UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

VITOR HENRIQUE DE MORAES ESCALFONI

DIARIZAÇÃO DE LOCUTOR UTILIZANDO REDES NEURAIS RECORRENTES

Vitória-ES

Outubro/2021

VITOR HENRIQUE DE MORAES ESCALFONI

DIARIZAÇÃO DE LOCUTOR UTILIZANDO REDES NEURAIS RECORRENTES

Parte manuscrita do Projeto de Graduação do aluno VITOR HENRIQUE DE MORAES ESCALFONI, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Vitória-ES

Outubro/2021

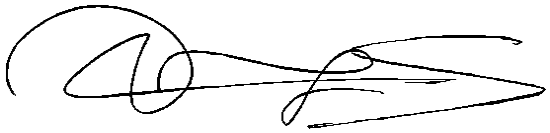
VITOR HENRIQUE DE MORAES ESCALFONI

DIARIZAÇÃO DE LOCUTOR UTILIZANDO REDES NEURAIS RECORRENTES

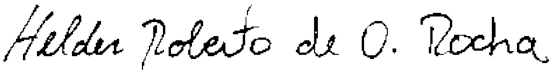
Parte manuscrita do Projeto de Graduação do aluno VITOR HENRIQUE DE MORAES ESCALFONI, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovado em 08 de outubro de 2021.

COMISSÃO EXAMINADORA:



Prof. Dr. Jorge Leonid Aching Samatelo
Universidade Federal do Espírito Santo
Orientador



Prof. Dr. Helder Roberto de Oliveira Rocha
Universidade Federal do Espírito Santo
Examinador

Eng. Lucas Grigoleto Scart
Examinador

Vitória-ES

Outubro/2021

"Se a educação sozinha não transforma a sociedade, sem ela tampouco a sociedade muda."

Paulo Freire

AGRADECIMENTOS

Gostaria de agradecer a todas essas pessoas especiais que estiveram comigo em minha caminhada acadêmica.

Agradeço aos meus pais, Vitor e Erika, e ao meu irmão Talles por todo apoio e incentivo ao longo desses anos. Obrigado por todo carinho, cuidado e por me proporcionarem condições para que eu pudesse me dedicar aos estudos.

Me faltam palavras para agradecer à minha esposa, Lara. O seu apoio e dedicação foram tudo para mim, não teria conseguido sem você e não é exagero. Sou feliz todos os dias por dividir a vida com você.

Sou grato também aos amigos e amigas que tive o prazer de conhecer e dividir essa jornada. Tive a sorte de construir amizades que sei que levarei por toda vida.

Ao meu orientador Jorge, só tenho a te agradecer. Muito obrigado por todo seu tempo, disposição e dedicação durante este período em que desenvolvemos o projeto. Saiba que você tem toda a minha admiração como professor e como pessoa.

À professora Raquel e aos amigos que fiz no Laboratório VIROS (atualmente VISIO) deixo meu muito obrigado. Obrigado por terem me acolhido e me feito sentir parte da família. Foi um prazer conhecer vocês, trabalhar junto e dar boas risadas.

Agradeço à banca examinadora pela aceitação do convite e pelo tempo investido para leitura e avaliação desse trabalho.

Agradeço à Universidade Federal do Espirito Santo pela minha formação. Agradeço aos fundadores e defensores do ensino e da educação, pilar tão fundamental em nossa sociedade. Foi um orgulho imenso poder estudar na UFES e fazer parte deste ambiente engrandecedor. Vida longa ao ensino público, gratuito e de qualidade!

A todos vocês, minha eterna gratidão.

RESUMO

A diarização de locutor consiste em identificar em um segmento de áudio os trechos nos quais existe fala e atribuir aos mesmos rótulos correspondentes com seus respectivos locutores. Em outras palavras, busca-se determinar "quem falou e quando falou", sem que se tenha informações prévias sobre os locutores ou o áudio. A diarização traz melhorias para a indexação e análise de várias categorias de dados de áudio, como consultas médicas, procedimentos jurídicos, apresentações em convenções, vídeos de redes sociais e outros. Através de sua aplicação, também é possível melhorar o desempenho de outras tarefas de processamento de áudio, como a transcrição automática de fala para texto. Este trabalho propõe o estudo e implementação de um sistema de diarização de locutor utilizando redes neurais recorrentes. Serão apresentados uma breve revisão bibliográfica e uma implementação do método considerado o estado da arte em diarização de locutor. Tal sistema substitui a tradicional etapa final de *clustering* por um modelo treinável chamado *Unbounded Interleaved-State Recurrent Neural Network* (UIS-RNN). A implementação foi avaliada no conjunto de dados *VoxConverse* e na faixa 4, de diarização, do desafio *VoxCeleb Speaker Recognition Challenge* 2021 (VoxSRC-21) e comparado com outras soluções submetidas. Estes conjuntos são compostos por áudios com múltiplos locutores e outros fatores que dificultam a tarefa de diarização, como falas sobrepostas e variados ruídos de fundo. O sistema implementado foi totalmente treinado de forma supervisionada utilizando bases de dados de mesmo domínio do conjunto do desafio. Ao ser avaliado na competição, o sistema obteve DER $61,3\%$ e JER $57,6\%$, resultados pouco competitivos. Motivos e possíveis melhoras destes resultados são indicados no apartado de conclusões deste trabalho.

Palavras-chave: Diarização de locutor; Redes neurais recorrentes; Processamento de fala.

ABSTRACT

Speaker diarization consists in identifying the segments of an audio in which there is speech and assigning them to their respective speakers. In other words, it seeks to determine "who spoke and when", without having prior information about the speakers or the audio. Diarization brings improvements for indexing and analyzing various categories of audio data, such as medical consultations, legal proceedings, convention presentations, social media videos, and others. Through its application, it is also possible to improve the performance of other audio processing tasks, such as automatic speech-to-text transcription. This paper proposes the study and implementation of a speaker diarization system using recurrent neural networks. A brief literature review and an implementation of the method considered the state of the art in speaker diarization will be presented. Such system replaces the traditional final step of clustering with a trainable model called Unbounded Interleaved-State Recurrent Neural Network (UIS-RNN). The implementation was evaluated on the VoxConverse dataset and the diarization track of the VoxCeleb Speaker Recognition Challenge 2021 (VoxSRC-21) and compared with other submitted solutions. These sets contain audios with multiple speakers and other factors that make the diarization task difficult, such as overlapping speech and varied background noise. The implemented system was fully trained in a supervised manner using databases of the same domain as the challenge set. When evaluated in the competition, the system obtained uncompetitive results for DER 61.3% and JER 57.6%. Reasons and possible improvements of these results are given in the conclusions section of this work.

Keywords: Speaker diarization; Recurrent neural networks; Speech processing.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

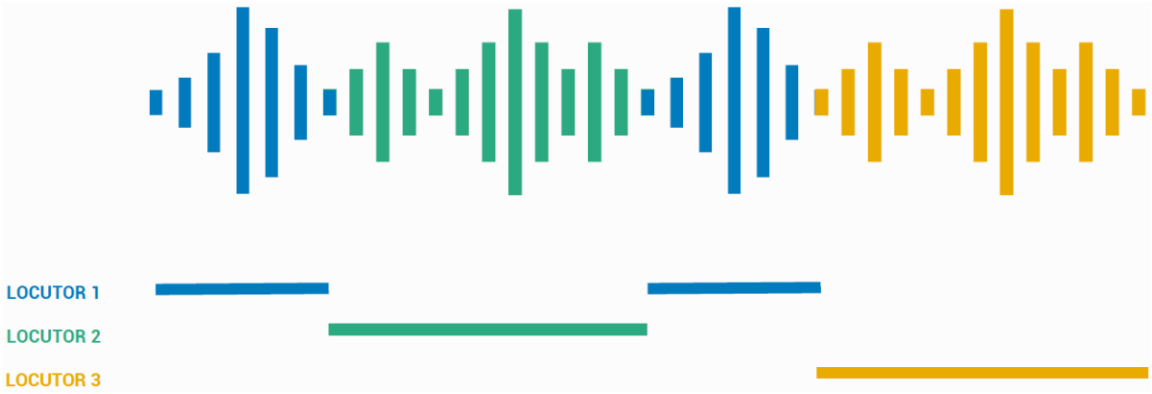
ASR	*Automatic Speech Recognition*
DER	*Diarization Error Rate*
DNN	*Deep Neural Network*
GPU	*Graphics Processing Unit*
GRU	*Gated Recurrent Unit*
JER	*Jaccard Error Rate*
LSTM	*Long Short-Term Memory*
MFCC	*Mel-Frequency Cepstral Coefficients*
RNN	*Recurrent Neural Network*
RTTM	*Rich Transcription Time Marked*
UIS-RNN	*Unbounded Interleaved-State Recurrent Neural Network*
UFES	Universidade Federal do Espírito Santo
VAD	*Voice Activity Detection*

SUMÁRIO

1 INTRODUÇÃO

Diarização, derivada da palavra inglesa *diarize*, significa fazer anotações ou registrar um evento. Quando aplicada ao processamento de áudio, a diarização é responsável por distinguir “quem fala e quando fala”, uma vez que detecta e atribui segmentos de falas aos seus respectivos locutores. A partir do resultado da diarização, também é possível extrair outros dados, tais como: tempo de fala, sobreposição de falas, troca de locutores e momentos sem atividade de voz (PARK et al., 2021).

Figura 1 – Diarização de locutor: "quem falou quando?"



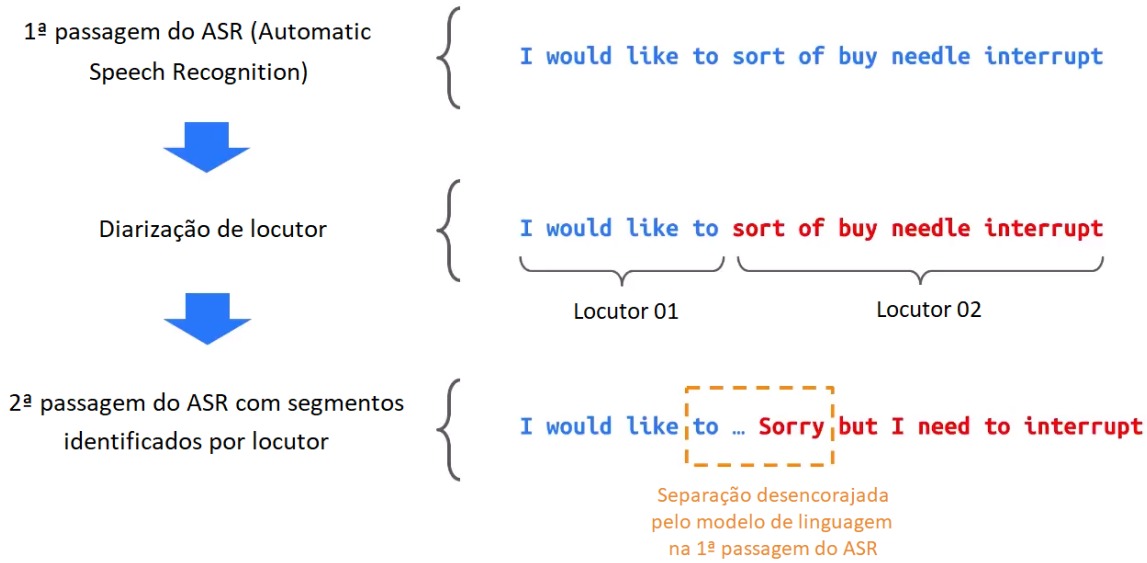
Fonte: Produção do próprio autor.

A obtenção de tais informações possibilita a melhoria de desempenho em sistemas diversos, como o de *Automatic Speech Recognition* (ASR) ou Reconhecimento Automático de Fala em tradução livre (SHAFEY; SOLTAU; SHAFRAN, 2019). A aplicação da diarização minimiza erros provenientes de modelos linguísticos, conforme o exemplo da Figura 2. Além disso, a diarização contribui para diversos outros cenários do cotidiano, como a transcrição automática de fala entre cliente e atendentes em *call centers*, processos judiciais, reuniões de negócios, telemedicina, entre outros (PARK et al., 2021). As pesquisas sobre processamento de áudio têm sido impulsionadas pela necessidade de recuperar informações a partir desses grandes volumes de áudio, de maneira automatizada e efetiva.

Os sistemas de diarização de locutor atualmente se baseiam em segmentar o áudio em partes menores e extrair informações acústicas destes segmentos. Posteriormente, agrupa-se os segmentos utilizando algoritmos de *clustering* de acordo com as informações extraídas. Desse modo, espera-se que cada grupo contenha os segmentos de fala de cada locutor. Estes são considerados os sistemas de base para diarização de locutor.

Embora os sistemas de diarização atualmente tenham resultados satisfatórios, ainda há espaço para melhorias. Um desses pontos refere-se ao agrupamento. A maior parte dos algoritmos de *clustering* não consideram a ordenação de entrada dos dados, nem sua

Figura 2 – Exemplo de aplicação: aumento da fidelidade de um ASR



Fonte: Wang et al. (2018).

Nota: Tradução pelo próprio autor.

relação temporal. Eles também não são capazes de aprender a partir de exemplos, pois são baseados em heurísticas e regras, e não nos dados e suas características. Como a diarização de locutor é um problema dentro do domínio de processamento de fala, é importante incorporar informações temporais, bem como aprender a partir de exemplos anteriores.

O método desenvolvido por Zhang et al. (2019) busca aprimorar esta fase de agrupamento, tirando proveito das propriedades acústicas dos dados. Propõe-se a utilização de uma rede neural recorrente *unbounded interleaved-state* (UIS-RNN) em substituição aos tradicionais algoritmos de *clustering*. Desta forma, o sistema de diarização de locutor torna-se treinável em todas as etapas e completamente supervisionado.

Este sistema consagrou-se como estado da arte quando divulgado em 2019, substituindo a proposta anterior (WANG et al., 2018) do mesmo grupo de pesquisa. Ambos trabalhos foram avaliados utilizando o conjunto de dados 2000 NIST *Speaker Recognition Evaluation* (PRZYBOCKI, 2001), disco 8, geralmente referido apenas como *CALLHOME* na literatura. Considerado o *benchmark* para sistemas de diarização de locutor, tal conjunto é composto por conversas extraídas de ligações telefônicas distribuídas entre 6 linguagens diferentes.

