimento de fala, foi elaborada a seguinte questão de pesquisa que norteará este projeto:

Q1 *"Algoritmos de redes neurais profundas que usam Transformers tem desempenho melhor que uma aplicação que usa redes neurais recorrentes (RNNs) na transcrição de áudios em português? Ou seja, possui taxa de erro similar ou inferior?"*

Diante desta questão de pesquisa foram definidos os seguintes objetivos para o desenvolvimento deste trabalho:

- Mapear algoritmos de redes neurais profundas na literatura, bem como analisar ferramentas comerciais prontas que já usam esses algoritmos em busca de analisar sua eficácia em conjunto de dados público e privado que será usado na avaliação dos modelos.
- Comparar os desempenhos obtidos pelos algoritmos *Wav2Vec2* e o *DeepSpeech2*.

A partir do modelo proposto espera-se que os resultados sejam mais favoráveis aos algoritmos de redes neurais usando *Transformers* para o mesmo conjunto de treinamento.

No próximo capítulo, será apresentado a fundamentação teórica para desenvolvimento dos modelos que serão usados. Passaremos pela definição de Redes Neurais, CNN, RNN e *Transformers*.

2 REFERENCIAL TEÓRICO

2.1 RECONHECIMENTO AUTOMÁTICO DE FALA

Reconhecimento automático de fala, ou *Automatic Speech Recognition* (ASR), diz respeito a sistemas capazes de transcrever áudio em texto. Esta técnica envolve a superação de vários desafios para ser bem executada, como ignorar o ruído nos áudios, compreender as diversas formas de falar uma mesma palavra dependendo do interlocutor, velocidade da narração, entonação das palavras, sotaques e qualidade do dispositivo de gravação (CUTAJAR *et al.*, 2013). Antes do advento do aprendizado profundo, a construção de sistemas de reconhecimento de fala envolviam vários componentes para o seu funcionamento ser bem executado, tais como (CUTAJAR *et al.*, 2013):

- Modelo de linguagem, que consiste numa espécie de base de dados contendo a distribuição de probabilidade de um conjunto de palavras. Este, ainda pode ser utilizado como complementação hoje em dia nos modelos baseados em redes neurais profundas.

- Extrair o *Mel Frequency Cepstral Coefficient* (MFCC) que consiste em um método de extração de características dos áudios. O MFCC se inspira no ouvido humano, onde, dado o espectro de áudio, as suas frequências são resolvidas de forma não linear (CUTAJAR *et al.*, 2013).

- Por fim, o Modelo Oculto de Markov, em inglês *Hidden Markov Model* (HMM), usado prioritariamente como um classificador em sistemas de reconhecimento de fala, este método consiste em encontrar a probabilidade de que um enunciado de fala fora gerado pela pronúncia de um fonema ou palavra em particular (CUTAJAR *et al.*, 2013).

2.1.1 PRINCIPAIS MÉTRICAS

O desempenho do reconhecimento automático de voz pode ser mensurado com algumas funções matemáticas, que também podem ser chamadas métricas de avaliação. Este estudo aplica as seguintes métricas: WER (*Word Error Rate*) e CER (*Character Error Rate*).

O WER é uma das métricas mais usadas em sistemas de reconhecimento de voz (TEVAH, 2006). O WER calcula a quantidade de palavras inseridas (I) incorretamente, as palavras quer foram substituídas (S) e as que foram excluídas (D) sobre a quantidade de palavras (N), quando comparadas com a frase de referência. Essa taxa é calculada pela

seguinte Equação 2.1:

$$WER = \frac{S + I + D}{N} \tag{2.1}$$

O CER é similar ao WER, porém este considera os caracteres ao invés das palavras. Nesse caso, o N é o número de caracteres de referência (FENG *et al.*, 2017). A equação do CER é dado por 2.2:

$$CER = \frac{S + D + I}{N} \tag{2.2}$$

2.1.2 ESTADO DA ARTE

Com a adoção maciça de redes neurais profundas, logo foi percebido ser possível a construção de sistemas de reconhecimento automático de voz utilizando apenas Redes Neurais Profundas. Essa abordagem é também conhecida como sistemas de ponta a ponta, "*end-to-end systems*", em inglês. Assim, é possível construir um ASR somente com redes neurais profundas. No trabalho de (GRAVES, 2012), os autores conseguem pela primeira vez construir um sistema ponta a ponta usando Classificação Temporal Conexionista ou *Connectionist Temporal Classification* (CTC). A CTC foi projetada exclusivamente para tarefas de classificação temporal, sendo aqueles problemas onde o alinhamento entre os dados de entrada e os dados de saídas são desconhecidos (QUINTANILHA, 2017).

Por fim, podemos citar os dois sistemas ponta a ponta que são o estado-da-arte atualmente, *Deep Speech 2* (HANNUN *et al.*, 2014) e *Wav2Vec2* (BAEVSKI *et al.*, 2020). *Deep Speech 2* é baseado em Redes Convolucionais e Redes Recorrentes com LSTM e *Wav2Vec2* sendo fundamentado em Redes Convolucionais para extração de *features*, *Transformer* e CTC. Ambos, não necessariamente, requerem um modelo de linguagem para seu funcionamento pleno. Detalharemos mais estes modelos na Seção 2.3.

2.2 REDES NEURAIS ARTIFICIAIS

Redes Neurais Artificiais (RNA) podem ser consideradas um conjunto de modelos matemáticos bioinspirados no funcionamento e estrutura das sinapses e neurônios do cérebro (CARVALHO, 2020).

Em 1943, o neurofisiologista Warren McCulloch e o matemático Walter Pitts (MCCULLOCH; PITTS, 1943) criaram o primeiro modelo de um "neurônio" usando circuitos elétricos. Este neurônio se baseava em um algoritmo de soma de entradas ponderadas por pesos e consequentemente passadas para uma função de ativação que por fim retornava uma saída, caso este ultrapassasse algum limiar de ativação. Já em 1958, Frank Rosenblatt criou o Perceptron (ROSENBLATT, 1957). Um algoritmo desenvolvido para o reconhecimento de padrões baseado em uma rede neural simples de uma camada, em que principal aplicabilidade é a classificação binária (KOVÁCS, 2002). Assim, o Perceptron garante

convergência quando as classes são linearmente separáveis, o que não é o caso do problema do XOR, que levou ao seu descrédito na década de 1960. A prova de convergência do procedimento de aprendizado proposto por Rosenblatt é conhecida como Teorema de Convergência do Perceptron (ROSENBLATT, 1957). Um perceptron modela um neurônio tomando uma soma ponderada de suas entradas e enviando a saída 1 (*spike*) se esta soma é maior que um determinado limiar de ativação (ROSENBLATT, 1957). A figura 4 ilustra o Perceptron:

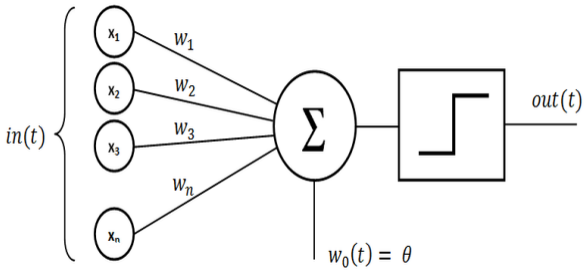

Figura 1 – Topologia de um Perceptron: As entradas x_n são ponderadas por pesos w_n, para então serem somadas e passada para uma função de ativação, que por fim daria uma saída conforme o limiar de ativação. Fonte:

(COMMONS, 2021)

Em 1969, Minsky e Papert lançaram o livro Perceptrons (MINSKY; PAPERT, 2017), com resultado negativo sobre a capacidade do Perceptron em resolver o problema do XOR (ou exclusivo). A partir desse evento, as redes neurais entraram em um "inverno" de novas publicações. Por serem desacreditas, durante 17 anos não houve muitos artigos sobre redes neurais artificiais. Em 1986, McClelland e Rumelhart lançam o livro ***Parallel Distributed Processing: Explorations in the Microstructures of Cognition*** (2 volumes) (RUMELHART; MCCLELLAND; ASANUMA, 1986) (RUMELHART; MCCLELLAND; ASANUMA, 1987), onde relançaram a ideia do "*backpropagation*", dando às redes neurais artificiais, em teoria, o poder de resolver qualquer problema, inclusive o **XOR**. A partir desse momento as redes neurais artificiais tiveram um retorno muito importante, sendo novamente objeto de grande interesse no mundo acadêmico e na indústria.

A partir dos anos 2000, um termo que já era conhecido desde a década de 1970 se tornou bastante robusto, o Aprendizado Profundo (GOODFELLOW; BENGIO; COURVILLE, 2016) (QUINTANILHA, 2017). Este veio introduzir um novo patamar para as redes neurais artificiais, que deixaram de ser apenas sistemas especialistas para de fato começarem a aprender. Podemos elencar algumas classes de redes neurais artificiais: Multi-Layer Perceptron; Redes Convolucionais; e Redes Recorrentes.

O Multi-Layer Perceptron (MLP) pode ser entendido como um conjunto de perceptrons, com a incorporação de um algoritmo de retro-propagação de erro. Pode ser referido também como uma rede de camadas densas. As unidades, ou nós, em uma MLP,

são organizadas em camadas, com conexões consequentes de uma camada para outra. Padrões de entrada são apresentados à camada de entrada e, então, estes são propagados através das "camadas escondidas" para a camada de saída (GRAVES, 2012). As MLPs são largamente utilizadas para reconhecimento de padrões, porém, seu uso não é recomendado para rotulagem de sequência (GRAVES, 2012), como reconhecimento de fala. A figura 2 ilustra a estrutura de uma MLP:

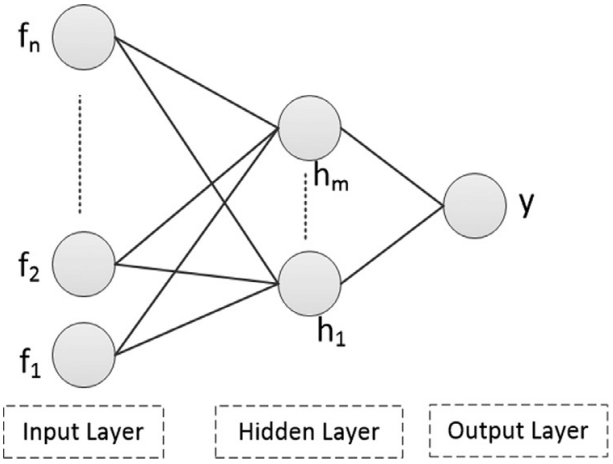

Figura 2 – Topologia de uma MLP: As entradas f_n são passadas para camadas posteriores h_n para ao final gerar uma saída y. A estrutura se assemelha ao Perceptron, porém com multi-camadas. Fonte:

(WANG YIQIN LU, 2019)

Rede Neural Convolucional (CNN ou ConvNets) é uma rede neural de aprendizado profundo. Sua concepção baseia-se em atribuir características as entradas por meio da convolução. Assim, pode-se recuperar características locais, o que não é possível com camadas densas. CNN são, provavelmente, a técnica de Aprendizado Profundo mais conhecida para resolução de problemas de classificação de imagens (PONTI *et al.*, 2017). CNNs possuem uma estrutura hierárquica de convolução, *pooling*, operadores, funções de ativação e camadas densas para classificação. Na figura 3 é possível checar a arquitetura da LeNet-5, uma rede neural convolucional desenvolvida por (LECUN *et al.*, 1998).

Rede Neural Recorrente (RNN) é uma classe de rede neural que usa dados sequenciais ou séries temporais (GRAVES, 2012). Também é considerada um algoritmo de aprendizado profundo. Esta categoria de rede neural é comumente usada para resolver problemas em que há relação de temporalidade entre os dados, tais como tradução automática, reconhecimento de voz, processamento de linguagem natural e outros (GRAVES, 2012).

2.2.1 REDES NEURAIS RECORRENTES

Redes neurais recorrentes foram criadas na década de 1980, caracterizada pelo uso de dados sequenciais ou séries temporais como entrada (GRAVES, 2012). Essas redes

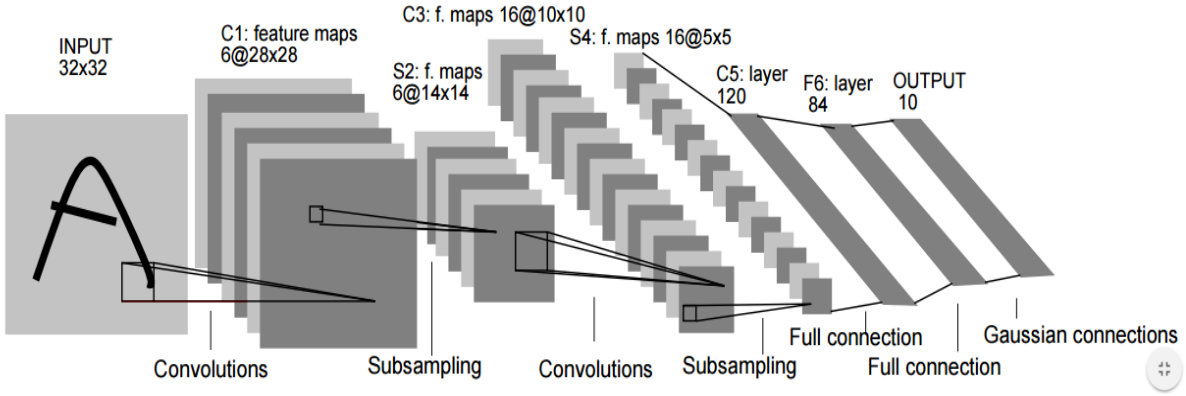

Figura 3 – Topologia de uma ConvNet(LeNet-5): Uma imagem é a entrada da rede, já escalonada em 32x32, passada para camadas de convolução para extração de características locais, C1, essas então são passadas para uma camada de *subsampling*, que gera mapa de caractrísticas mais específicas e tamanho menor, 14x14. Ao fim os mapas são enviados a uma camada densa, MLP, para classificação. Fonte:

(LECUN *et al.*, 1998)

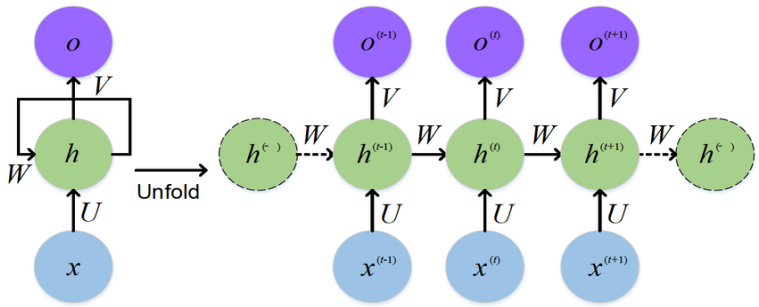

Figura 4 – Topologia de uma RNN: Uma RNN tem um laço de repetição na sua unidade oculta. A sua entrada pode ser caracterizada como um conjunto de 3 categorias de camadas: a camada de entrada x, a oculta h e a de saída o. Assim, podemos observar que essa estrutura é replicada temporalmente, sendo o resultado da camada h como entrada para a próxima camada. U, V e W são as matrizes de pesos das camadas ocultas, Fonte: (FENG *et al.*, 2017)

usam laços interativos entre os nós para guardar informação e as características geradas se tornam uma escolha atrativa para sequências rotuladas, como NLP (*Natural Language Processing*), reconhecimento de voz e escrita a mão (GRAVES, 2012). Uma importante característica das Redes Recorrentes é que elas podem aprender correlações entre as instâncias, diferentemente de uma Rede Neural densa (MLP) (GRAVES, 2012). Outra característica que difere as RNNs da MLP é que as RNNs formam ciclos entre os nós, o que são chamados "*feedback neural networks*", enquanto as MLPs não formam ciclos, por isso, são chamadas "*feed forward neural networks*".

As redes neurais recorrentes possuem o problema do desaparecimento do gradiente. Então, em 1997, os autores (HOCHREITER; SCHMIDHUBER, 1997) introduziram uma nova classe de RNN chamada ***Long Short-Term Memory*** (LSTM), endereçando a solução desse problema, ou seja, onde o estado atual não pode ser predito por um estado passado distante. (IBM, 2020a). A arquitetura de uma LSTM, basicamente, consiste em um conjunto de sub-redes conectadas recorrentemente, isso é conhecido como "blocos de memória". Cada bloco possui um ou mais células de memória auto-conectadas e três unidades multiplicadoras — a entrada, a saída e os portões de esquecimento (IBM, 2020b).

Outra variação de uma RNN é conhecida como ***GRU (Gated Recurrent Unit)***, proposta em 2014 por (CHO *et al.*, 2014). Assim como a LSTM, também possui uma estrutura de portões, porém sem o *cell state*, sendo adicionadas de um portão de "esquecimento" acoplado ao portão de entrada. Assim, ao invés de tomar decisões de forma separada, como esquecer ou adicionar uma informação, deve-se tomar essa decisão de forma conjunta (QUINTANILHA, 2017). GRUs são consideradas mais simples e mais rápidas de serem treinadas do que uma LSTM convencional.

2.2.2 TRANSFORMERS

Transformers é uma rede neural baseada no mecanismo de atenção. O mecanismo de atenção permite que o modelo olhe diretamente e com base no estado de qualquer ponto anterior da sequência. A camada de atenção tem acesso a todos os estados anteriores e os pondera de acordo com alguma medida de relevância aprendida para o *token* atual, fornecendo informações mais claras sobre *tokens* relacionados distantes. Inicialmente, *Transformers* foi publicada no artigo "*Attention is All You Need*" (VASWANI *et al.*, 2017). Neste estudo foi demonstrado que esta rede neural conseguiu superar o estado-da-arte em tradução de inglês para alemão e inglês para francês. Este marco foi atingido com um tempo de treinamento bem menor em relação a outros modelos de redes neurais profundas, principalmente, aqueles que utilizavam RNN. Isso foi alcançado por paralelização com 8 GPUs.

Um problema comum no modelo anterior, baseado em atenção e redes neurais recorrentes, é sua intrínseca natureza sequencial, onde cada entrada e saída gera um estado escondido ("*hidden state*") para cada entrada e saída anterior (VASWANI *et al.*, 2017). Assim, o treinamento pode durar bastante tempo e não usar eficientemente as GPUs e TPUs, perdendo-se assim o grande poder de paralelização que esses modelos entregam e aumentando consideravelmente o tempo de treinamento.