Este projeto de pesquisa implementa o método que utiliza redes neurais recorrentes com *unbounded interleaved-state*, o qual foi considerado estado da arte de diarização de locutor reportando $7,6\%$ de taxa de erro de diarização ao ser avaliado no NIST SRE 2000

CALLHOME. O sistema implementado foi validado no conjunto de dados *VoxConverse* e submetido ao desafio *VoxCeleb Speaker Recognition Challenge* 2021 (VoxSRC-21), de modo a avaliar seu desempenho em bases de dados mais abrangentes e complexas. Os conjuntos de dados usados na avaliação do *VoxCeleb* foram obtidos a partir de vídeos do *YouTube* e consistem em áudios de qualidade variada. O material possui múltiplos locutores, falas sobrepostas e outros fatores que dificultam o processamento do áudio, tais como conversas de fundo, risadas e ruídos.

O sistema implementado foi avaliado utilizando métricas de que determinam sua taxa de erro de diarização. A mais utilizada pela literatura é a DER, do inglês *Diarization Error Rate*. Recentemente, também passou-se a usar a métrica *Jaccard Error Rate* (JER) (YUAN; CHAO; LO, 2017). Tais métricas combinam o percentual de falas que não foram detectadas, as que foram detectadas incorretamente e também os locutores classificados erroneamente.

1.1 Objetivos

Objetivo Geral

- Implementar um sistema de diarização de locutor baseado no método considerado estado da arte, utilizando redes neurais recorrentes. Avaliar o sistema implementado usando o conjunto de dados *VoxConverse* e do desafio *VoxCeleb Speaker Recognition Challenge* 2021 (VoxSRC-21).

Objetivos Específicos

- Revisar os conceitos de diarização de locutor e as etapas e estratégias de um sistema tradicional;
- Combinar diferentes bibliotecas de código aberto para implementar um sistema de diarização de ponta a ponta, com base no projeto que utiliza redes neurais recorrentes *unbounded interleaved-state* como *clustering*;
- Testar e avaliar o sistema implementado no conjunto de dados *VoxConverse* e na faixa de diarização do desafio *VoxCeleb Speaker Recognition Challenge* 2021 (VoxSRC-21);
- Comparar os resultados obtidos com outras abordagens submetidas ao desafio.

1.2 Estrutura do Texto

O presente trabalho está estruturado da seguinte maneira:

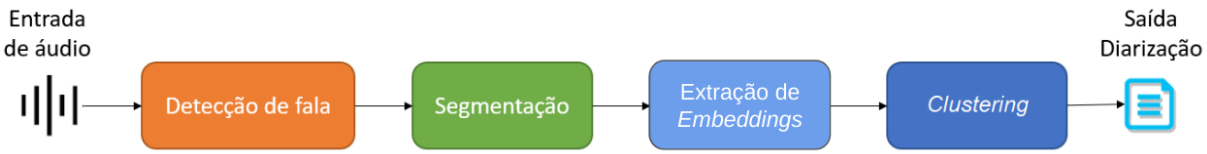
- **Referencial Teórico**: este capítulo inicial tem como objetivo apresentar uma breve revisão dos métodos utilizados para diarização de locutor. Serão discutidos os blocos que compõem sistemas tradicionais de diarização;
- **Proposta**: neste capítulo é apresentado o sistema escolhido para ser implementado neste projeto e suas características específicas;
- **Experimentos e Resultados**: neste capítulo são detalhados os experimentos feitos com o sistema implementado, bem como seus resultados;
- **Conclusão**: no capítulo final deste trabalho são discutidos os resultados obtidos nos experimentos e são feitas propostas para melhorias e projetos futuros.

2 REFERENCIAL TEÓRICO

2.1 Introdução

De acordo com Park et al. (2021), um sistema tradicional de diarização de locutor é composto pela sequência de blocos representada de forma simplificada na Figura 3.

Figura 3 – Sistema tradicional de diarização de locutor



Fonte: Produção do próprio autor.

Inicialmente, existe um bloco de detecção de atividade de fala, referido como VAD, do inglês *Voice Activity Detector*. Este bloco é responsável por detectar as ocorrências de falas dentro de um áudio, distinguindo partes não relacionadas com a fala, como silêncio, risadas e ruídos de fundo indesejados.

A próxima etapa do processo é a segmentação da fala. O objetivo é dividir os segmentos de modo que cada um deles contenha apenas um locutor. Em abordagens clássicas, a segmentação costuma ser determinada a partir da detecção de mudança de locutor. Entretanto, mais recentemente a segmentação utilizando janelas fixas, com uma parcela de sobreposição entre si, vem sendo mais utilizada (PARK et al., 2021).

Em seguida, o processamento se dá pelo bloco de extração de *embeddings*. Nesta etapa, o objetivo é extrair informações que diferenciam os locutores presentes no áudio. E por fim, ao obter as representações vetoriais das características dos segmentos de áudio, é feito um agrupamento utilizando algoritmos de *clustering*. Cada *cluster*, idealmente, representa um dos locutores presentes no áudio, agrupando todos os segmentos pelo seu locutor respectivo.

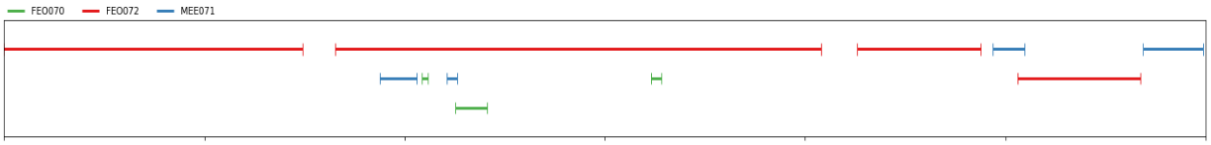
Tais blocos são descritos em maior detalhe nas seções a seguir.

2.2 *Voice Activity Detection*

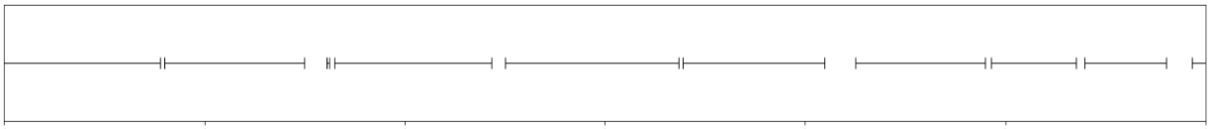
A etapa do VAD é responsável por reconhecer e separar os segmentos contendo fala, descartando trechos de não fala, como ruídos e conversas de fundo. Este bloco é crucial no processo de diarização, uma vez que é passível a criar erros que podem se propagar em todo o processo subsequente. Na Figura 4 está ilustrada a resposta de um VAD, marcando onde há atividade de fala em relação ao áudio de referência. Nota-se que a Figura 4b foi produzida a partir de um sistema real e não ilustra a resposta perfeita esperada.

Figura 4 – Resposta de um sistema de detecção de atividade de fala.

(a) Referência



(b) Detecção de Atividade de Fala



Fonte: Bredin et al. (2020)

De maneira geral, um VAD é subdividido em duas partes. A primeira envolve a extração de características, onde obtém-se propriedades acústicas como, por exemplo, os coeficientes mel-cepstrais popularmente referidos como MFCCs (*Mel-Frequency Cepstrum Coefficients*) (WANG; XU; LI, 2011). A segunda parte do VAD consiste um modelo classificador, responsável por decidir se o segmento contém fala ou não.

As técnicas de detecção de fala são diversas e têm sido continuamente estudadas e aprimoradas. Atualmente, as solução com modelos classificadores baseiam-se em *Gaussian Mixture Models* (GMM) (NG et al., 2012), *Hidden Markov Models* (HMM) (SARIKAYA; HANSEN, 1998) ou em *Deep Neural Networks* (DNNs), Redes Neurais Profundas, em tradução livre (DRUGMAN et al., 2015). Também existem abordagens que envolvem o uso de tom de voz, sinais de energia e relação sinal-ruído (SNR) para determinar os segmentos contendo fala, como feito por Morita, Lu e Unoki (2014).

2.3 Segmentação

A segmentação tem como objetivo fragmentar o áudio processado em segmentos menores de forma a atribuí-los para cada locutor. Esta etapa têm sido aplicada de duas maneiras principais: por detecção de mudança de locutor ou por segmentação uniforme (PARK et al., 2021).

A detecção de segmentos usando mudança de locutor já foi considerada o padrão dos sistemas de diarização. Nesta metodologia, consideram-se duas hipóteses, onde:

- A hipótese H_0 assume que as amostras adjacentes são do mesmo locutor;
- A hipótese H_1 assume que as amostras adjacentes são de locutores diferentes.

Para o teste dessas hipóteses, diferentes abordagens podem ser aplicados sendo o método BIC (*Bayesyan Information Criterion*) (DELACOURT; KRYZE; WELLEKENS, 1999) o comumente usado para detectar os pontos de mudança de locutor.

Entretanto, a segmentação por mudança de locutor produz fragmentos inconsistentes e de tamanhos variados, o que introduz uma variabilidade adicional nas características de locutor, que pode causar infidelidade em sua representação. Em contrapartida, surge a segmentação uniforme, utilizando-se de *i-vectors* (DEHAK et al., 2010) e *embeddings* baseados em DNN (VARIANI et al., 2014). Neste tipo de metodologia, a segmentação do áudio é feita em janelas fixas, com sobreposição entre si. Assim, o resultado de saída da diarização do locutor permanece com a mesma unidade de duração.

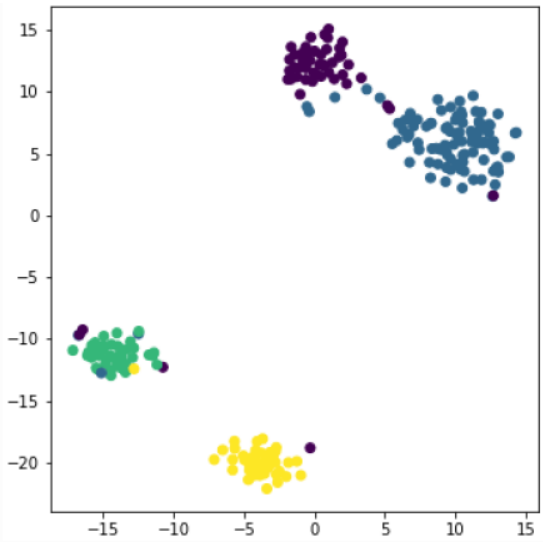
Mesmo com essa vantagem, o processo de segmentação uniforme pode acarretar em alguns problemas. Essa segmentação introduz um *trade-off* relacionado à duração do segmento. Isto é, o segmento precisa ser suficientemente curto para garantir que não possui múltiplos locutores e, ao mesmo tempo, deve ter duração o bastante para capturar informações acústicas necessárias de forma a representar o locutor satisfatoriamente.

2.4 Extração de *Embeddings*

Embeddings são vetores de altas dimensões traduzidos em um espaço de baixa dimensão de forma a representar dados como textos, áudios, imagens, e mais. Estes podem ser

obtidos por diferentes processos que geralmente são desenhados para criar *embeddings* com características que podem ser usadas para diferenciá-los entre si. No contexto de diarização, os *embeddings* representam as características de voz do locutor em questão em determinado segmento, com o objetivo de poder identificar se diferentes segmentos pertencem ao mesmo locutor ou não. Na Figura 5 está representado um grupo de *embeddings* pertencentes a diferentes locutores, diferenciados pela cor. Tal figura foi produzida utilizando o algorítmo t-SNE (*t-distributed stochastic neighbor embedding*).

Figura 5 – Visualização de *embeddings* usando o algoritmo t-SNE



Fonte: Bredin et al. (2020).

Portanto a etapa de extração de *embeddings* é responsável por obter representações discriminativas dos locutores a partir dos segmentos de áudio pré-processados.

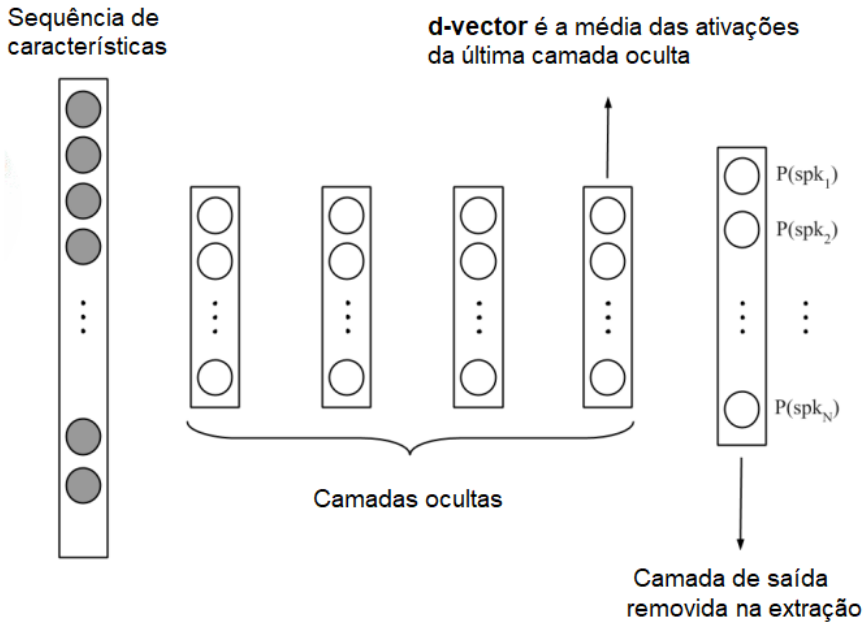
Desde o advento do Aprendizado profundo, nos anos 2010, surgiu um considerável número de pequisas que tiram proveito das poderosas capacidades de redes neurais para aprimorar os sistemas de diarização de locutor. Os novos modelos de extração contribuíram para a melhora da performance geral, facilitando o treinamento com ainda mais dados e adicionando robustez contra a variabilidade e às condições acústicas. Entre esses modelos, tem-se o *d-vector* e o *x-vector*.

O *d-vector* é um modelo de extração que utiliza as DNN para obter as características do locutor. De acordo com Variani et al. (2014), uma DNN é treinada para mapear as características acústicas com o objetivo de identificar o seu locutor correspondente. Durante o processo, o modelo é computado como a média de ativações derivadas na última camada oculta da DNN, referida como *deep vector* ou simplesmente *d-vector*, conforme representado na Figura 6. Na etapa de avaliação, são consideradas as distâncias entre o

alvo *d-vector* e o teste *d-vector*, utilizando-se de similaridade de cosseno.