A arquitetura de uma rede neural do tipo Transformer pode ser dividida da seguinte maneira (VASWANI *et al.*, 2017), conforme apresentando na Figura 5:

- **Input Embedding e Positional Encoding:** A entrada da rede Transformers

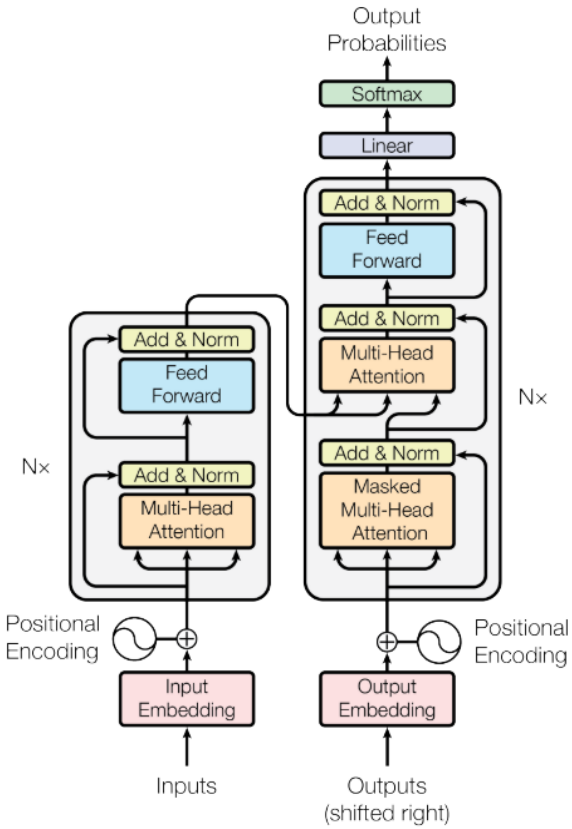

Figura 5 – Arquitetura do modelo Transformers. A figura está separada em duas grandes camadas, a esquerda o Encoder, e a direita o Decoder. O encoder se preocupa em processar a sequência de entrada em uma representação contínua, e o Decoder processa essa representação para uma única saída. Fonte: (VASWANI *et al.*, 2017)

precisa ser vetorizada, ou seja, transformar os dados, que podem ser de diversos formatos, em um vetor de números. Aliado a isso, é necessário passar a posição de cada item no vetor, isso se dá pela entrada ser totalmente paralelizada, ou seja, todos os dados são processados de uma única vez. Assim é necessário saber qual a ordem original em que os dados foram passados. No citado artigo é usado seno e coseno para representar as posições das sequências.

- **Multi-Head Attention:** Este módulo computa o mecanismo de atenção diversas vezes em paralelo. Cada uma dessas repetições é chamada de "*Attention Head*". O módulo de atenção separa seus parâmetros em *Queries*, Chaves e Valores de várias formas e passa cada um para uma "Cabeça (Head)" separada. Após todos os cálculos, tudo é agrupado para produzir uma pontuação de atenção. Isso é chamado "Multi-Head Attention" e dá à rede Transformer um grande poder de paralelização e de reconhecimento de várias "nuances" nos dados de entrada.

- **Feed Forward Neural Network:** Os vetores de atenção devem ser passados para uma rede neural densa para serem convertidos em um único vetor para ser consumido pela próxima camada de *encoder* ou *decoder*. Cada vetor é totalmente independente do outro, o que permite paralelização.

- **Masked Multi-Head Attention:** Similar ao módulo "*Multi-Head Attention*", este aplica um vetor de atenção a cada sequência, porém mascara as sequências que aparecem mais tarde, fazendo assim que a rede não precise processá-las posteriormente.

2.3 APLICAÇÃO DAS REDES NEURAIS NO RECONHECIMENTO DE VOZ

Nesta seção são apresentados os trabalhos relacionados com o tema proposto, no que se refere aos processos de construção de uma solução de reconhecimento de voz no contexto de Redes Neurais Profundas usando um conjunto de dados em português do Brasil. Assim, elenco trabalhos que abordam reconhecimento de fala usando Redes Neurais Recorrentes com LSTM, como a solução encontrada no Deep Speech, e soluções que usam Transformers, com o Wav2Vec2.

No trabalho (GRIS *et al.*, 2021) é demonstrado o uso do Wav2Vec2 para a construção de uma solução de reconhecimento de voz estado-da-arte, conseguindo 11.95% de WER sem o uso de modelo de linguagem. O Wav2Vec2 foi lançado por um time de pesquisadores do Facebook. Este modelo tem sua arquitetura baseada em Transformers, trabalhando ponta-a-ponta, ou seja, somente redes neurais para realizar o processo. O modelo foi inspirado nos trabalhos antecessores do Wav2Vec (SCHNEIDER *et al.*, 2019) e Vq-Wav2vec (BAEVSKI *et al.*, 2020). Este modelo foi treinado usando uma *corpus* multi-língua, o que, segundo os autores, teve uma performance superior a um modelo treinado apenas com uma *corpora* em português. O modelo é muito robusto, porém, sua inferência em CPU é bem inferior a GPU. Outro ponto negativo é que mesmo tendo uma WER relativamente baixa, algumas palavras vêm com alguns caracteres errados, o que pode ser melhorado com um modelo de linguagem para um contexto específico.

Em (KANDA *et al.*, 2021) é apresentada a construção de uma rede neural ponta-a-ponta baseada em Transformers para reconhecimento de voz no idioma inglês. Neste estudo, os autores exemplificam como adaptaram uma rede LSTM para poder ser usada como Transformers. Este trabalho é importante devido à demonstração de como adaptar uma rede neural LSTM para **Transformers**. Demonstrando a eficiência dessa arquitetura em melhorar o resultado do reconhecimento de voz.

Em (HANNUN *et al.*, 2014) é demonstrado pela primeira vez o Deep Speech, onde os autores apresentam a ferramenta, informam um resultado de Word Error Rate (WER) de 16% no idioma inglês usando apenas rede neurais recorrentes bem otimizadas e treinadas em GPU. No trabalho de (QUINTANILHA, 2017), os autores usam uma rede baseada

no Deep Speech para construir uma solução de reconhecimento de voz. Os autores fazem diversos experimentos na construção da rede, ajustando os hiper-parâmetros, variando o número de camadas e aplicando diversos métodos de regularização. Nos seus testes, conseguiram uma taxa de erro de caractere de 25,13%, que segundo o autor é 11% maior que em sistemas comerciais mensurados a época.

No trabalho de (RODRIGUES; PINHEIRO, 2020), os autores constroem uma rede neural profunda simplificando a arquitetura do *Deep Speech*, para ser possível um treinamento mais rápido. Neste trabalho utilizam a base de áudios aberta e validada da Mozilla, *Common Voice* que estava com tamanho de 750 MB. A última versão desta base de dados, "2021-07", possui pouco mais de 3 GB. Os resultados foram satisfatórios para a rede reconhecer palavras simples, como "não" e "foi", porém, não atingiu bons resultados nos vários testes propostos no trabalho, tendo uma taxa de WER superior a 100% em vários cenários. Este trabalho é importante para identificar possíveis pontos de melhoria na criação das redes neurais para ASR que este projeto usará.

Com o Aprendizado Profundo foi verificado um salto expressivo na construção de sistemas de reconhecimento de fala robustos, diminuindo os componentes necessários para se produzir e treinar. Nesse contexto, podemos destacar os modelos baseados em redes neurais profundas, como o *DeepSpeech* e o *Wav2Vec2*, que trazem modelos ponta-a-ponta, mais fáceis de treinar e com resultados estado-da-arte quando comparados com os sistemas de reconhecimento de fala tradicionais, como os baseados em Cadeias Ocultas de Markov com modelos de linguagens. Assim, este trabalho apresenta uma implementação de ambos os modelos citados, fazendo uma comparação de desempenho no treinamento e inferência após os modelos treinados. Os modelos usarão *checkpoints* de outros modelos já treinados, fazendo assim um *fine-tunning* nesses, para garantir um melhor desempenho ao final do teste. O modelo de avaliação de ambos será o mesmo, o CommonVoice da Mozila na versão "2021-07", e a inferência após treinados usará uma base de áudios limitada a um contexto de assistente pessoal voltado para vendas.

No próximo capítulo, será aprofundado as técnicas desenvolvidas na implementação dos modelos Wav2Vec2 e DeepSpeech2/CoquiSTT. Descrição das Redes Neurais utilizadas, bem como os córpus usados nos treinamentos.

3 DESENVOLVIMENTO

3.1 CONSIDERAÇÕES INICIAIS

Este capítulo visa apresentar o processo de implementação dos dois modelos citados, através do treinamento de ambas as redes neurais artificiais. Esta análise comparará as duas redes quanto ao desempenho na tarefa proposta, reconhecimento automático de fala. Assim como comparar a taxa de erro (WER e CER) e a *loss* (CTC loss) no conjunto de validação e teste do córpus Common Voice e explicar as vantagens e desvantagens na utilização de ambas.

Também foi levantado uma base de áudios própria no domínio da empresa da qual este autor presta serviço. Essa base refere-se a um contexto de atendimento automático através de ligações telefônicas feitas para uma central de clínica de saúde e através da captação de áudio de voluntários em um aplicativo de mensagens. Com isso, pretende-se usá-la para validação das duas redes, bem como aplicação de ruídos para verificar se o desempenho da rede continua satisfatório.

3.2 BASE DE DADOS - COMMON VOICE 8.0

Para este trabalho foi utilizado o córpus de áudios Common Voice (CV), na versão 8, no escopo de treinamento, validação e teste. O córpus Common Voice foi desenvolvido pela empresa Mozilla, visando avançar o campo do reconhecimento automático de fala para idiomas com poucos recursos, ou seja, que não possuam um córpus satisfatório para treinamento de modelos para esta tarefa. A empresa disponibiliza uma plataforma aberta para que pessoas do mundo todo possam contribuir. Estas contribuições podem ser feitas enviando áudios de transcrições informadas pela plataforma, ou validando a transcrição de áudios enviados por outros usuários (ARDILA *et al.*, 2020).

A última versão do CV possui mais de 13 mil horas de horas gravadas e mais de 11 mil horas validadas em 76 idiomas. Este trabalho usa o córpus em português, com 130 horas gravadas e 112 horas validadas.

O histograma desta base quanto a estratificação por gênero e idade podem ser conferidas nas Figuras 6 e 7, respectivamente. Pode-se observar um desbalanceamento em relação ao sexo masculino, representando 76% da base, enquanto o sexo feminino é representado por apenas 4%.

O CV já dispõe de uma divisão de bases de treinamento, validação e teste, representada pelos arquivos "train.tsv", "dev.tsv"e "test.tsv". Para os experimentos, será considerada essa divisão.

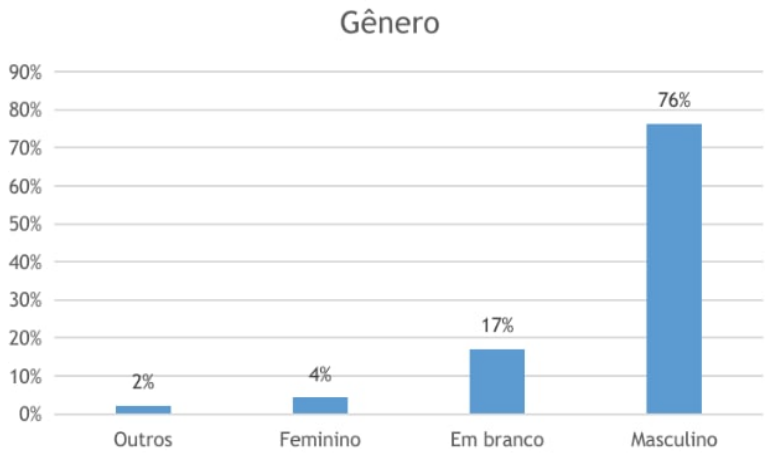

Figura 6 – Base dividida por gênero. A seleção de gênero não é obrigatória, deixando alguns registros em branco.

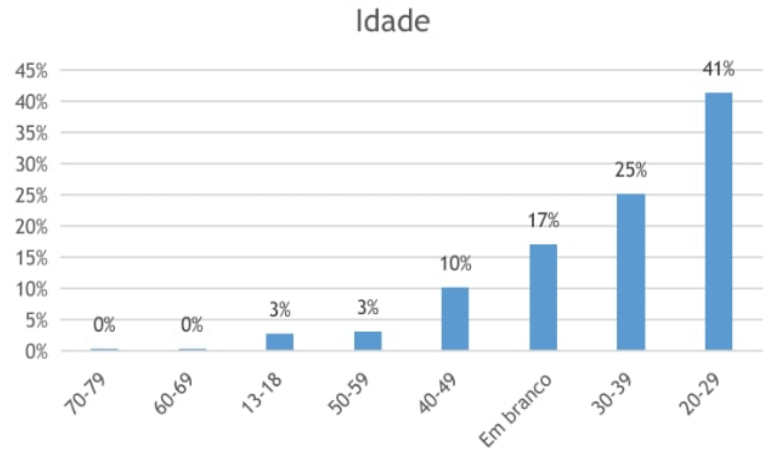

Figura 7 – Base dividida por idade. A seleção de idade não é obrigatória, deixando alguns registros em branco.

3.2.1 PRÉ-PROCESSAMENTO

Devido a uma particularidade nas versões 7 e 8 do CV, alguns áudios não estão em 48Khz, como definido na documentação e, assim, precisam ser redimensionados para a frequência esperada. O fato de ter áudios em frequências diferentes não é necessariamente um problema para treinamento em redes neurais, já que esse pode ser considerado um comportamento de *data-augmentation*. No entanto, por questão de padronização, foi necessário ser executado um script em toda a base para checar a frequência e, caso necessário, fazer a mudança para 48Khz.

Outro ponto de limpeza realizado foi fazer algumas mudanças nas transcrições para remover caracteres que não façam diferença nos fonemas. Como, por exemplo, o trema, crase, pontos de exclamação e interrogação, vírgulas, pontos, etc.

3.3 AMBIENTE DE TREINAMENTO

O ambiente de treinamento trata-se de uma instância no Google Cloud Platform (GCP). O servidor em questão possui as configurações informadas na Tabela 1.

Tabela 1 – Especificações do ambiente de treinamento

Especificações	
CPU	8 vCPU Intel 2.8 Ghz
RAM	16 GB
GPU	NVIDIA Tesla T4 16 GB
SO	Ubuntu 18.04.2 LTS

3.4 EXPERIMENTO 1: COQUI STT

Como já mencionado no capítulo 2, utilizaremos o CoquiSTT para a validação da arquitetura, pois este é um bifurcação do DeepSpeech2, e continua tendo atualizações de erros e adição de melhorias, o que não foi constatado na base de código do DeepSpeech2. A sua arquitetura é basicamente a mesma do DeepSpeech2. A arquitetura do DeepSpeech2/CoquiSTT pode ser verificada na figura 8.

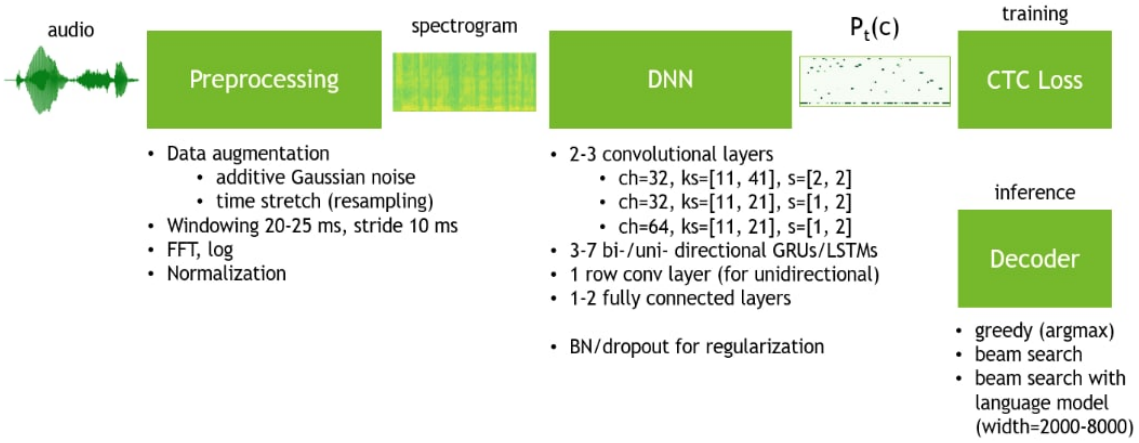

Figura 8 – Arquitetura padrão do DeepSpeech2/CoquiSTT. O áudio de entrada é passado por uma camada de pré-processamento, onde são aplicados algumas técnicas de aumentação de dados. A próxima camada é gerado seu espectrograma para extração de características através de 2-3 camadas de CNN. 3-7 bi-uni direcionais camadas de LSTM/GRU. Mais uma camada de CNN. 1-2 camadas densas. É aplicada a CTC Loss ao final.

[1]

Para o treinamento com o CoquiSTT foi utilizado uma imagem *docker* da NVIDIA com a versão do driver CUDA compatível, pois este é suportado apenas até o CUDA 10 e CuDNN 7.6, devido à utilização do Tensorflow na versão 1.15.4. Nesta imagem *docker*, o Python instalado está na versão 3.7. Os comandos necessários para treinamento, está

em um *shell script* no *github*.[2]. Os parâmetros de treinamento podem ser verificados na Tabela 2.

Tabela 2 – Parâmetros de treinamento CoquiSTT

Treinamento	
Epochs	500
steps	100
optmizer	Stochastic Gradient Descent with Momentum optimizer
learning_rate	0.001
train_batch_size	100
dev_batch_size	100
train_cudnn	true
automatic_mixed_precision	true

É necessário realizar o *download* do CV para o servidor, descompactar e pôr em uma pasta visível a imagem *docker*. Para o treinamento com 500 épocas, foi necessário cerca de 5 horas para completar, e mais 45 minutos para validação no conjunto de testes.