É importante citar que os *d-vector* também podem ser obtidos utilizando o mesmo processo em redes *Long Short-Term Memory* (LSTM), como proposto por Wan et al. (2018).

Figura 6 – Modelo d-vector para verificação de locutor



Fonte: Variani et al. (2014).

Nota: Tradução pelo próprio autor.

2.5 *Clustering*

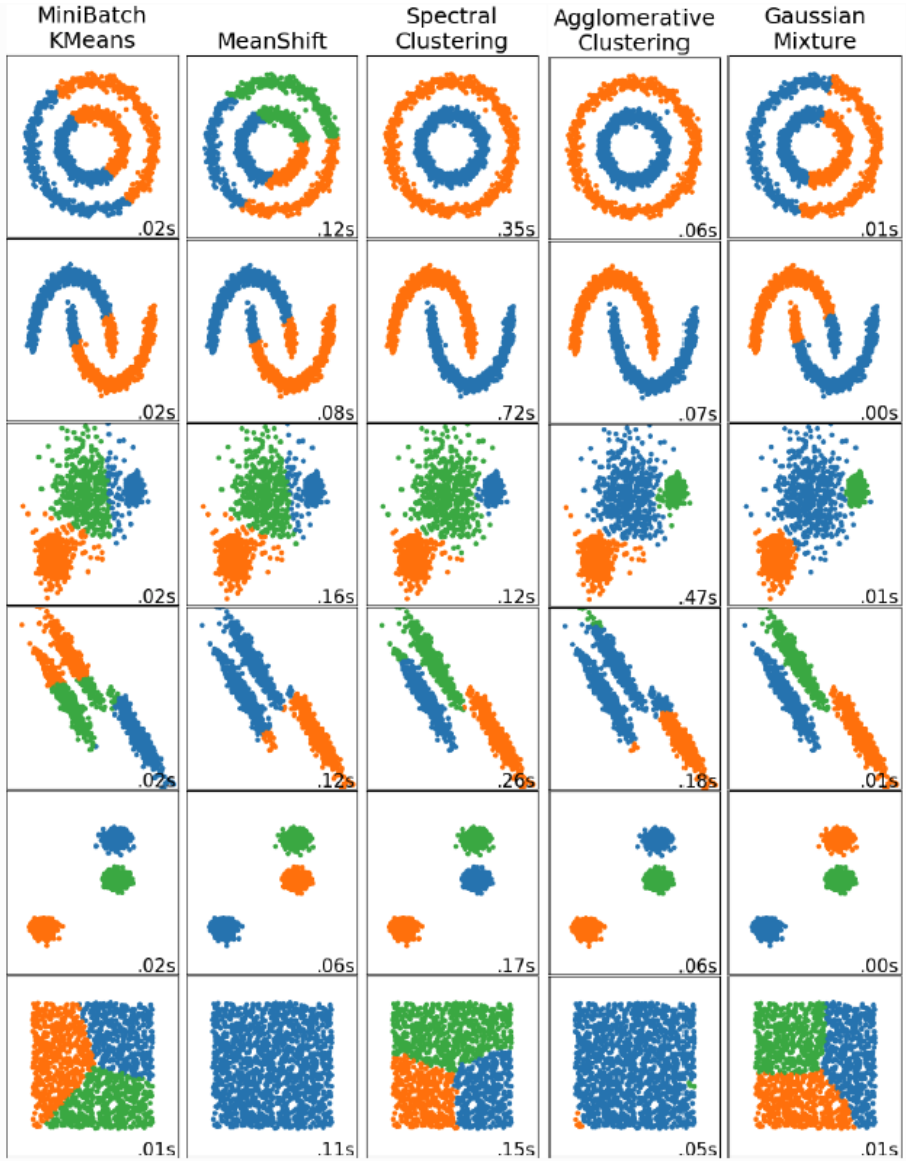
Após extração de *embeddings*, um algoritmo de *clustering* deve ser aplicado para dar prosseguimento à diarização de locutor. Tais algoritmo podem ser subdivididos em categorias de acordo com a sua latência, sendo elas:

- *Online clustering*: Os rótulos dos locutores são atribuídos assim que os segmentos tornam-se disponível, desconsiderando segmentos futuros.
- *Offline clustering*: Os rótulos dos locutores são atribuídos somente depois que todos os *embeddings* tornam-se disponíveis.

De maneira geral, os algoritmos *offline* apresentam um desempenho mais efetivo, uma vez que possuem uma maior quantidade de informações contextuais em comparação aos

algoritmos *online*. No entanto, a escolha de cada categoria deve ser feita baseada no contexto de uma aplicação. Na Figura 7 estão ilustrados alguns dos métodos de *clustering* do tipo *offline* mais utilizados em sistemas de diarização de locutor. GMM, K-Means (LIKAS; VLASSIS; VERBEEK, 2003) e *mean shift* (SENOUSSAOUI et al., 2013) são considerados básicos e não são comumente usados atualmente. Já o *spectral clustering* (WANG et al., 2018) e AHC (*Agglomerative Hierarchical Clustering*) (HAN; NARAYANAN, 2007) estão entre os mais empregados e com melhores resultados

Figura 7 – Comparação entre algoritmos de *clustering* usados em diarização de locutor



Fonte: scikit-learn desenvolvido por Pedregosa et al. (2011).

Alguns dos principais métodos de *clustering* do tipo *online* serão descritos nas seções a seguir.

2.5.1 *Naive* e *Links Clustering*

O *Naive clustering* é um algoritmo base, onde um limiar é aplicado nas similaridades entre os segmentos de *embeddings*. A métrica utilizada para o cálculo é a similaridade por cosseno.

Neste algoritmo, cada *cluster* é representado pelo centroide de seus *embeddings* correspondentes. Quando um novo segmento de *embedding* torna-se disponível, computa-se as similaridades com os centroides dos *embeddings* já existentes. Caso seja menor que o limiar, um novo *cluster* é criado, contendo apenas este *embedding*. Caso contrário, ele é adicionado a um *cluster* com características mais semelhantes e o centroide é atualizado.

Já o *Links clustering* (MANSFIELD et al., 2018) é um aprimoramento do algoritmo de *clustering* anterior. Esta abordagem estima a distribuição de probabilidade de cada *cluster* com base em seus vetores constituintes. Usa-se essas estimativas para atribuir novos vetores aos *clusters* e para atualizar as distribuições de cada vetor adicionado. Concomitantemente, essa etapa de atualização corrige as tarefas de *cluster* anteriores, ao passo que melhora o modelo interno no decorrer do processo.

2.5.2 *Spectral clustering*

O *spectral clustering* é uma abordagem popular no sistema de diarização do locutor e possui diversas variações. O método aplicado por Wang et al. (2018) envolve uma série de etapas de processamento.

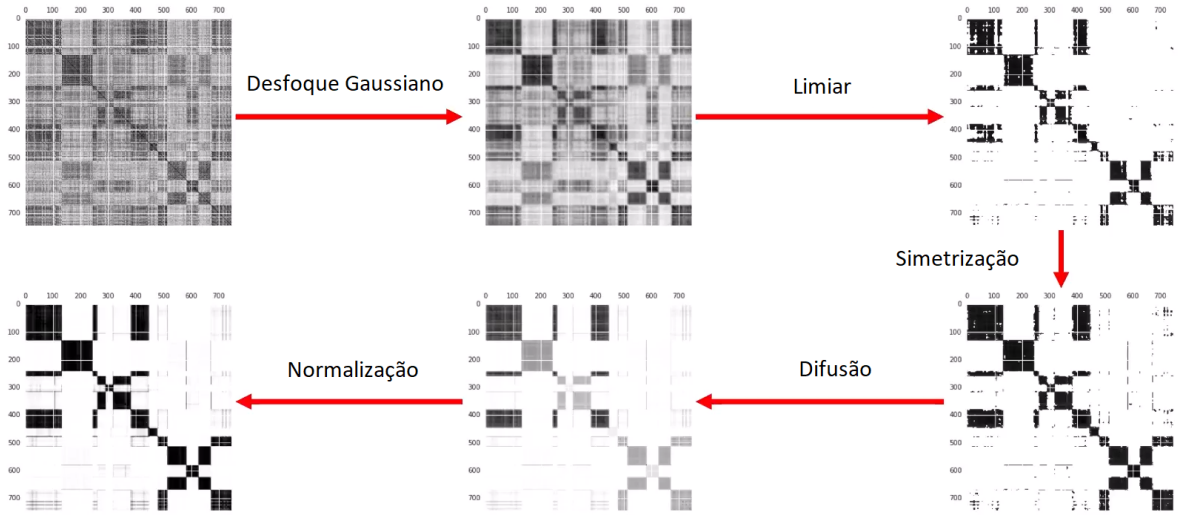
Primeiramente é feita a construção de uma matriz de afinidade $\mathbf{A}$, onde A_{ij} é a similaridade de cosseno entre os *embeddings* dos i-ésimo e j-ésimo segmentos, quando $i \neq j$. Os elementos diagonais recebem o valor máximo de cada linha: $A_{ii} = max_{j \neq i}\{A_{ij}\}$.

Em seguida, aplica-se uma sequência de operações de refinamento na matriz $\mathbf{A}$, representadas na Figura 8 e descritas abaixo:

- Desfoque Gaussiano (*Gaussian Blur*) com desvio padrão σ, feito para suavizar os dados e reduzir os efeitos de *outliers*.
- Limiar por linha (*Row-wise Thresholding*): Para cada linha, definir os elementos menores que o p-percentual desta linha em relação a zero. Essa operação serve para zerar afinidades entre *embeddings* pertencentes a locutores distintos.

- Simetrização (*Symmetrization*): $Y_{ij} = max(X_{ij}, X_{ji})$. Restaura a simetria da matriz, o que é crucial para o algoritmo *spectral clustering*.
- Difusão (*Diffusion*): $\mathbf{Y} = \mathbf{X}\mathbf{X}^T$. Esta etapa busca definir melhor os limites entre seções da matriz de afinidade pertencentes a locutores distintos.
- Normalização máxima por linha (*Row-wise Max Normalization*): $Y_{ij} = X_{ij}/max_k\{X_{ik}\}$. Serve para re-escalar o espectro da matriz, de forma a assegurar que efeitos indesejados de escala não ocorram durante a etapa subsequente de *spectral clustering*.

Figura 8 – Etapas de refinamento do *spectral clustering*



Fonte: Wang et al. (2018).

Nota: Tradução pelo próprio autor.

Estes refinamentos são cruciais para o sucesso do algoritmo. Eles atuam tanto para suavizar quanto para diminuir os ruídos dos dados no espaço de similaridade. Os refinamentos são baseados na localidade temporal da fala; sendo que os segmentos de discursos contíguos devem ter *embeddings* similares e, portanto, valores semelhantes na matriz de afinidade.

Depois que todas as etapas de refinamento foram aplicadas, é feita a decomposição por autovalores na matriz de afinidade refinada. Sejam os n autovalores: $\lambda_1 > \lambda_2 > \ldots > \lambda_n$. Usa-se o máxima razão entre autovalores sucessivos (*eigengap*) para determinar o número de locutores (*clusters*) $\tilde{k}$, representado na Equação (2.1).

$$\tilde{k} = \arg \max_{1 \leq k \leq n} \frac{\lambda_k}{\lambda_{k+1}} \tag{2.1}$$

Após determinar o número de *clusters*, usa-se os autovetores $v_1, v_2, \ldots, v_{\tilde{k}}$ correspondentes ao maior $\tilde{k}$ para realizar a redução da dimensão dos *embeddings* de cada segmento. No caso, substitui-se o i-ésimo *embedding* pelo seus autovetores correspondentes: $e_i = [v_{1i}, v_{2i}, \ldots, v_{\tilde{k}i}]$.

Por fim, os *embeddings* de dimensão reduzida (projeções) são agrupados utilizando algum algoritmo de *clustering* como o *K-Means* por exemplo, de forma que são obtidos resultados melhores de agrupamento.

Todo este processo executado pelo método descrito tem como objetivo mitigar problemas enfrentados na etapa de *clustering* decorrentes das propriedades e natureza de dados de áudio. Tais propriedades dificultam e pioram o resultado de algoritmos em geral usados para agrupamento. Dentre essas propriedades inclui-se:

- Distribuição Não-Gaussiana: Dados de fala geralmente tem essa característica, que faz com que o centroide de seu *cluster* não seja uma representação suficiente para diferenciar de outros;
- Desbalanceamento de *Cluster*: É comum que um ou mais locutores falem mais que outros, de forma que estes tenham um maior número de amostras. Deste modo algoritmos menos sofisticados podem erroneamente dividir um *cluster* grande em vários outros menores;
- Estrutura Hierárquica: Locutores pertencem à vários grupos de acordo com seu gênero, idade, sotaque, entre outros. Tais estruturas trazem problemas pois a diferença de voz entre um homem e uma mulher é muito maior que a diferença entre dois homens, por exemplo. Isto pode acarretar no agrupamento não por locutor, e sim por grupos de características similares.

2.6 Procedimentos extras

Os blocos anteriormente apresentados constituem um sistema de diarização padrão e hoje considerados de base. Entretanto existem vários outros processos e etapas que podem e são executadas por diferentes sistemas para melhorar o processo de diarização ou solucionar algum problema específico de domínio. Tais operações se dividem em duas grandes categorias de pré e pós processamento.

As operações extras de pré processamento também são conhecidas por processamento *front-end* pois trabalham o áudio em si e suas características. Tem o processo de aprimoramento de fala ou *denoising* (LU et al., 2013) cujo objetivo é filtrar ruídos indesejados do áudio sem perder qualidade no sinal de interesse (LU et al., 2013). Similarmente tem-se o processo de *dereverberation* que busca remover os efeitos de reverberação do áudio, geralmente causado pelos microfones usados na gravação, e que podem deteriorar a qualidade do processo de diarização (NAKATANI et al., 2010). Nesta etapa pode ser empregada também a separação de fala, de modo a evitar falas sobrepostas (CHEN et al., 2020).

A outra categoria é de pós processamento, na qual geralmente é empregada a etapa de re-segmentação que é um processo para refinar os resultados obtidos na etapa de *clustering* (KENNY; REYNOLDS; CASTALDO, 2010). Por fim é possível fundir diferentes sistemas de diarização e utilizar sua combinação para melhorar os resultados obtidos pelo sistema como um todo (STOLCKE; YOSHIOKA, 2019).

3 PROPOSTA

3.1 Introdução

Como visto no Capítulo 2, existem diferentes formas de construir um sistema de diarização, cada um com sua particularidade, propósito e limitações. Para a execução deste projeto, foi proposta a escolha e implementação de um sistema que tivesse bons resultados de diarização e pelo menos parte do código aberto. Desta forma escolheu-se o sistema proposto por Zhang et al. (2019) no artigo intitulado *"Fully Supervised Speaker Diarization"*, considerado estado da arte quando publicado, reportando $7,6\%$ de taxa de erro de diarização ao ser avaliado no NIST SRE 2000 CALLHOME. Apesar de atualmente existirem trabalhos com resultados marginalmente melhores, tal escolha se deu por conta das características do sistema. Por ser totalmente supervisionado, é possível treiná-lo e adaptá-lo a diferentes cenários, tornando-o robusto e passível de ser usado em variadas aplicações. Outro ponto de interesse é sua inferência que pode ser realizada de forma *online*, ou seja, não requer o áudio inteiro para realizar o processamento. Assim, é possível obter o resultado da diarização em transmissões ao vivo por exemplo, processando um fluxo contínuo de áudio, sem perda de desempenho.

Neste capítulo será apresentada a implementação do sistema de diarização de locutor desenvolvido com base no sistema proposto por Zhang et al. (2019). As etapas de processamento foram divididas em três grandes blocos, apresentados nas próximas seções.

3.2 Pré-processamento

Neste primeiro bloco, o áudio é processado e segmentado de modo que na próxima etapa seja possível obter seus *embeddings*. Inicialmente, carrega-se o arquivo de áudio que então é processado pelo bloco VAD, o qual retorna apenas os trechos de áudio que possuem fala detectada e sua marcação de tempo. A biblioteca utilizada foi a *py-webrtcvad*, que é uma interface *Python* do VAD produzido pela *Google* para o projeto *WebRTC*. Este VAD é amplamente utilizado por ser considerado rápido, de baixa latência, eficiente e gratuito. Seu funcionamento é baseado na classificação de trechos (com fala ou não) usando máxima verossimilhança com modelos GMM.

A solução proposta utiliza segmentos de tamanho fixo, os quais serão processados e

atribuídos para seu respectivo locutor. Portanto, divide-se o áudio em janelas de 240ms de duração com passos de 120ms (50%) entre elas, sobrepondo-as. Então, de cada janela são extraídos *log-mel-filterbank energies* de dimensão 40 para serem usados como entrada na rede responsável por gerar os *d-vectors*.

Este bloco foi desenvolvido com base no código disponibilizado por Volek (2020). O tamanho das janelas e o passo entre elas foi o recomendado e utilizado pelos autores no trabalho base.

3.3 Extração de *embeddings* utilizando LSTM

Para esta tarefa, a solução proposta utiliza uma rede LSTM desenvolvida anteriormente por Wan et al. (2018) e Wang et al. (2018), publicados pelo mesmo grupo de pesquisa responsável pelo desenvolvimento da UIS-RNN. Tal rede foi treinada para a tarefa de verificação de locutor, independente do uso de anotações, também conhecida como *Text Independent Speaker Verification* ou TI-SV. Portanto, é uma rede capaz de obter representações relevantes (*embeddings*) de cada intervalo de fala, que podem posteriormente ser atribuídas ao devido locutor.

Nesta etapa, as características de cada janela de áudio determinada no bloco anterior são usadas como entradas em uma rede recorrente constituída de células LSTM. Os vetores obtidos na saída, extraídos do último *frame* da rede, são usados como a representação de janela processada e referidos como *d-vectors*. A representação deste processo pode ser visto na Figura 9. Nota-se que a LSTM utilizada é do tipo *many-to-one*.

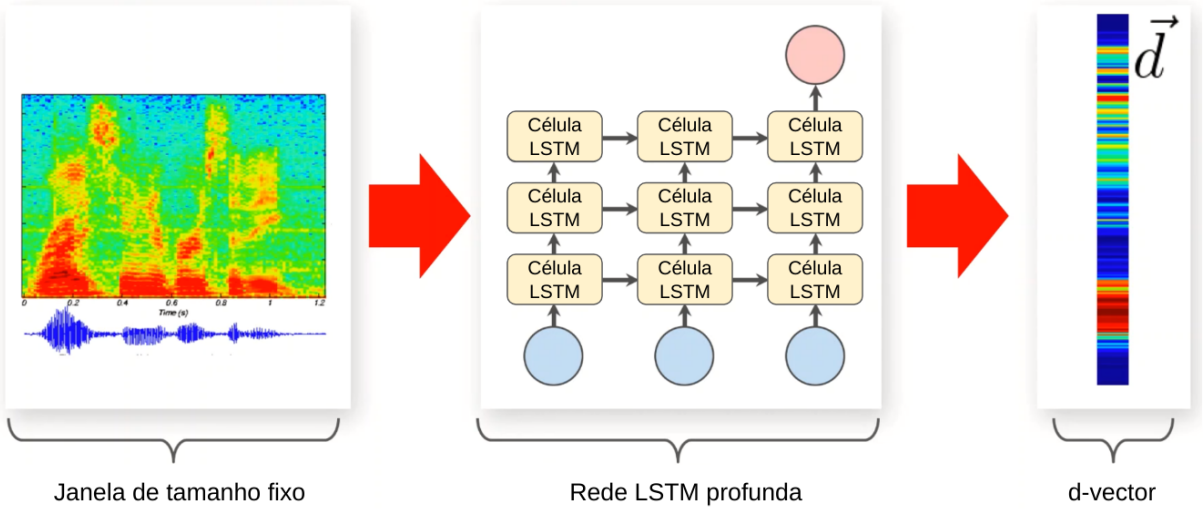
O modelo possui uma estrutura de três células LSTM, e cada uma delas possui 768 unidades. Na camada de saída da rede existe camada linear com projeção de 256 unidades, o que determina a dimensão dos *d-vectors* e consequentemente dos *embeddings* de saída.

Em seguida calcula-se a norma L2 (3.1) de cada um dos *d-vectors* obtidos que, por fim, são agregados em segmentos de tamanho fixo. No caso, foram utilizados segmentos de 400ms, que determinam a resolução do sistema de diarização.

$$d_w = \frac{\overrightarrow{d}}{|| \overrightarrow{d} ||^2} \tag{3.1}$$

A cada trabalho citado, esta rede tem sido retreinada e aperfeiçoada. Em seu modelo

Figura 9 – Diagrama do sistema de extração de *d-vectors*



Fonte: Wan et al. (2018)

Nota: Tradução do próprio autor.

mais recente, foram usados mais de 60 milhões de uterâncias[1] e 150 mil falantes de dados anônimos privados do *Google*. Alguns outros conjuntos de dados públicos também foram utilizados para compor o treinamento. Em suas publicações, sempre é destacada a importância de se ter um sistema bem treinado, com uma grande quantidade de dados, para obter bons *embeddings*.

Por conta do escopo deste projeto e levando em consideração a dificuldade em treinar um modelo tão robusto, optou-se pela utilização de um modelo pré-treinado para realizar esta tarefa. O modelo escolhido foi o desenvolvido por Lobo (2019), que disponibilizou o código para treinamento e validação, bem como os pesos da rede pré-treinada. Parte do código para extração de *embeddings* foi aproveitada deste repositório, que utiliza o *framework Tensorflow* para definir e executar a rede.

Os bancos de dados utilizados para treinamento do modelo estão representados na Tabela 1. As características de cada banco são apresentadas em maior detalhe no Anexo B. O ponto a ser observado aqui é a disparidade em relação a quantidade de dados utilizados no modelo de extração de *embeddings* utilizado nesse projeto e no modelo treinado pela *Google*.

[1] Uma uterância é tida como uma unidade da fala. Consite de um trecho contínuo de fala que começa e termina com uma pausa clara. No caso das línguas orais é geralmente, mas nem sempre, delimitada pelo silêncio.

Tabela 1 – Bancos utilizados para treinar o modelo de extração de *embeddings*

	Total de locutores	**Duração total**
LibriSpeech	2338	$\sim$ 1000 horas
VCTK	110	44 horas
VoxCeleb 1	1251	352 horas
VoxCeleb 2	6112	2442 horas

Fonte: Produção do próprio autor.

3.4 Agrupamento usando UIS-RNN

Após a obtenção dos *embeddings* de cada segmento, o último passo envolve descobrir a qual locutor cada segmento pertence e, consequentemente, quantos locutores estão presentes no áudio. Para esta tarefa, o sistema no qual este projeto se baseia propôs o modelo UIS-RNN (ZHANG et al., 2019), substituindo os tradicionais módulos de *clustering* não supervisionados.

Tal método trata-se de um processo generativo o qual utiliza dados anotados para seu treinamento. Sua nomenclatura, rede neural recorrente com *unbounded interleaved-state*, dá-se pelo seguinte:

- Cada locutor é modelado por uma instância RNN, e estas instâncias compartilham os mesmos parâmetros;
- Um número ilimitado de instâncias da rede podem ser gerados;
- Os estados das instâncias RNN, correspondentes a diferentes locutores, se intercalam no domínio do tempo.

Para determinar se um *embedding* pertence a um locutor que já foi registrado ou a um novo, utiliza-se um processo Bayesiano não-paramétrico, conhecido como *Distance-Dependent Chinese Restaurant Process* (ddCRP) ou Processo de Restaurante Chinês Dependente de Distâncias, em tradução livre (BLEI; FRAZIER, 2011). De acordo com o treinamento realizado, o modelo aprende o comportamento dos locutores e das suas interações. Se a conversa é dominada por um locutor, a chance de um segundo ou terceiro locutor ser inserido é baixa, como acontece em conversas telefônicas por exemplo. Mas se existem vários locutores e ocorrem interrupções constantemente (como em reuniões ou entrevistas), a chance de que o segmento pertence a um novo locutor é alta. Todo o método é descrito com maiores detalhes no Anexo A.

Na implementação desenvolvida para este projeto, seguindo o projeto base, foi utilizada a *Gated Recurrent Unit* (GRU) (CHO et al., 2014) como modelo RNN, de modo a memorizar as dependências de longo prazo. O modelo de geração de sequências é composto por uma camada de 512 células GRU com ativação *tanh*, seguida de duas camadas densas com 512 unidades cada e ativação ReLU. Para inferência, foi utilizado *beam search* de comprimento 10.

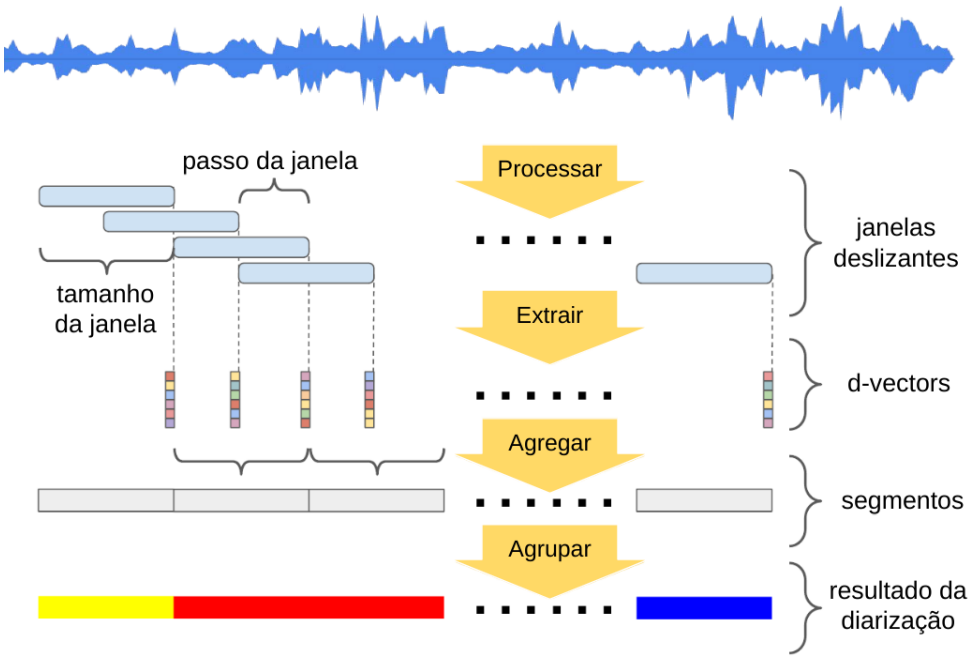
Este módulo foi implementado utilizando o código aberto do repositório disponibilizado por Wang (2021) do grupo que desenvolveu a rede UIS-RNN. Diferentemente do bloco anterior, foi utilizado o *framework PyTorch* para definição, treinamento e avaliação da rede.

É importante ressaltar que o código disponibilizado é uma implementação *open source*, ligeiramente diferente da implementação original da qual foram obtidos os resultados do artigo de divulgação. Isso se dá pois eles utilizam códigos e bibliotecas proprietárias, da infraestrutura interna do *Google*, que não podem ser compartilhadas.

3.5 Resumo do sistema implementado

Por fim, na Figura 2 está representada a integração dos blocos apresentados e implementados, resultando no sistema de diarização de locutor. No topo da figura observa-se a representação do áudio, em amplitude. Este áudio é segmentado em janelas deslizantes, das quais são processadas as informações acústicas de cada trecho. Essas informações são alimentadas na rede LSTM, a qual produz as representações *d-vectors* em sua saída. Esses vetores são agregados em segmentos de tamanho fixo e não sobrepostos, os quais são concatenados e usados como entrada da rede UIS-RNN. Como resultado do processamento da rede, os segmentos são agrupados e classificados e acordo com o locutor, determinando quem falou e quando.

Tabela 2 – Diagrama do sistema de diarização de locutor



Fonte: Wang et al. (2018).

Nota: Tradução pelo próprio autor.