3.5 EXPERIMENTO 2: WAV2VEC2

Para este experimento foi efetuado um *fine-tuning* do modelo "facebook/wav2vec2-xls-r-300m"[3] para a tarefa de reconhecimento de fala. Este modelo já foi pré-treinado com 436h de áudios não rotulados a partir de diversas bases de dados, como VoxPopuli, MLS, CommonVoice, BABEL, e VoxLingua107. Estas bases possuem áudios em 128 idiomas diferentes. Este modelo possui 300 milhões de parâmetros. O fluxograma de trabalho do Wav2vec pode ser verificado na Figura 10.

Não foi necessário utilizar a imagem docker da NVIDIA aqui, pois pode-se usar as versões mais recentes dos *softwares*. A versão do Python utilizado foi a 3.9. Os *drivers* da NVIDIA devem ser instalados manualmente, usado CUDA 11 com *drivers* na versão 546. Para utilização do Common Voice não foi necessário realizar o *download* por meio do site, como efetuado no experimento 1. Para este caso, foi utilizado o pacote "transformers.dataset"para realizar o seu uso.

Os *scripts* de treinamento e avaliação foram desenvolvidos a partir da bifurcação do projeto do (GROSMAN, 2022). Neste projeto está estruturado uma forma eficaz de fazer o treinamento e gerar os resultados de avaliação. Sendo implementado a métrica CER e WER em lote, necessário para o meu caso de uso. Este projeto pode ser acessado a partir do link do GitHub.[4].

[2] https://github.com/tonyalves/coquistt-trainer
[3] https://huggingface.co/facebook/wav2vec2-xls-r-300m
[4] https://github.com/tonyalves/huggingsound

Tabela 3 – Parâmetros de treinamento Wav2Vec2

Treinamento	
epochs	50
total_train_batch_size	32
optmizer	Adam with betas=(0.9,0.999) and epsilon=1e-08
learning_rate	7.5e-05
train_batch_size	8
eval_batch_size	8
gradient_accumulation_steps	4
lr_scheduler_type	linear
lr_scheduler_warmup_steps	2000
mixed_precision_training	Native AMP

O tempo de treinamento superou 48 horas para executar 50 épocas, e mais 5 horas para executar a validação no conjunto de testes. Os parâmetros de treinamento para essa rede, podem ser conferidas na Tabela 3.

3.6 CÓRPUS PRIVADO

Um base de áudios específica de negócio foi construída no decorrer deste trabalho. Essa base consiste em áudios referentes a uma central de atendimento especializada em atendimento de clientes de uma clínica médica e cobrança automática de faturas em atraso.

Esta base foi construída de duas formas: por um sistema automatizado integrado ao software Telegram, um "bot", onde os usuários eram apresentados a uma frase, e então estes enviavam os áudios correspondentes ao texto visualizado. Este sistema permitia ao usuário validar se o áudio estava condizente e, caso contrário, poderia enviar novamente o áudio ou cancelar a operação. Esta aplicação foi inspirada no modelo de funcionamento de captura de áudios do Common Voice. Após o levantamento destes áudios, uma segunda etapa de validação era verificada por terceiros se os áudios realmente correspondiam com os textos passados.

Os usuários concordaram em ceder os áudios para esta pesquisa e os tornassem públicos, porém não foi obtido uma quantidade significativa de áudios. Ao todo foram 100 áudios validados, totalizando 10 minutos de áudio validado.

Outra fonte de dados foi desenvolvida a partir de análise de ligações telefônicas, de modo a capturar as respostas dos clientes ao sistema de cobrança eletrônica. Estes dados foram capturados para anonimizar os interlocutores e os dados sensíveis que possam ser informados por telefone. Após a coleta dessas gravações foi desenvolvido um sistema para identificar apenas a parte onde o cliente fala. Para isso as gravações tiveram que ser capturadas no modo estéreo, e o sistema desenvolvido foi responsável por processar apenas o canal de voz referente ao cliente, remover os silêncios e dividir em sub-áudios.

Em um segundo momento, os sub-áudios extraídos foram processados por um sistema de reconhecimento de fala automático comercial para serem geradas as suas transcrições.

Após gerada as transcrições foi efetuado um trabalho manual de validação, de modo a checar se as transcrições estavam condizentes e efetuar correções nos textos, caso necessário. Como esta é uma etapa bastante demorada, o tamanho do córpus gerado foi reduzido para 200 áudios. Após a validação foi obtido um total de 100 áudios validados.

3.6.1 APLICAÇÃO DE RUÍDO

Foi aplicado ruídos nesta base para que se possa validar a inferência no melhor modelo. O ruído aplicado foi baseado no Projeto Voices[5]. Este projeto montou um conjunto de áudios de vários sons ambientes, de várias fontes de entrada diferentes, para serem usados para aumentação de dados em tarefas de reconhecimento de fala.

Assim, foi escolhido alguns áudios desse conjunto de dados, e aplicado de forma aleatórios nesta base. O método utilizado foi através do PyTorch[6]. Foi aplicado o ruído somando os tensores do áudio da base com o áudio do ruído, seguindo a técnica *Signal-to-Noise Ratio (SNR)*[7]. De modo até um áudio inteligível, foi feito um trabalho de, após aplicar o ruído, ouvi-lo e checar se um humano conseguiria entender. Ao conseguir o entendimento, então os parâmetros de ruídos eram aplicados ao restante dos áudios para treinamento na rede neural.

O processo para aplicação de ruído na base pode ser conferido neste link: https://colab.research.google.com/drive/1Bq9t2AggkjrLlitcvPU7LJYVY00nWMxI?usp=sharing

[5] https://iqtlabs.github.io/voices/Lab41-SRI-VOiCES$_{README}$

[6] https://pytorch.org/audio/main/tutorials/audio$_{data}$$_{augmentation}$$_{tutorial.html}$

[7] https://en.wikipedia.org/wiki/Signal-to-noise$_{ratio}$

4 RESULTADOS

Neste capítulo são apresentados os resultados de ambas as redes implementadas para o conjunto Common Voice, assim como o córpus gerado e comparações das performances entre os modelos. Os resultados são compostos por taxas de predições, WER e CER, dos melhores e piores resultados, assim como resultados médios. Também é disponibilizado gráficos da função de perda pela quantidade de épocas no conjunto de validação durante a etapa de treinamento.

4.1 COQUI STT

Após vários experimentos com configurações diferentes, a melhor configuração de parâmetros conseguiu os seguintes resultados no conjunto de testes do Common Voice WER: 0,96 (96%), CER: 0,50 (50%) e loss: 70,96. No entanto, os outros experimentos tiveram resultados bem parecidos a partir de 100 épocas.

Um resumo das predições pode ser observado na Tabela 4. O gráfico da *loss* pode ser conferido na figura 9.

Com esses resultados observa-se que o CoquiSTT não atingiu um desempenho satisfatório nesta tarefa. Isso pode ser creditado ao fato do conjunto de treinamento ser relativamente pequeno. Há modelos treinados para o idioma inglês em que o WER fica abaixo dos 4%, porém o conjunto de treinamento possui 1700h (COQUI,), algo bem distante do conjunto de treinamento usado neste experimento, que possui menos de 100h de áudios.

Tabela 4 – Resumo dos resultados CoquiSTT

Melhor WER			
Frase	Predição	WER	CER
malacacheta	malacacheta	0,00	0,00
caldeirão grande	caldeirão grande	0,00	0,00
bacabal	bacabal	0,00	0,00
WER Médio			
Frase	Predição	WER	CER
estrelas no calor do dia guerra fome e doença	strelas pocavartantia era formnnhtons	1,00	0,48
eu fugi sem dizer nada sobre o que aconteceu	e vogisemndizerravras sou quiconi sê	1,00	0,52
uma pessoa andando de caiaque em uma pequena cachoeira	mapesindangicagonqunma piquimecachuina	1,00	0,55
o sujeito escorregou para a multidão e se perdeu	osorjeti coregoparamuridãtidarde	1,00	0,52
Pior WER			
Frase	Predição	WER	CER
caçapava	caj amarn	2,00	0,28
firefox	tiar farratrariter micatra	3,00	3,28
queimados	que maco s	3,00	0,33
ananindeua	araem de a	3,00	0,50

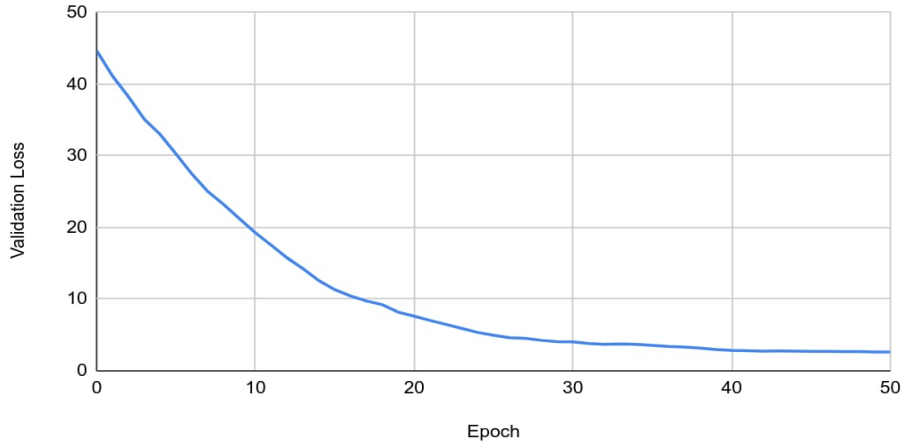

Figura 9 – No eixo y, a taxa de perda no conjunto de validação, usando a CTC Loss. No eixo x, a época. Treinada por 50 épocas.

4.2 WAV2VEC2

Após cerca de 48h de treinamento, com os hiper-parâmetros ajustados de acordo como mencionado anteriormente, os resultados seguiram da seguinte forma: WER: 0.13 (13,5%), CER: 0.038 (3,8%) e Loss 0,15. Os resultados podem ser verificados em https://huggingface.co/tonyalves/output. Devido ao resultado bastante superior em relação ao primeiro modelo, não foi testado novas configurações de hiper-parâmetros. A tabela 5 mostra alguns exemplos de inferência no conjunto de teste do Common Voice.

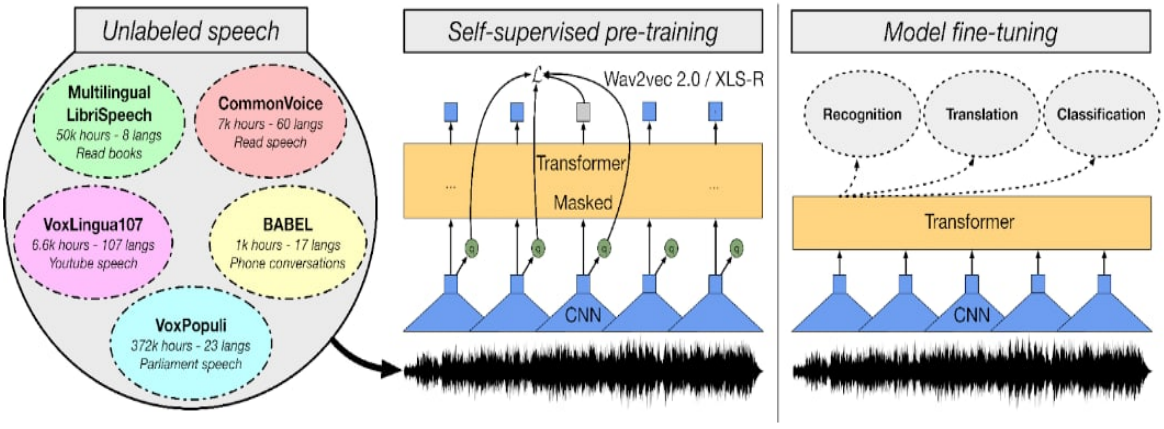

Figura 10 – Fluxograma do desenvolvimento do Wav2Vec2. Foi juntado várias bases de áudios não rotulados, aplicados no modelo Wav2vec2 e por fim, descrição das possíveis tarefas que podem ser realizadas a partir deste modelo, como: Reconhecimento de fala, tradução automática e classificação de áudios

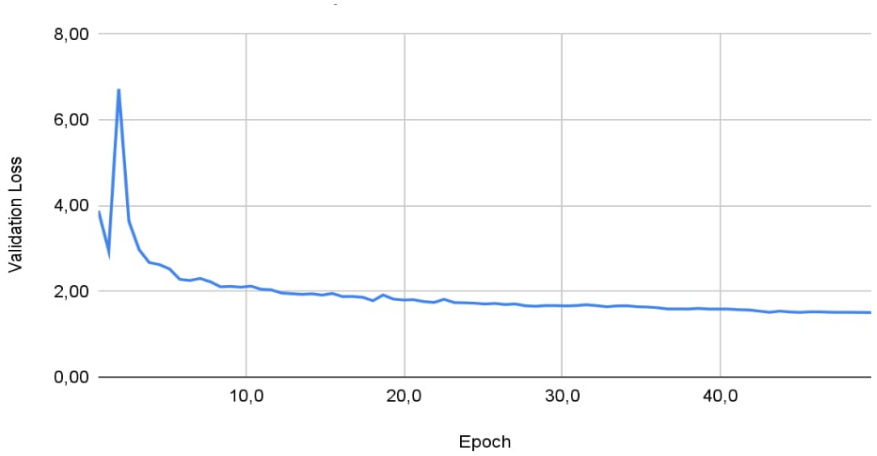

Figura 11 – No eixo y, a taxa de perda no conjunto de validação, usando a CTC Loss. No eixo x, a época. Treinada por 500 épocas.

Tabela 5 – Resumo dos resultados usando Wav2Vec2

Melhor WER			
Frase	Predição	WER	CER
estamos aqui para pedir emprestado	estamos aqui para pedir emprestado	0,00	0,00
precisamos nos apressar	precisamos nos apressar	0,00	0,00
este trabalho já começou	este trabalho já começou	0,00	0,00
ter um plano de longo prazo	ter um plano de longo prazo	0,00	0,00
nós não temos relação de sangue	nós não temos relação de sangue	0,00	0,00
tudo estava sincronizado	tudo estava sincronizado	0,00	0,00
projetor de tela	projetor de tela	0,00	0,00
a alça da taça está quebrada	a alça da taça está quebrada	0,00	0,00
WER Médio			
Frase	Predição	WER	CER
para começar ambos sabemos que eu tenho	para começar amos sabemos que eu tenho	0,14	0,03
um menino louro novo joga na areia	um menino louro nolo joga na areia	0,14	0,03
e nós temos que conversar o que eu acho que é tempo de paixão	e nós temos que conversar que eu acho que hé tempo de paixão	0,14	0,05
o casco do navio entrou em colapso	o casco do navio entrou em colaps	0,14	0,03
Pior WER			
Frase	Predição	WER	CER
ananindeua	ananim deuá	2,00	0,30
itapecurumirim	itapecuru mirim	2,00	0,07
anhanguera	lenhao guervo	2,00	0,60
witmarsum	wite marsum	2,00	0,22
baleiajubarte	baleias de ubarte	3,00	0,38
nãometoque	não me toque	3,00	0,20
beneditinos	bené de tinos	3,00	0,36

4.3 CÓRPUS PRIVADO

O resultado obtido depois desta transformação nos áudios pode ser examinada a seguir. A Tabela 6 mostra o resultado da primeira validação sem aplicação de ruídos, enquanto a Tabela 7 mostra os resultados após a aplicação.

O WER deste experimento sem ruídos ficou em 23,69% e o CER em 5,08%. No experimento aplicando ruído o resultado de WER e CER foram: 117% e 25,79%, respectivamente. Pode-se observar que a aplicação de ruídos elevou muito as taxas de erros, indicando que o modelo proposto não se adequa bem em ambientes muito ruidosos. O que pode ser corrigido com mais dados de treinamento com os tipos de ruídos aplicado e/ou aplicação desse tipo de ruídos na aumentação dos dados no pré-processamento.

Tabela 6 – Resumo dos resultados da base de dados própria, sem ruídos, no Wav2Vec2

Melhor WER			
Frase	Predição	WER	CER
senhor não sei quem é essa moça	senhor não sei quem é essa moça	0,00	0,00
sim eu posso fazer isso senhora	sim eu posso fazer isso senhora	0,00	0,00
já fechei esse pagamento	já fechei esse pagamento	0,00	0,00
nunca ouvi falar	nunca ouvi falar	0,00	0,00
especialista em angiologia	especialista em angiologia	0,00	0,00
por favor me dá um cartão da clínica	por favor me dá um cartão da clínica	0,00	0,00
gostaria de saber qual o preço da consulta médica	gostaria de saber qual o preço da consulta médica	0,00	0,00
WER Médio			
Frase	Predição	WER	CER
já vi que esse exame não pode ser demorado	já vi que esse examenão pode ser demorado	0,22	0,10
especialista em cirurgia pediátrica	especialista em sirurgia pediátrica	0,25	0,10
especialista em medicina intensiva	especialista e medicina intensiva	0,25	0,11
especialista em cirurgia torácica	especialista em cirurgia torássica	0,25	0,12
nefrologia me lembra açúcar	nefrologia me lembra açuca	0,25	0,13
o fonoaudiólogo é simpático com os ouvidos	o fon audiólogo é simpático com os ouvidos	0,29	0,02
olha lá quem vem é o gastroenterologista	olha lá quem vem é o gastro enterologista	0,29	0,03
Pior WER			
Frase	Predição	WER	CER
a ordontia é um ortodontista enjaulado	a ordontia é um otor don tista enjaaulado	0,67	0,13
olha lá quem vem é o gastroenterologista	olhe lá quem venha o gasto imperalojista	0,71	0,28
especialista em clínica médica	sfez à lista em clínica médica	0,75	0,20
nunca vi um periodontista	nunca hvia um périoo dantista	0,75	0,24
uma prótese precisa de remendos	uma protas e preces a de remendos	0,80	0,16
nem sou eu quem tá devendo aquilo	nem soei o que tadever naquilo	0,86	0,33
especialista em geriatria	pecialista engery atria	1,00	0,24

Tabela 7 – Resumo dos resultados da base de dados própria, com ruídos, no Wav2Vec2