4 RESULTADOS

4.1 Recursos Computacionais

Recursos de *Hardware*: Os recursos computacionais necessários para o desenvolvimento deste trabalho consiste em um servidor com alta capacidade de processamento. O principal servidor utilizado, disponível no laboratório VISIO da Universidade Federal do Espírito Santo, tem como configuração:

- Sistema Operacional Linux, distribuição Ubuntu 16.04.6 LTS;
- 2 processadores Intel(R) Xeon(R) Silver 4214, 2.2GHz, com 12 núcleos cada;
- Memória RAM de 64 GB;
- Unidade de armazenamento SSD de 240GB;
- 3 placas de vídeo Nvidia Titan V;
- 1 placa de vídeo Nvidia GeForce GTX 1080 Ti.

Recursos de *Software*: O código foi escrito usando a linguagem *Python*, e as bibliotecas de aprendizagem de máquina *Tensorflow* e *PyTorch* foram utilizadas para treinar e avaliar os modelos. O ambiente de desenvolvimento foi criado usando *Docker*, de modo que as imagens criadas foram aproveitadas para rodar os programas no ambiente de execução disponibilizado pelo laboratório VISIO. Para processamento de áudio e extração de características, foi utilizada a biblioteca *Librosa*. Por último, os resultados dos experimentos de diarização foram calculado através da biblioteca *pyannote.metrics* (BREDIN, 2017).

4.2 Métricas

Os sistemas de diarização de locutor são comumente avaliados utilizando duas métricas:

- *Diarization Error Rate* - DER (tradução livre, Taxa de Erro de Diarização)
- *Jaccard Error Rate* - JER (tradução livre, Taxa de Erro Jaccard)

A métrica DER é tida como padrão na avaliação e comparação de sistemas de diarização de locutor. Seu cálculo é baseado em quatro componentes distintos, definindo a métrica de acordo com a Equação (4.1).

$$DER = \frac{FA + MD + CONF}{TOTAL}. \tag{4.1}$$

Onde:

- $TOTAL$ representa a duração completa dos trechos contendo fala no arquivo de referência;
- FA (*False Alarm* ou alarme falso) representa a duração de segmentos do áudio que foram incorretamente classificados como fala;
- MD (*Missed Detection* ou detecção perdida) se refere a duração dos trechos imprecisamente classificados com ausência de fala;
- $CONF$ (*CONFusion* ou confusão) trata da duração dos trechos nos quais atribuiu-se inadequadamente o locutor, ou seja, confundido por outro.

É importante notar que esta métrica leva em consideração as falas sobrepostas, de modo que o erro pode aumentar consideravelmente caso o sistema de diarização não inclua um módulo de detecção de falas sobrepostas.

Recentemente a métrica JER foi introduzida como avaliação pelo DIHARD II (RYANT et al., 2019). Esta se baseia no índice Jaccard, o qual é uma medida de similaridade tipicamente utilizada para avaliar a saída de sistemas de segmentação de imagem (YUAN; CHAO; LO, 2017).

No caso, em sistema de diarização, um mapeamento otimizado é feito entre os locutores de referência (*ground truth*) e os produzidos pelo sistema. Para cada par mapeado, calcula-se o índice Jaccard, de similaridade. O JER então é definido como 1 menos a média destes *scores*. Embora se assemelhe à DER, esta métrica difere ao ponderar a contribuição de cada locutor igualmente, independente do volume de fala produzido individualmente.

De maneira prática, assume-se N locutores de referência e M locutores produzidos pelo sistema. Um mapeamento otimizado entre locutores é determinado usando o algoritmo Húngaro, de forma que cada locutor de referência seja associado a no máximo um locutor

do sistema. Para cada locutor do sistema também é associado no máximo um locutor de referência. Então, para cada locutor de referência ref calcula-se o JER_{ref} específico por locutor usando a Equação (4.2).

$$JER_{ref} = \frac{FA + MISS}{TOTAL}. \tag{4.2}$$

Onde:

- $TOTAL$ é a duração da união dos segmentos dos locutor de referência com os do locutor do sistema. Caso o locutor de referência não seja pareado com um locutor do sistema, usa-se a duração de todos os segmentos do locutor de referência;
- FA representa a duração total de falas do locutor do sistema não atribuído ao locutor de referência. Caso o locutor de referência não seja pareado com um locutor do sistema, seu valor é 0;
- $MISS$ representa a duração total de falas do locutor de referência não atribuído ao locutor do sistema. Caso o locutor de referência não seja pareado com um locutor do sistema, seu valor é igual ao $TOTAL$.

Por fim, o *Jaccard Error Rate*, visto na Equação (4.3), é obtido através da média dos JER_{ref}, específicos por locutor:

$$JER = \frac{1}{N} \sum_{ref} JER_{ref} \tag{4.3}$$

As métricas JER e DER são altamente correlatas, com JER tipicamente sendo maior, especialmente em gravações nas quais um ou mais locutores são particularmente dominantes. É importante destacar que o DER pode alcançar valores como 200%, 300% ou até mais, enquanto o JER poder chegar no máximo a 100%.

4.3 Formato do arquivo de avaliação

Para a avaliação, espera-se que o sistema de diarização anote as informações de locutores e intervalos de fala de uma maneira estruturada. O formato do arquivo mais utilizado para tais sistemas é o *Rich Transcription Time Marked* (RTTM). Seus arquivos são de

texto, delimitados por espaços, contendo um turno por linha e cada linha contendo dez campos de informações. Em diarização, nem todos os campos são utilizados, de modo que estes devem ser preenchidos com `<NA>` para serem ignorados no cálculo das métricas. A seguir são apresentados estes campos, em ordem de escrita, e seus preenchimentos para sistemas de diarização:

1. Tipo: deve ser sempre preenchido como `SPEAKER`;
2. Identificação do arquivo: Nome de base da gravação, sem conter a extensão;
3. Identificação do canal: Canal no qual a sequência ocorre. Deve sempre ser 1 (*single channel*);
4. Início do turno: Identificação, em segundos, do início da gravação;
5. Duração do turno: Duração em segundos;
6. Campo de Ortografia: Deve sempre ser `<NA>`;
7. Tipo de locutor: Deve sempre ser `<NA>`;
8. Nome do locutor: Identificação do locutor do turno; deve sempre ser único dentro do escopo de cada arquivo.
9. Pontuação de confiança: Confiabilidade do sistema (probabilidade) de que a informação está correta. Deve sempre ser `<NA>`;
10. Tempo de antecipação do sinal: Deve sempre ser `<NA>`.

4.4 Experimentos

O sistema implementado neste projeto foi avaliado em dois experimentos. Em ambos os experimentos empregou-se o modelo pré-treinado para extração de *embeddings*, alternando apenas o treinamento da rede UIS-RNN e a forma de validar.

O primeiro experimento baseia-se no conjunto de dados *VoxConverse* que consiste de áudios extraídos de vídeos do *YouTube*, contendo debates políticos, entrevistas e alguns outros segmentos. Sua descrição detalhada de suas características no Anexo B. O treinamento do modelo foi feito utilizando seu conjunto de desenvolvimento e a validação é feita usando o conjunto de teste. Este experimento tem por objetivo a análise inicial da rede e o desenvolvimento do seu treinamento.

O segundo e principal experimento é a submissão à faixa de diarização da competição *VoxCeleb Speaker Recognition Challenge* (VoxSRC) 2021. Para treinamento da rede UIS-RNN, foi utilizado todo o conjunto de dados *VoxConverse*, ambos os conjuntos de desenvolvimento e teste que juntos resultam em mais de 60h de conversas.

A avaliação deste experimento foi feita usando o conjunto de teste disponibilizado pela competição. Tal conjunto consiste em 264 arquivos de áudio, de mesmo domínio do *VoxConverse*, contendo trechos de conversas e entrevistas extraídas do *YouTube*, envolvendo múltiplos locutores em cada áudio.

As métricas de validação foram computadas através da submissão do resultado de diarização à plataforma *online* que hospeda a competição. O arquivo é processado pela plataforma e o resultado é reportado aos participantes, produzindo um quadro de melhores soluções. O código utilizado para cálculo das métricas é disponibilizado por Ryant (2019). Só não se tem acesso ao *ground-truth* do conjunto de testes.

Para cada um dos experimentos, também são reportados os resultados obtidos ao substituir a rede UIS-RNN, na etapa de agrupamento, pelo algorítimo *spectral clustering*, comumente utilizado para esta tarefa e considerado estado da arte em 2018.

4.5 Resultados

Na Tabela 3 estão dispostos os resultados dos experimentos. Os valores em negrito, na coluna do DER, são os principais obtidos. Para o primeiro experimento, os valores de interesse são os da parcela de confusão, pois este é valor usado na divulgação dos resultados em Zhang et al. (2019), Wang et al. (2018) e outros mais. Assim como nestes trabalhos, as métricas também foram calculadas desconsiderando trechos de falas sobrepostas, o que trás pouca influência na parcela de confusão. A faixa de teste VoxSRC-21 não possui os componentes do DER discriminados pois estes não são apresentados pela plataforma de submissão do desafio. Logo, seu resultado principal é o DER total. Trechos de falas sobrepostas foram levados em consideração.

Observa-se que os experimentos usando o método UIS-RNN não apresentaram um resultado competitivo considerando o DER e foram um pouco melhores considerando o JER, em comparação com o método *spectral clustering*. Os componentes de detecção perdida e alarme falso permaneceram constantes e seus valores foram relativamente baixos. Já os valores totais das métricas foram muito elevados, sendo a parcela de "confusão" a principal contribuinte. Nota-se também que os resultados dos dois testes são semelhantes,

Tabela 3 – Resultados de diarização obtidos nos experimentos

Teste	**Método**	**Treino**	**DER**(%) Confusão	Detecção Perdida	Alarme Falso	Total	**JER**(%)
VoxConverse Teste	UIS-RNN	*VoxConverse Dev*	**50,5**	2,0	2,4	54,9	63,6
	Spectral Clustering	—	**36,0**	2,0	2,4	40,4	81,5
VoxSRC-21 Teste	UIS-RNN	*VoxConverse Dev* + Teste	—	—	—	**61,3**	57,6
	Spectral Clustering	—	—	—	—	**41,2**	77,2

Fonte: Produção do próprio autor.

Nota: Nas métricas utilizadas, valores menores indicam melhores resultados.

apresentando pouca variação entre si.

4.6 Comparação com resultados do desafio

Na Tabela 4 estão os resultados das melhores submissões ao *VoxCeleb Speaker Recognition Challenge* 2021, divulgados na plataforma *CodaLab* (CODALAB, 2021). Foram listadas aqui apenas as equipes que publicaram um relatório técnico no *workshop* da competição, detalhando seu sistema submetido (UNIVERSITY OF OXFORD, 2021). O resultado da equipe de *Oxford*, organizadores do desafio, também foi incluído aqui pois seu trabalho é considerado como referencia da competição. Apesar de não terem submetido um relatório técnico, sabe-se que eles utilizaram um sistema com *py-webrtcvad*, extração de *embeddings* usando o modelo ResNetSE34 (CHUNG et al., 2020a) e *spectral clustering*.

Em relação às abordagens empregadas nas soluções apresentadas, pode-se dizer que não existe um consenso sobre o melhor sistema. Apesar de todas as propostas ter a mesma estrutura base de um sistema de diarização, cada equipe utilizou e combinou blocos diferentes, contendo múltiplas técnicas. Entretanto, mesmo diante de muitas combinações diferentes, alguns abordagens base foram empregadas por quase todas as equipes. Um exemplo é a utilização de modelos DNN e RNN, presentes em todas as soluções, especialmente nos blocos de extração de *embeddings* e VAD. Os modelos mais utilizados na etapa de extração foram os baseados na ECAPA-TDNN (DESPLANQUES; THIENPONDT; DEMUYNCK, 2020) e na ResNet (HE et al., 2016), por vezes até combinados.

Para a tarefa de *clustering*, a maioria das soluções basearam-se no método AHC de modo que muitos deles aplicavam esta etapa em mais de uma ocasião (*pre-clustering*,

Tabela 4 – *Ranking* geral da faixa 4 (diarização) do desafio VoxSRC 2021

Posição	**Equipe**	**DER**(%)	**JER**(%)
1	DKU-DukeECE-Lenovo	5, 0726	29, 1640
2	ByteDance	5, 1455	26, 0231
3	Tencent	5, 3162	24, 5044
...	...	...	...
6	HUAWEI	5, 5434	25, 3164
7	XMUSPEECH	5, 5452	27, 1181
8	Sogou-inc	5, 8145	27, 9022
...	...	...	...
15	SamsungAmerica	6, 1673	27, 9596
...	...	...	...
25	Oxford (*baseline*)	17, 9946	38, 7221
...	...	...	...
28	Este projeto	41, 1762	77, 2213
...	...	...	...

Fonte: CodaLab (2021).

re-clustering). Também foram combinados mais de um algoritmo de *clustering*. O *spectral clustering* foi utilizado pelas melhores solução, como foi o caso do time DKU que o combinou com AHC. Nota-se que nenhuma das submissões utilizou o modelo UIS-RNN em sua fase de agrupamento.

Um ponto a ser destacado é que todos os melhores resultados utilizaram blocos de detecção de falas sobrepostas ou OSD (*overlapped speech detection*). A equipe Tencent, por exemplo utilizou um bloco OSD logo na entrada do sistema, após o VAD e outro bloco ao final, após a etapa de *clustering*, para realizar uma re-segmentação ciente de falas sobrepostas (*overlap-aware resegmentation*). Já a equipe ByteDance utilizou tanto o OSD quanto o VAD do *pyannote* 2.0 desenvolvido por Bredin e Laurent (2021).

Outro ponto interessante dos sistemas apresentados é que todos usaram fusões de sistemas. Basicamente foram desenvolvidos alguns *pipelines* usando abordagens diferentes, mas com blocos semelhantes de diarização, que podem variar em dados de treinamento também. Cada sistema é treinado e avaliado separadamente. Ao final, os melhores são selecionados e tem seus componentes fundidos. Tal fusão pode ocorrer a nível de bloco, onde são montados *ensembles* dos modelos, ou então cada sistema produz uma resposta e ao final estas são combinadas, com o objetivo de produzir o melhor resultado. Para este último formato, diferentes competidores usaram o algoritmo DOVER (STOLCKE; YOSHIOKA, 2019) e também sua evolução DOVER-Lap (RAJ et al., 2021), que agrega a detecção de falas sobrepostas.

4.7 Análise de resultados

Os resultados reportados na Tabela 3 são globais e referentes a todo conjunto utilizado como teste. Entretanto a análise dos resultados individuais de determinados áudios auxilia na compreensão do desempenho do sistema implementado. Para ilustrar estes resultados, foram escolhidos três áudios do conjunto *VoxConverse Test*, de diferentes durações. Para cada um destes será apresentada uma tabela com as métricas obtidas, uma figura contendo a visualização da sua resposta de diarização e por fim uma figura contendo a representação dos *embeddings* e suas anotações de locutor produzidas pelo respectivo sistema.

O primeiro áudio a ser analisado é o "fuzfh", o qual possui $26,04$ segundos de duração e 3 locutores no total. Este áudio é o de menor duração do conjunto de testes e possui menos da metade da média de locutores do restante do conjunto. As métricas obtidas ao processar esse áudio estão representados na Tabela 5. Nota-se que o método UIS-RNN obteve melhor resultado tanto em DER como em JER, com apenas $2,3\%$ de confusão. Como o método *spectral clustering* falhou em detectar o terceiro locutor, seu resultado em JER foi de $39,4\%$, quase três vezes maior que o JER obtido com o método UIS-RNN. Outro ponto a ser observado é o componente de alarme falso o qual obteve resultado nulo, mostrando que para este caso não foram detectados trechos de fala onde não havia. Entretanto, o sistema deixou de detectar $5,1\%$ das falas, como mostra o componente de detecção perdida.

Tabela 5 – Resultados de diarização do áudio "fuzfh"

Áudio	Método	Locutores Preditos	DER (%) Confusão	Detecção Perdida	Alarme Falso	Total	JER (%)
VoxConverse Teste "fuzfh"	UIS-RNN	4	**2,3**	5,1	0,0	7,4	**13,4**
	Spectral Clustering	2	**5,4**	5,1	0,0	10,6	**39,4**

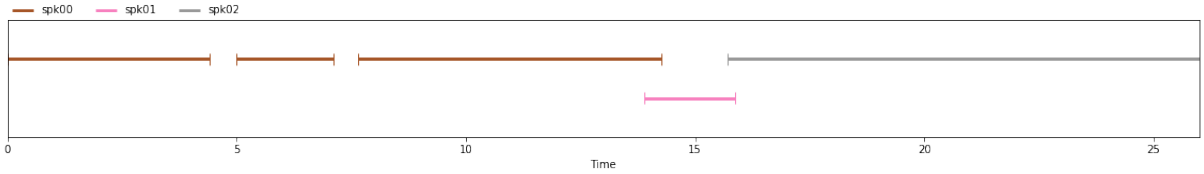
Fonte: Produção do próprio autor.

Nota: $26,04$ segundos de duração e 3 locutores no total.

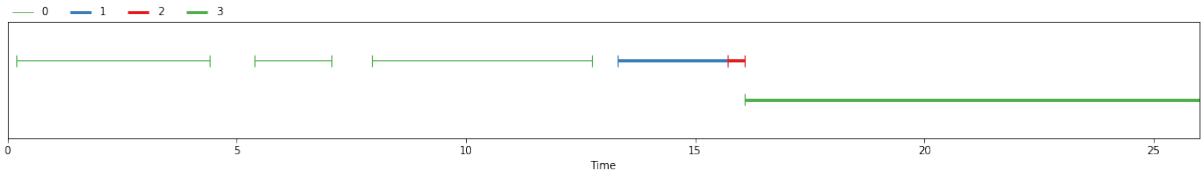
Na Figura 10 estão representadas as visualizações dos resultados de diarização do áudio "fuzfh". A Figura 10a ilustra o resultado de referência, ou *ground truth*, para este áudio. Já as Figuras 10b e 10c representam os resultados obtidos usando os métodos UIS-RNN e *spectral clustering* respectivamente.

Figura 10 – Visualização dos resultados de diarização do áudio "fuzfh"

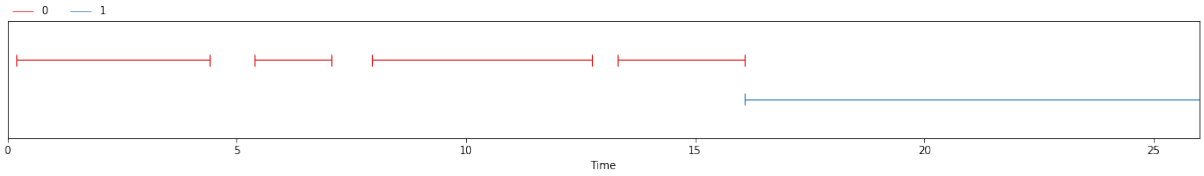
(a) Referência



(b) UIS-RNN



(c) *Spectral Clustering*

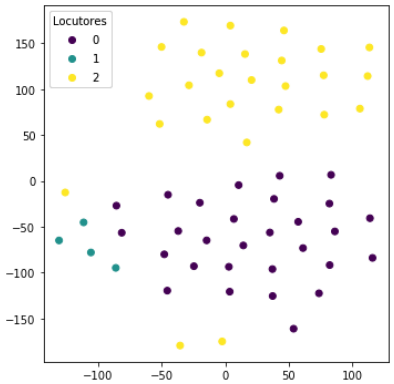


Fonte: Produção do próprio autor.

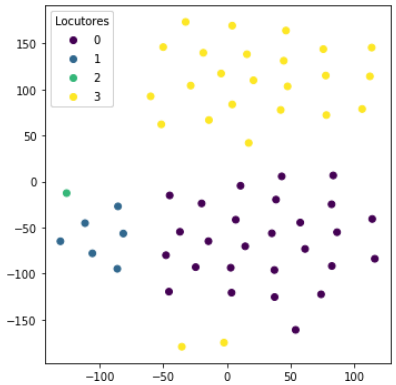
Por fim, na Figura 11 estão representados os *embeddings* de cada segmento do áudio "fuzfh", obtidos pelo sistema implementado neste trabalho. As Figuras 11a, 11b e 11c ilustram tais *embeddings* rotulados de acordo com a referência e com os resultados obtidos dos métodos UIS-RNN e *spectral clustering* respectivamente.

Figura 11 – Representação dos *embeddings* classificados do áudio "fuzfh"

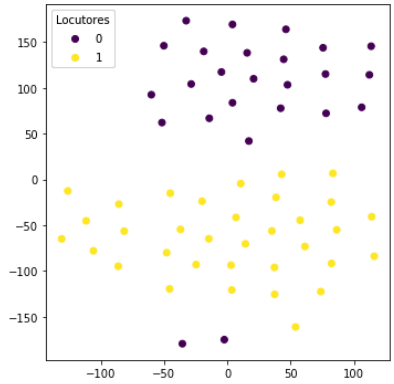
(a) Referência



(b) UIS-RNN



(c) *Spectral Clustering*



Fonte: Produção do próprio autor.

Em seguida, serão analisados os resultados de diarização obtidos do áudio "dzsef", o qual possui 164, 24 segundos de duração e 3 locutores no total. Sua duração, apesar de ser maior que a do áudio "fuzfh", ainda é considerada de tamanho curto já que a média de duração do conjunto de teste é de 675, 6 segundos, como pode ser visto na Tabela 9. As métricas obtidas ao processar esse áudio estão representados na Tabela 6. Para este áudio, o método *spectral clustering* obteve melhor resultado na métrica DER, por uma curta diferença de aproximadamente 3 pontos percentuais. Entretanto o método UIS-RNN obteve melhor resultado na métrica JER, pois novamente o método *spectral clustering* falhou em classificar um locutor o que tem grande impacto no JER. Observa-se também que os componentes de detecção perdida e alarme falso tiveram resultados quase nulos, demonstrando bom funcionamento do VAD.

Tabela 6 – Resultados de diarização do áudio "dzsef".

Áudio	Método	Locutores Preditos	DER (%) Confusão	Detecção Perdida	Alarme Falso	Total	JER (%)
VoxConverse Teste "dzsef"	UIS-RNN	6	**13,5**	0,1	0,7	14,3	**19,7**
	Spectral Clustering	2	**10,3**	0,1	0,7	11,1	**45,7**

Fonte: Produção do próprio autor.

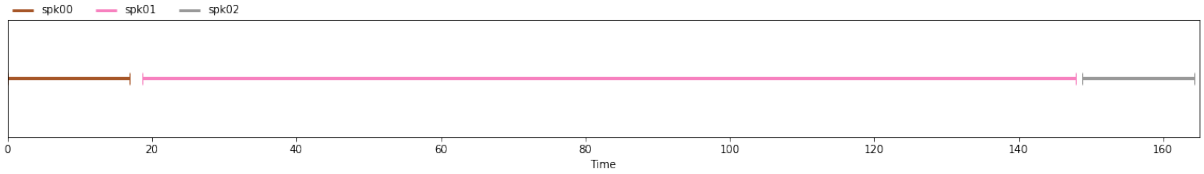
Nota: 164, 24 segundos de duração e 3 locutores no total.

Na Figura 12 estão representadas as visualizações dos resultados de diarização do áudio "dzsef". A Figura 12a ilustra o resultado de referência, ou *ground truth*, para este áudio. Já as Figuras 12b e 12c representam os resultados obtidos usando os métodos UIS-RNN e *spectral clustering* respectivamente. É possível observar que, apesar do elevado número de locutores detectados, o método UIS-RNN ainda acerta boa parte do áudio, de modo que os locutores extras tem duração reduzida com pouco impacto no resultado.

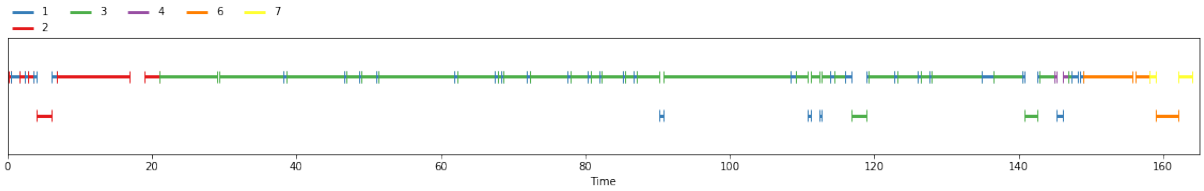
Finalmente, na Figura 13 estão representados os *embeddings* de cada segmento do áudio "dzsef". As Figuras 13a, 13b e 13c ilustram tais *embeddings* rotulados de acordo com a referência e com os resultados obtidos dos métodos UIS-RNN e *spectral clustering* respectivamente. Como o áudio analisado é de duração maior que o analisado anteriormente, nota-se que as figuras possuem mais *embeddings* representados. Portanto, pode-se dizer que quanto mais segmentos mais difícil se torna a tarefa de *clustering*, principalmente se tais *embeddings* não forem suficientemente discriminativos.

Figura 12 – Visualização dos resultados de diarização do áudio "dzsef"

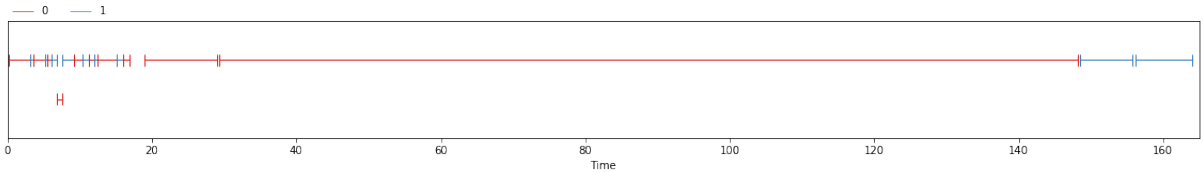
(a) Referência



(b) UIS-RNN



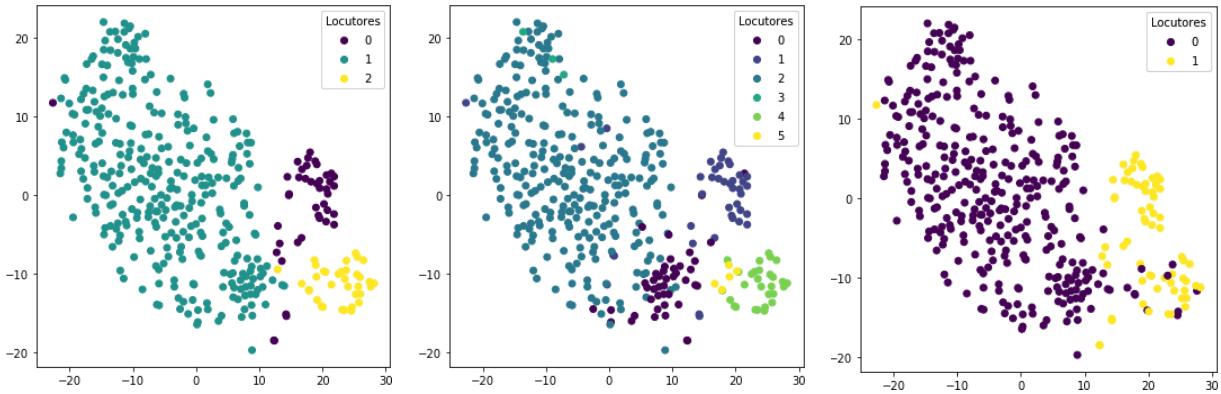
(c) *Spectral Clustering*



Fonte: Produção do próprio autor.

Figura 13 – Representação dos *embeddings* classificados do áudio "dzsef"

(a) Referência (b) UIS-RNN (c) *Spectral Clustering*



Fonte: Produção do próprio autor.

Por último, são analisados os resultados de diarização obtidos do áudio "epygx", um dos maiores do conjunto de testes com 1032 segundos de duração e 8 locutores no total. Apesar da grande duração, seu número de locutores é reduzido. O maior número de locutores presentes em um áudio no conjunto de testes é 21, como pode ser visto na Tabela 9. As métricas obtidas ao processar esse áudio estão representados na Tabela 7. Para este áudio, novamente o método *spectral clustering* obteve melhor resultado na métrica DER, com uma diferença de aproximadamente 23 pontos percentuais, muito superior aos outros

áudios analisados. E assim como nos anteriores, o resultado em JER do método *spectral clustering* foi pior pois este falhou em detectar um grande número de locutores. Entretanto o método UIS-RNN também obteve JER alto pois este detectou erroneamente um número muito superior de locutores, mais que o dobro da referência. Novamente observam-se bons resultados dos componentes de alarme false e detecção perdida, as quais não são fortemente impactadas pelo aumento na duração do áudio, assim como os outros componentes.