Melhor WER			
Frase	Predição	WER	CER
especialista em pneumologia	especialista em pneumologia	0,00	0,00
especialista em cardiologia	especialista em cardiologia	0,00	0,00
sim sou eu pode falar	sim sou eu pode falar	0,00	0,00
também não entendo o que é errado nisso	também não entendo que é errado nisso	0,13	0,05
não reconheço nada do que é isso	não raiconheco nada do que é isso	0,14	0,09
não sei quem é que está falando	não sei quem é quem está falando	0,14	0,03
sim eu posso fazer isso senhora	sim eu posso fazer isso senhor	0,17	0,03
WER Médio			
Frase	Predição	WER	CER
nunca vi um periodontista	nunca vi um terde olgundiske	0,50	0,40
por favor me dá um cartão da clínica	o popvo me dar um cartão da fina	0,50	0,36
especialista em medicina intensiva	especialista é umadesin intensiva	0,50	0,21
especialista em cirurgia torácica	especialista em sirugia tarática	0,50	0,12
residência médica e título de especialista	residência médica enti ide especialista	0,50	0,17
talvez o acaso vá me proteger	talvez o acaso vamo protegir	0,50	0,14
residência médica e título de especialista	residência mégica e titile de espacialista	0,50	0,12
gostaria de saber qual o preço da consulta médica	gostaria de saber qual os prês so da feso cement	0,56	0,37
Pior WER			
Frase	Predição	WER	CER
especialista em clínica médica	especalistenteinicomédi	1,00	0,40
especialista em ortopedia e traumatologia	etatialistetatatiatal matologia	1,00	0,49
certo já vi esse filme anteriormente	saco já desco se eu me atariamente	1,00	0,47
especialista em genética médica	pessoalista engenéti camédi	1,00	0,35
especialista em alergia e imunologia	es pasados em ala sria e u manaerofi	1,20	0,56
especialista em neurocirurgia	espéciarista e neuro siridia	1,33	0,28
especialista em otorrinolaringologia	especialistas não torni no lar em bologia	2,00	0,36

4.4 VANTAGENS E DESVANTAGENS - COMPARAÇÃO DE DESEMPENHO

Como observado pelo resultado dos experimentos, o modelo Wav2vec2 se mostrou bastante superior ao modelo baseado no DeepSpeech2, CoquiSTT. No decorrer do desenvolvimento deste trabalho, analisando trabalhos publicados sobre esse tema, pode ser verificado que a quantidade de dados é mais importante para redes conforme o DeepSpeech2. Pois, essa rede usa apenas camadas convolucionais para extração de características e LSTM e GRU para processar as sequências de entrada. Não se equipara com a rede do Wav2vec2 devido esta ser pré-treinada com centenas de horas de áudios não rotulados. A quantidade de dados com suas características extraídas previamente, melhoram expressiva quando é feito o *fine-tuning* para a tarefa de reconhecimento automático de fala.

Mesmo com a visível superioridade do modelo baseado em Transformers, Wav2vec2, tem que ser considerado o tempo de treinamento que supera bastante ao tempo do modelo CoquiSTT. Usando o córpus Common Voice, de cerca de 100h de áudios, o modelo Wav2vec2 levou cerca de 48h para terminar todo o treinamento com 50 épocas, e quase 4 horas para realizar a validação no conjunto de teste. Enquanto o CoquiSTT levou apenas 4 horas para rodar 500 épocas, e mais 40 minutos para rodar a validação no conjunto de testes.

Outro ponto a ser considerado ao usar o modelo Wav2Vec2 em produção, é a sua alta latência em CPU. Esta limitação fica ainda mais evidente quando são realizadas várias requisições simultâneas ao serviço. Nos testes executados no servidor mencionado anteriormente, sem a adição da placa de vídeo, ao lidar com 10 requisições simultâneas, foi observado em média uma latência de 10 segundos para cada 3 segundos de áudio enviado. Com o uso da placa de vídeo com CUDA, essa latência foi diminuída para 100 milissegundos. Isso deve-se ao fato da rede Wav2Vec2 ter uma quantidade de parâmetros relativamente elevada, sendo a rede testada com 300 milhões de parâmetros.

Esta situação não foi observada de maneira tão expressiva quando comparada ao modelo CoquiSTT. Por esta ter uma estrutura relativamente reduzida, tanto o seu processo de treinamento como de inferência são bem rápidos em CPU, mas que conseguem um desempenho bem melhor quando executados em GPU. O CoquiSTT/DeepSpeech2 usam a versão 1.XX do Tensorflow, obrigando a usar a GPU com drivers desatualizados e CUDA 10. Não se pode usar versões mais recentes devido à incompatibilidade com o Tensorflow nesta versão.

O tamanho dos modelos difere muito. O modelo gerado a partir do CoquiSTT gira em torno de 100 MB, enquanto o modelo gerado do Wav2Vec2 possui em torno de 1.8 GB. O que neste caso, inviabiliza um início rápido do modelo para inferência e aproveitar das tecnologia “serverless” dos provedores de nuvem.

5 CONCLUSÃO

Wav2Vec2 se mostrou bastante robusto quanto ao desempenho apresentado, mesmo com uma base de áudios relativamente pequena, em relação a mesma em inglês, atingindo resultado comercialmente viável. Todo esse desempenho pode ser creditado ao processo de pré-treinamento que o modelo passou ao ser treinado com centenas de horas de áudios não rotulados, juntando isso ao modelo de Transformers, que possui uma arquitetura bem robusta para processamento de dados de sequência para sequência. O modelo Wav2Vec2 ainda pode ser treinado para tarefas de classificação e tradução automática. Mesmo com um resultado bastante promissor, o modelo treinado em português ainda está com desempenho abaixo de um mesmo treinado em inglês, onde há modelos na plataforma HuggingFaces com WER de 3.4%[1]. No entanto, o modelo em questão foi treinado com 960 horas de áudios rotulados. No Brasil há projetos importantes para a construção de córpus robustos, como o projeto Tarsila da USP[2] que desenvolveu o córpus CORAA com cerca de 460 horas de áudios validados.

O modelo Wav2Vec2 ainda pode ser melhorado por meio da adição de modelos de linguagem. Sendo o KenLM bastante usado e possuindo uma integração relativamente fácil no processo de inferência.

O modelo CoquiSTT/DeepSpeech2 não teve um desempenho semelhante devido a esse ser um modelo puramente de treinamento, sem dados pré-treinados anteriormente, o que necessitaria de muito mais horas de áudios rotulados do que as 100 horas disponibilizadas para treino. Para efeito de comparação, a CoquiSTT disponibilza modelos treinados por ela para *download*, e o modelo em inglês conseguiu atingir WER de 4.5% e CER de 1.6%[3], no entanto, o treinamento foi feito com os córpus Common Voice versão 7, LibriSpeech e LibriSpeech Multi-Língua, totalizando cerca de 47 mil horas de áudios. O que está bastante distante da realidade de córpus em português.

Por fim, trabalhos futuros podem se aprofundar em desenvolver uma base mais robusta de áudios rotulados. Outro ponto a ser explorado, é a adaptação do código para usar a nova ferramenta lançada pela HuggingFaces para melhorar o tempo de treinamento e inferência, o Optimum (https://github.com/huggingface/optimum). Esta aplicação promete acelerar o tempo de treinamento em até 30% e o tempo de inferência em até 20%. Com isso, um dos principais problemas sobre o Wav2Vec2 apontado neste trabalho, pode ser amenizado com esta ferramenta. Outra comparação pertinente a ser realizada, é com o mais novo modelo lançado pela empresa Meta, para treinamento multi-modal, ou seja, um

[1] https://huggingface.co/facebook/wav2vec2-base-960h
[2] https://sites.google.com/view/tarsila-c4ai
[3] https://coqui.ai/english/coqui/v1.0.0-huge-vocab

mesmo modelo para treinar tarefas de reconhecimento automático de fala, NLP e visão computacional, o Data2Vec (https://huggingface.co/facebook/data2vec-audio-large-960h). Este modelo superou o Wav2Vec2 em WER no conjunto de teste do LibriSpeech. Então uma análise comparando o treinamento em português pode ser efetuada.

REFERÊNCIAS

ARDILA, R. *et al.* Common voice: A massively-multilingual speech corpus. *In*: **Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)**. [*S.l.: s.n.*], 2020. p. 4211–4215.

BAEVSKI, A. *et al.* wav2vec 2.0: A framework for self-supervised learning of speech representations. **Advances in Neural Information Processing Systems**, Neural information processing systems foundation, v. 2020-December, jun 2020. Available at: https://arxiv.org/abs/2006.11477v3.

CARVALHO, A. **Redes Neurais Artificiais**. 2020. Available at: https://sites.icmc.usp.br/andre/research/neural/.

CHO, K. *et al.* Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. 2014.

COMMONS, W. C. 2021. Available at: https://commons.wikimedia.org/wiki/File:Perceptron_moj.png.

COQUI. **English STT 0.9.3**. https://github.com/coqui-ai/STT-models.

CUTAJAR, M. *et al.* Published in iet signal processing comparative study of automatic speech recognition techniques. p. 1–15, 2013. ISSN 1751-9675. Available at: www.ietdl.org.

FENG, W. *et al.* Audio visual speech recognition with multimodal recurrent neural networks. *In*: . [*S.l.: s.n.*], 2017. p. 681–688.

GEORGESCU, A. L. *et al.* Performance vs. hardware requirements in state-of-the-art automatic speech recognition. **Eurasip Journal on Audio, Speech, and Music Processing**, EURASIP Journal on Audio, Speech, and Music Processing, v. 2021, n. 1, 2021. ISSN 16874722.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep learning. MIT Press, 2016. Http://www.deeplearningbook.org.