Tabela 7 – Resultados de diarização do áudio "epygx"

Áudio	Método	Locutores Preditos	DER (%) Confusão	Detecção Perdida	Alarme Falso	Total	JER (%)
VoxConverse Teste "epygx"	UIS-RNN	17	**50,4**	1,6	2,8	54,8	**62,5**
	Spectral Clustering	3	**27,2**	1,6	2,8	31,6	**82,9**

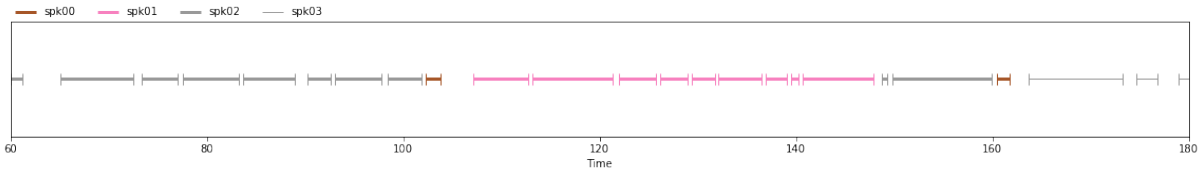
Fonte: Produção do próprio autor.

Nota: 1032 segundos de duração e 8 locutores no total.

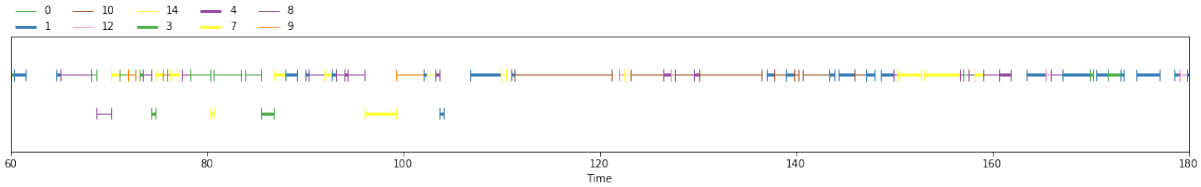
Na Figura 14 pode-se observar os resultados de diarização do áudio "epygx". A Figura 14a ilustra o resultado de referência, para este áudio. Já as Figuras 14b e 14c representam os resultados obtidos usando os métodos UIS-RNN e *spectral clustering* respectivamente.

Figura 14 – Visualização dos resultados de diarização do áudio "epygx"

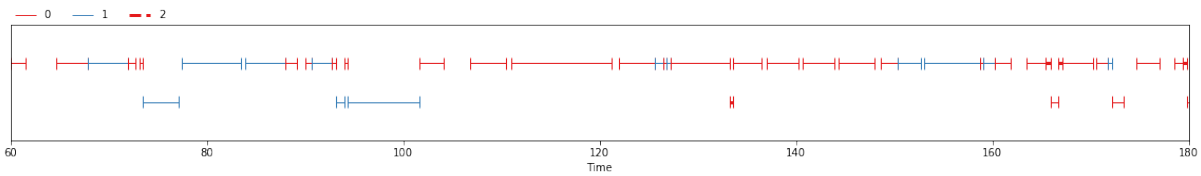
(a) Referência



(b) UIS-RNN



(c) *Spectral Clustering*



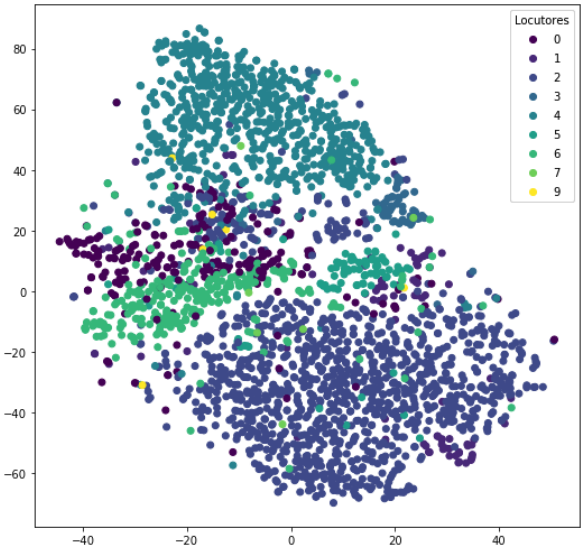
Fonte: Produção do próprio autor.

Como este áudio é bem longo, está representado apenas um de serus intervalos, de 60 a 180 segundos. Neste intervalo já é possível observar como o UIS-RNN obteve muitos locutores avulsos e com uma alternância muito grande entre eles, prejudicando tanto o resultado em DER como em JER.

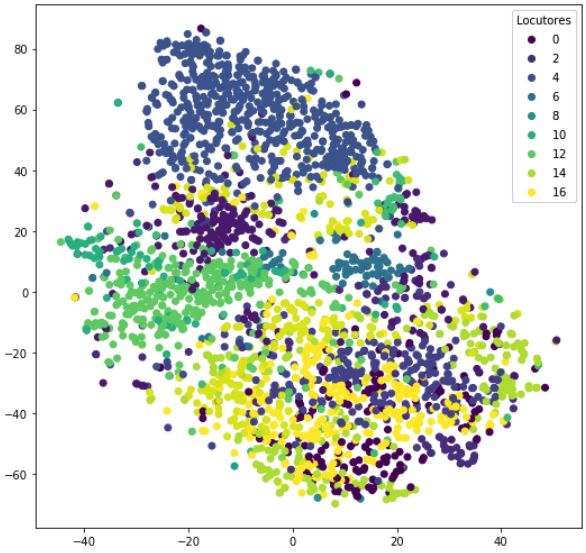
E por fim, na Figura 15 estão representados os *embeddings* de cada segmento do áudio "epygx", separados pelas respectivas classificações. Como sua duração é a maior de todos os áudios analisados, nota-se que suas figuras possuem muito mais *embeddings* representados. Logo a tarefa de *clustering* se torna ainda mais difícil, e pode-se notar que que os *embeddings* não são diferentes entre si o suficiente para poderem ser apropriadamente classificados.

Figura 15 – Representação dos *embeddings* classificados do áudio "epygx"

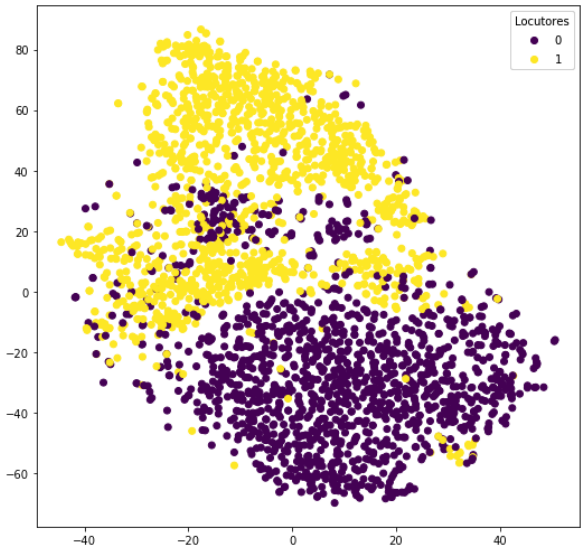
(a) Referência



(b) UIS-RNN



(c) *Spectral Clustering*



Fonte: Produção do próprio autor.

4.8 Pesquisa reproduzível

Para que esta pesquisa possa ser posteriormente reproduzida, todos os códigos relevantes para realização dos experimentos estão disponíveis em <https://github.com/scalfs/voxsrc21-dia>.

5 CONCLUSÕES E PROJETOS FUTUROS

5.1 Conclusões

O objetivo principal deste trabalho foi implementar um sistema de diarização de locutor baseado no modelo considerado estado da arte e validar sua implementação na competição VoxSRC-21. No caso, foi escolhido como base o sistema de diarização que combina uma LSTM para extração de *embeddings* e a UIS-RNN para agrupar os seus respectivos locutores. O sistema completo inclui um VAD usando GMM e utiliza segmentos fixos de áudio durante o processo de diarização.

A metodologia proposta baseia-se na utilização de redes neurais recorrentes com o objetivo de processar e aprender melhor os dados de áudio e de fala. E como o sistema completo envolve modelos treináveis e supervisionados, em tese é possível adequá-lo para variadas aplicações em diferentes domínios. A possibilidade de utilizar o sistema de modo *online*, permitindo realizar a inferência em *streams* de áudio também amplia suas aplicações.

O sistema implementado foi então avaliado na competição, na qual foram obtidos DER $61,3\%$ e JER $57,6\%$. Para referência, o ganhador do desafio obteve DER $5,07\%$ e JER $29,2\%$ e o sistema considerado *baseline* obteve DER $17,9\%$ e JER $38,7\%$.

Para entender melhor os resultados, também foram realizados experimentos substituindo a UIS-RNN pelo seu método antecessor, *spectral clustering*. Com este sistema foram obtidos DER $41,2\%$ e JER $77,2\%$, o que também configurou um resultado bem abaixo do esperado.

A partir do experimentos no conjunto de teste, foi possível observar melhor a composição da métrica DER. Em ambos métodos testados, foram obtidos $2,0\%$ e $2,4\%$ nos componentes de detecção perdida e alarme falso respectivamente. Resultados relativamente baixos e portanto bons, demonstrando que os blocos de segmentação e VAD, responsável por detectar trechos de fala, estão funcionando corretamente.

Observa-se que os experimentos com o método UIS-RNN foram melhores pela métrica secundária JER. Como essa métrica demonstra a quantidade de erros em atribuição de locutores sem considerar sua contribuição temporal, pode-se dizer que este método teve um sucesso maior em identificar locutores.

Entretanto, como essa vantagem é perdida segundo os resultados em DER, conclui-se que o método UIS-RNN teve pior desempenho agravado por conta do tamanho dos áudios. Ao analisar os arquivos RTTM de saída deste sistema, constata-se que a rede gera um número de locutores muito superior ao esperado. Quanto maior o áudio, mais locutores são gerados, aumentando o erro por confusão de locutor. Já as saídas produzidas pelo método *spectral clustering* indicavam que esta gerava muito menos locutores do que o efetivamente correto, o que explica seu erro JER tão elevado.

Devido aos resultados ilustrados, acredita-se que a UIS-RNN não tenha sido treinada de maneira adequada e nem teve seus hiper-parâmetros devidamente ajustados. O *spectral clustering* também pode não ter sido configurado corretamente.

Como ambos métodos de *clustering* obtiveram tais resultados, pode-se inferir também que o bloco de extração de *embeddings* não tenha conseguido produzir representação dos locutores discriminativas o suficiente para serem devidamente classificadas. Pelo fato de ter sido usada uma rede pré treinada, é necessário investigar melhor a qualidade dos *embeddings* produzidos por esta rede e optar por um novo modelo caso necessário.

É importante salientar que, como visto na Tabela 9, o conjunto de avaliação possui diversas ocorrências de falas sobrepostas. Devido ao sistema aqui implementado não tratar essas ocorrências, já era esperado que o resultado sofresse influência. Portanto sabe-se que o sistema pode melhorar, mas não é o mais indicado para lidar com este domínio de áudios.

5.2 Temas a serem pesquisados

Como trabalhos futuros, além de buscar solucionar os problemas desta implementação, recomenda-se dois caminhos a serem seguidos. Caso se tenha interesse em obter melhores resultados no conjunto de dados testado neste trabalho, recomenda-se o estudo e implementação de abordagens mais adequadas ao problema. É muito importante adicionar no sistema um método para tratar das falas sobrepostas e também componentes de pós processamento. Outros pontos que podem trazer grande melhoria seria a experimentação de outros modelos para extração de *embeddings* e implementação de estratégias mais completas de *clustering*. Como ponto de partida se tem os sistemas que ficaram nas melhores colocações do desafio VoxSRC-21.

O segundo caminho seria seguir investindo na melhoria do sistema aqui implementado, mas aplicá-lo em domínios de áudio com menos falantes e mais controlado. Pois foi nesse tipo de áudio que o sistema base foi desenvolvido e obteve seus melhores resultados.

REFERÊNCIAS

BLEI, D. M.; FRAZIER, P. I. Distance dependent chinese restaurant processes. Journal of Machine Learning Research, v. 12, n. 8, 2011. Citado 3 vezes nas páginas 29, 54 e 56.

BREDIN, H. pyannote. metrics: A toolkit for reproducible evaluation, diagnostic, and error analysis of speaker diarization systems. In: INTERSPEECH. [S.l.: s.n.], 2017. p. 3587–3591. Citado na página 32.

BREDIN, H.; LAURENT, A. End-to-end speaker segmentation for overlap-aware resegmentation. arXiv preprint arXiv:2104.04045, 2021. Citado na página 38.

BREDIN, H.; YIN, R.; CORIA, J. M.; GELLY, G.; KORSHUNOV, P.; LAVECHIN, M.; FUSTES, D.; TITEUX, H.; BOUAZIZ, W.; GILL, M.-P. Pyannote. audio: neural building blocks for speaker diarization. In: IEEE. ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2020. p. 7124–7128. Citado 2 vezes nas páginas 17 e 19.

CHEN, Z.; YOSHIOKA, T.; LU, L.; ZHOU, T.; MENG, Z.; LUO, Y.; WU, J.; XIAO, X.; LI, J. Continuous speech separation: Dataset and analysis. In: IEEE. ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2020. p. 7284–7288. Citado na página 25.

CHO, K.; MERRIËNBOER, B. V.; GULCEHRE, C.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014. Citado 2 vezes nas páginas 30 e 54.

CHUNG, J. S.; HUH, J.; MUN, S.; LEE, M.; HEO, H. S.; CHOE, S.; HAM, C.; JUNG, S.; LEE, B.-J.; HAN, I. In defence of metric learning for speaker recognition. In: Interspeech. [S.l.: s.n.], 2020. Citado na página 37.

CHUNG, J. S.; HUH, J.; NAGRANI, A.; AFOURAS, T.; ZISSERMAN, A. Spot the Conversation: Speaker Diarisation in the Wild. In: Proc. Interspeech 2020. [S.l.: s.n.], 2020. p. 299–303. Citado na página 59.

CHUNG, J. S.; NAGRANI, A.; ZISSERMAN, A. VoxCeleb2: Deep Speaker Recognition. In: Proc. Interspeech 2018. [S.l.: s.n.], 2018. p. 1086–1090. Citado na página 58.