GRAVES, A. **Supervised Sequence Labelling with Recurrent Neural Networks**. [*S.l.: s.n.*], 2012. v. 385. ISBN 978-3-642-24796-5.

GRIS, L. R. S. *et al.* Brazilian portuguese speech recognition using wav2vec 2.0. p. 1–15, 2021. Available at: http://arxiv.org/abs/2107.11414.

GROSMAN, J. **HuggingSound: A toolkit for speech-related tasks based on Hugging Face's tools**. 2022. https://github.com/jonatasgrosman/huggingsound.

HANNUN, A. *et al.* Deep speech: Scaling up end-to-end speech recognition. dec 2014. Available at: http://arxiv.org/abs/1412.5567.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-temr memory. 1997. Available at: https://www.bioinf.jku.at/publications/older/2604.pdf.

IBM, C. E. **What are Recurrent Neural Networks? | IBM**. 2020a. Available at: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

IBM, C. E. **What are Recurrent Neural Networks?** 2020b. Available at: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

JURAFSKY, D.; MARTIN, J. **Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition**. [*S.l.: s.n.*], 2008. v. 2. 11-11 p.

KANDA, N. *et al.* End-to-end speaker-attributed asr with transformer. apr 2021. Available at: https://arxiv.org/abs/2104.02128v1.

KOVÁCS, Z. L. **Redes neurais artificiais**. [*S.l.: s.n.*]: Editora Livraria da Fisica, 2002.

LECUN, Y. *et al.* A B7CEDGF HIB7PRQTSUDGQICWVYX HIB edCdSISIXvg5r ‘ CdQTw XvefCdS. **proc. OF THE IEEE**, 1998. Available at: http://ieeexplore.ieee.org/document/726791/#full-text-section.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **BULLETIN OF MATHEMATICAL BIOPHYSICS**, v. 5, 1943.

MINSKY, M.; PAPERT, S. A. Perceptrons: An introduction to computational geometry. **Perceptrons**, The MIT Press, jan 2017.

O’SHAUGHNESSY, D. Invited paper: Automatic speech recognition: History, methods and challenges. **Pattern Recognition**, v. 41, n. 10, p. 2965–2979, 2008. ISSN 00313203.

PONTI, M. A. *et al.* Everything you wanted to know about Deep Learning for Computer Vision but were afraid to ask. 2017.

QUINTANILHA, I. M. End-to-end speech recognition applied to brazilian portuguese using deep learning. 03 2017.

RODRIGUES, M.; PINHEIRO, R. Desenvolvimento de um ambiente de experimentaÇÃo para o reconhecimento de fala utilizando aprendizado profundo: Uma aplicaÇÃo para a lÍngua portuguesa. 2020.

ROSENBLATT, F. **The Perceptron — A Perceiving and Recognizing Automaton – Brain Wars**. 1957. Available at: https://blogs.umass.edu/brain-wars/1957-the-birth-of-cognitive-science/the-perceptron-a-perceiving-and-recognizing-automaton/.

RUMELHART, D. E.; MCCLELLAND, J. L.; ASANUMA, C. Parallel distributed processing: Foundations. **Parallel distributed processing: explorations in the microstructure of cognition**, MIT Press, San Diego, California, p. 45 – 76, 1986. Available at: https://books.google.co.uk/books?id=ktLhoQEACAAJ.

RUMELHART, D. E.; MCCLELLAND, J. L.; ASANUMA, C. **Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations | MIT Press eBooks | IEEE Xplore**. San Diego, California: [*S.l.: s.n.*], 1987. Available at: https://ieeexplore.ieee.org/book/6276825.

SCHNEIDER, S. *et al.* wav2vec: Unsupervised pre-training for speech recognition. **Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH**, International Speech Communication Association, v. 2019-September, p. 3465–3469, apr 2019. ISSN 19909772. Available at: https://arxiv.org/abs/1904.05862v4.

TEVAH, R. T. **Implementação de um sistema de reconhecimento de fala contínua com amplo vocabulário para o português brasileiro.** 2006.

VASWANI, A. *et al.* Attention is all you need. p. 5999–6009, 2017. Available at: https://arxiv.org/abs/1706.03762v5.

WANG YIQIN LU, J. Q. M. **A Dynamic MLP-Based DDoS Attack Detection Method Using Feature Selection and Feedback - Scientific Figure on ResearchGate.** 2019. Available at: https://www.researchgate.net/figure/Topology-of-MLP-classifier_fig3_336433756.

APÊNDICES

APÊNDICE A – COLETA DE DADOS DA BASE PRIVADA

O processo de coleta de dados para a base própria foi feito de duas formas. A primeira foi coletando as ligações em que houve interação do cliente com o sistema de URA. Esse processo foi montado através da construção de uma consulta analítica em SQL. Nessa consulta é calculado o tempo da ligação, o tempo em que o cliente começa falar e quando termina e o caminho do áudio no sistema de arquivos. A partir disso, foi construído um programa em Python para executar essa consulta, e de posse do resultado, verificar se o áudio existe, carregar em memória, fazer um corte extraindo a parte que o cliente fala, e então gerar um novo áudio convertido para WAVE 16Khz apenas com essa parte.

Depois dos áudios segmentados e separados em uma pasta específica, foi criado um programa em Java para se comunicar com o serviço de reconhecimento automático de fala atualmente em uso na empresa em que trabalho. Este programa tem o papel de carregar os dados gerados anteriormente, mandar para o ASR comercial, recuperar a transcrição e criar um arquivo no formato CSV contendo o caminho do arquivo de áudio e a transcrição gerada.

Ao término desse processo, de posse do arquivo CSV, comecei o processo de ouvir cada gravação e comparar com a transcrição gerada. Caso houvesse algum erro, então fosse corrigido o texto gerado. Foi aplicado um processo manual de excluir as frases repetidas várias vezes, porém deixando algumas frases repetidas com interlocutores diferentes.

O segundo processo de coleta de dados, se deu através da construção de um robô, para o Telegram. Esse programa foi inspirado no modelo de uso da plataforma colaborativa do Common Voice, onde os usuários são apresentados a um texto e estes tem de enviar um áudio lendo este texto. Assim, o robô em questão, ao receber uma mensagem de inicialização, oferece um texto e um botão solicitando o envio do áudio daquele texto. Quando o usuário submete o arquivo de áudio, então o robô envia mais uma mensagem perguntando se o usuário deseja confirmar o envio do áudio ou reenviá-lo, por dois botões com essas perguntas. Caso seja confirmado, é enviado outra solicitação perguntando se há interesse em efetuar uma nova submissão com outro texto.

Os textos gerados para esse experimento com o Telegram, foram recuperados de um formulário para criação de URA para um serviço de clínica médica. Foram inseridos textos gerados aleatórios com palavras de contexto médico.

Após todos os usuários terminarem com o processo de envio dos áudios, foi realizado um processo de validação dos dados. Para isso, foi implementado um nesse robô um formulário de verificação, onde é solicitado um áudio enviado com o texto original, então é ouvido a gravação e checado se está condizente, se sim, então é clicado no botão de

validação, caso contrário, é clicado no botão de invalidação.

Por fim, foi implementado a função de exportar os áudios validos para o formato de treinamento nos modelos usados. Nesse caso, o sistema converte todos os áudios para WAVE 16Khz e adiciona um arquivo CSV com as transcrições e local dos arquivos.

APÊNDICE B – APLICAÇÃO DO PROJETO E TESTE DE CARGA

O modelo resultante do treinamento com WavVec2 foi implantado comercialmente na empresa que trabalho. Para que isso fosse possível, foi necessário a implementação de uma solução para servir este modelo. Esta aplicação foi criada usando Python 3.9, FastAPI 0.79 e Docker. Esta aplicação recebe como entrada um áudio em formato WAV 16Khz no corpo da requisição e retorna a respectiva transcrição no formato JSON.

Inicialmente foram feitos testes de carga para se ter conhecimento sobre a quantidade de requisições simultâneas o servidor iria lidar, dado um tempo de resposta aceitável, definido, com base no sistema comercial já usado na empresa, em até 3 segundos para até 10 segundos de áudio. Este experimento foi feito utilizando a aplicação Apache JMeter. Esta aplicação permite, de forma fácil, executar testes de carga.

Foi montado um conjunto de testes para identificar o número de requisições ideal para comportar a meta de até 3 segundos de latência. Com 100 requisições simultâneas, observou-se um tempo médio de 3191ms com máximo de 5459ms, conforme visto na tabela 8. Este resultado não é o ideal, então foi diminuído para 80 requisições, neste experimento ficou em média 2594ms de latência com máximo de 4269ms, conforme tabela 8. Já é um resultado melhor que o anterior, porém as máximas acima do aceitável nos põe a testar com menos requisições. Com 60 requisições, a latência ficou em média em 2015ms e o máximo em 3078ms, conforme tabela 8. Por ficar bem próximo do desempenho esperado, ficou definido um limite de até 60 requisições simultâneas para a configuração do servidor, sendo a mesma do servidor de treinamento definido na tabela 1.

Tabela 8 – Resultado do teste de carga usando o Apache JMeter

Requisições	Média	Min	Max	Desvio Padrão
60	2015	634	3078	724.46
80	2594	463	4269	1080.73
100	3191	458	5459	1438.38