CODALAB. The VoxCeleb Speaker Recognition Challenge 2021 - Track 4 (Diarisation, open). 2021. Disponível em: <https://competitions.codalab.org/competitions/34113#results>. Acesso em: 27 set. 2021. Citado 2 vezes nas páginas 37 e 38.

DEHAK, N.; KENNY, P. J.; DEHAK, R.; DUMOUCHEL, P.; OUELLET, P. Front-end factor analysis for speaker verification. IEEE Transactions on Audio, Speech, and Language Processing, IEEE, v. 19, n. 4, p. 788–798, 2010. Citado na página 18.

DELACOURT, P.; KRYZE, D.; WELLEKENS, C. J. Speaker-based segmentation for audio data indexing. In: ESCA Tutorial and Research Workshop (ETRW) on Accessing Information in Spoken Audio. [S.l.: s.n.], 1999. Citado na página 18.

DESPLANQUES, B.; THIENPONDT, J.; DEMUYNCK, K. ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification. In: Proc. Interspeech 2020. [S.l.: s.n.], 2020. p. 3830–3834. Citado na página 37.

DRUGMAN, T.; STYLIANOU, Y.; KIDA, Y.; AKAMINE, M. Voice activity detection: Merging source and filter-based information. IEEE Signal Processing Letters, IEEE, v. 23, n. 2, p. 252–256, 2015. Citado na página 17.

HAN, K. J.; NARAYANAN, S. S. A robust stopping criterion for agglomerative hierarchical clustering in a speaker diarization system. In: CITESEER. Interspeech. [S.l.], 2007. p. 1853–1856. Citado na página 21.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. p. 770–778. Citado na página 37.

KENNY, P.; REYNOLDS, D.; CASTALDO, F. Diarization of telephone conversations using factor analysis. IEEE Journal of Selected Topics in Signal Processing, IEEE, v. 4, n. 6, p. 1059–1070, 2010. Citado na página 25.

LIKAS, A.; VLASSIS, N.; VERBEEK, J. J. The global k-means clustering algorithm. Pattern recognition, Elsevier, v. 36, n. 2, p. 451–461, 2003. Citado na página 21.

LOBO, D. Diarization Experiments. 2019. Disponível em: <https://github.com/dalonlobo/diarization-experiments>. Acesso em: 27 set. 2021. Citado na página 28.

LU, X.; TSAO, Y.; MATSUDA, S.; HORI, C. Speech enhancement based on deep denoising autoencoder. In: Interspeech. [S.l.: s.n.], 2013. v. 2013, p. 436–440. Citado na página 25.

MANSFIELD, P. A.; WANG, Q.; DOWNEY, C.; WAN, L.; MORENO, I. L. Links: A High-Dimensional Online Clustering Method. 2018. Citado na página 22.

MORITA, S.; LU, X.; UNOKI, M. Signal to noise ratio estimation based on an optimal design of subband voice activity detection. In: The 9th International Symposium on Chinese Spoken Language Processing. [S.l.: s.n.], 2014. p. 560–564. Citado na página 17.

NAGRANI, A.; CHUNG, J. S.; XIE, W.; ZISSERMAN, A. Voxceleb: Large-scale speaker verification in the wild. Computer Speech & Language, Elsevier, v. 60, p. 101027, 2020. Citado na página 58.

NAKATANI, T.; YOSHIOKA, T.; KINOSHITA, K.; MIYOSHI, M.; JUANG, B.-H. Speech dereverberation based on variance-normalized delayed linear prediction. IEEE Transactions on Audio, Speech, and Language Processing, IEEE, v. 18, n. 7, p. 1717–1731, 2010. Citado na página 25.

NG, T.; ZHANG, B.; NGUYEN, L.; MATSOUKAS, S.; ZHOU, X.; MESGARANI, N.; VESELỲ, K.; MATĚJKA, P. Developing a speech activity detection system for the darpa rats program. In: Thirteenth annual conference of the international speech communication association. [S.l.: s.n.], 2012. Citado na página 17.

PANAYOTOV, V.; CHEN, G.; POVEY, D.; KHUDANPUR, S. Librispeech: an asr corpus based on public domain audio books. In: IEEE. 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP). [S.l.], 2015. p. 5206–5210. Citado 2 vezes nas páginas 59 e 60.

PARK, T. J.; KANDA, N.; DIMITRIADIS, D.; HAN, K. J.; WATANABE, S.; NARAYANAN, S. A review of speaker diarization: Recent advances with deep learning. arXiv preprint arXiv:2101.09624, 2021. Citado 3 vezes nas páginas 12, 16 e 18.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, v. 12, p. 2825–2830, 2011. Citado na página 21.

PRZYBOCKI, A. M. M. The 2000 NIST speaker recognition evaluation. Philadelphia, PA: Produzido por Lingustic Data Consortium, 2001. ISBN 1-58563-192-2. Citado na página 13.

RAJ, D.; GARCIA-PERERA, L. P.; HUANG, Z.; WATANABE, S.; POVEY, D.; STOLCKE, A.; KHUDANPUR, S. Dover-lap: A method for combining overlap-aware diarization outputs. In: IEEE. 2021 IEEE Spoken Language Technology Workshop (SLT). [S.l.], 2021. p. 881–888. Citado na página 38.

RYANT, N. Diarization scoring tools. 2019. Disponível em: <https://github.com/nryant/dscore>. Acesso em: 27 set. 2021. Citado na página 36.

RYANT, N.; CHURCH, K.; CIERI, C.; CRISTIA, A.; DU, J.; GANAPATHY, S.; LIBERMAN, M. Second dihard challenge evaluation plan. Linguistic Data Consortium, Tech. Rep, 2019. Citado na página 33.

SARIKAYA, R.; HANSEN, J. H. Robust detection of speech activity in the presence of noise. In: CITESEER. Proc. ICSLP. [S.l.], 1998. v. 4, p. 1455–8. Citado na página 17.

SENOUSSAOUI, M.; KENNY, P.; STAFYLAKIS, T.; DUMOUCHEL, P. A study of the cosine distance-based mean shift for telephone speech diarization. IEEE/ACM Transactions on Audio, Speech, and Language Processing, IEEE, v. 22, n. 1, p. 217–227, 2013. Citado na página 21.

SHAFEY, L. E.; SOLTAU, H.; SHAFRAN, I. Joint speech recognition and speaker diarization via sequence transduction. arXiv preprint arXiv:1907.05337, 2019. Citado na página 12.

STOLCKE, A.; YOSHIOKA, T. Dover: A method for combining diarization outputs. In: IEEE. 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). [S.l.], 2019. p. 757–763. Citado 2 vezes nas páginas 25 e 38.

UNIVERSITY OF OXFORD. The VoxSRC Workshop 2021. 2021. Disponível em: <https://www.robots.ox.ac.uk/~vgg/data/voxceleb/interspeech2021.html>. Acesso em: 27 set. 2021. Citado na página 37.

VARIANI, E.; LEI, X.; MCDERMOTT, E.; MORENO, I. L.; GONZALEZ-DOMINGUEZ, J. Deep neural networks for small footprint text-dependent speaker verification. In: IEEE. 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). [S.l.], 2014. p. 4052–4056. Citado 3 vezes nas páginas 18, 19 e 20.

VOLEK, H. PyTorch Speaker Verification. 2020. Disponível em: <https://github.com/HarryVolek/PyTorch_Speaker_Verification>. Acesso em: 27 set. 2021. Citado na página 27.

WAN, L.; WANG, Q.; PAPIR, A.; MORENO, I. L. Generalized end-to-end loss for speaker verification. In: IEEE. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2018. p. 4879–4883. Citado 3 vezes nas páginas 20, 27 e 28.

WANG, H.; XU, Y.; LI, M. Study on the mfcc similarity-based voice activity detection algorithm. In: IEEE. 2011 2nd International Conference on Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC). [S.l.], 2011. p. 4391–4394. Citado na página 17.

WANG, Q. UIS-RNN. 2021. Disponível em: <https://github.com/google/uis-rnn>. Acesso em: 27 set. 2021. Citado na página 30.

WANG, Q.; DOWNEY, C.; WAN, L.; MANSFIELD, P. A.; MORENO, I. L. Speaker diarization with lstm. In: IEEE. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2018. p. 5239–5243. Citado 8 vezes nas páginas 13, 21, 22, 23, 27, 31, 36 e 53.

YAMAGISHI, J.; VEAUX, C.; MACDONALD, K. et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92). University of Edinburgh. The Centre for Speech Technology Research (CSTR), 2019. Citado na página 59.

YUAN, Y.; CHAO, M.; LO, Y.-C. Automatic skin lesion segmentation using deep fully convolutional networks with jaccard distance. IEEE transactions on medical imaging, IEEE, v. 36, n. 9, p. 1876–1886, 2017. Citado 2 vezes nas páginas 14 e 33.

ZHANG, A.; WANG, Q.; ZHU, Z.; PAISLEY, J.; WANG, C. Fully supervised speaker diarization. In: IEEE. ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.], 2019. p. 6301–6305. Citado 7 vezes nas páginas 13, 26, 29, 36, 53, 54 e 57.

Anexos

ANEXO A – UIS-RNN

O modelo chamado *Unbounded Interleaved-State Recurrent Neural Network* (UIS-RNN) foi proposto por Zhang et al. (2019) como melhoria para substituir o processo de *clustering* anteriormente aprimorado por Wang et al. (2018). Em um sistema de diarização, este bloco é responsável por determinar, a partir de uma sequência de *embeddings*, quais são os locutores representados em cada segmento. Portanto, sua entrada é uma sequência de *embeddings* $\mathbf{X} = (\mathbf{x}_t \in \mathbb{R}^d | t = 1, \ldots, T)$. Cada elemento desta sequência corresponde a um segmento na utêrancia da qual os *embeddings* foram extraídos. Em um cenário de diarização de locutor supervisionada, também tem-se o *ground truth* $\mathbf{Y} = (y_t \in \mathbb{N} | t = 1, \ldots, T)$ como a sequência de rótulos dos locutores para cada segmento.

A probabilidade conjunta de $\mathbf{X}$ e $\mathbf{Y}$ pode ser decomposta de acordo com a Equação (A.1).

$$P(\mathbf{X}, \mathbf{Y}) = P(\mathbf{x}_1, y_1) \cdot \prod_{t=2}^{T} P(\mathbf{x}_t, y_t | \mathbf{x}_{1:t-1}, y_{1:t-1}) \tag{A.1}$$

Para modelar a distribuição de mudança de locutor, é introduzida a variável latente $\mathbf{Z} = (z_t \in 0, 1 | t = 2, \ldots, T)$, onde z_t torna-se 1 caso os rótulos do locutor nos tempos $t-1$ e t sejam diferentes, e 0 caso permaneçam o mesmo. Por exemplo, caso $\mathbf{Y} = (1, 1, 2, 3, 2, 2)$, então $\mathbf{Z} = (0, 1, 1, 1, 0)$. A probabilidade conjunta adicionando $\mathbf{Z}$ é representada na Equação (A.2).

$$P(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) = P(\mathbf{x}_1, y_1) \cdot \prod_{t=2}^{T} P(\mathbf{x}_t, y_t, z_t | \mathbf{x}_{1:t-1}, y_{1:t-1}, z_{1:t-1}) \tag{A.2}$$

Finalmente, o termo $P(\mathbf{x}_t, y_t, z_t | \mathbf{x}_{1:t-1}, y_{1:t-1}, z_{1:t-1})$ pode ser decomposto em 3 componentes, como visto na Equação (A.3).

$$P(\mathbf{x}_t, y_t, z_t | \mathbf{x}_{1:t-1}, y_{1:t-1}, z_{1:t-1}) = \underbrace{P(\mathbf{x}_t | \mathbf{x}_{1:t-1}, y_{1:t})}_{\text{geração de sequência}} \cdot \underbrace{P(y_t | z_t, y_{1:t-1})}_{\text{atribuição de locutor}} \cdot \underbrace{P(z_t | z_{1:t-1})}_{\text{mudança de locutor}} \tag{A.3}$$

Cada um dos componentes representa uma funcionalidade do modelo completo. No caso, $P(\mathbf{x}_t | \mathbf{x}_{1:t-1}, y_{1:t})$ representa a probabilidade de geração de sequência, a qual é modelada por

uma RNN baseada em uma *Gated Recurring Unit* (GRU) (CHO et al., 2014). $P(y_t|z_t, y_{1:t-1})$ representa a probabilidade de atribuição de locutor, a qual é modelada pelo processo de restaurante chinês dependente de distância (ddCRP) (BLEI; FRAZIER, 2011), capaz de modelar a distribuição de um número ilimitado de locutores. Por último, $P(z_t|z_{1:t-1})$ representa a probabilidade de mudança de locutor, modelada pela distribuição de Bernoulli. Nas seções seguintes, cada um destes componentes é explicado em maior detalhe.

A.1 Mudança de locutor

Assume-se que a probabilidade de $z_t \in 0, 1$ segue a Equação (A.4), onde $g_{\boldsymbol{\lambda}}(\cdot)$ é uma função parametrizada por $\boldsymbol{\lambda}$.

$$P(z_t = 0|z_{1:t-1}, \boldsymbol{\lambda}) = g_{\boldsymbol{\lambda}}(z_{1:t-1}) \tag{A.4}$$

Como z_t indica mudança de locutor no tempo t, tem-se a Equação (A.4).

$$P(y_t = y_{t-1}|z_t, y_{1:t-1}) = 1 - z_t \tag{A.5}$$

De forma geral, $g_{\boldsymbol{\lambda}}(\cdot)$ poderia ser qualquer função, com uma RNN por exemplo. Entretanto, para simplificar, no projeto base proposto por Zhang et al. (2019), este é definido como um valor constante $g_{\boldsymbol{\lambda}}(z_{1:t-1}) = P_0 \in [0, 1]$. Isto significa que $\{z_t\}_{t\in[2,T]}$ são variáveis binárias independentes parametrizadas por $\boldsymbol{\lambda} = \{P_0\}$, de modo que se tem a Equação A.6.

$$z_t \sim_{iid.} \text{Binary}(P_0). \tag{A.6}$$

A.2 Processo de atribuição de locutor

Um dos maiores desafios em diarização de locutor é poder determinar o número total de locutores para cada uterância sem informações prévias. Para modelar o comportamento de turno dos locutores em uma uterância, foi proposta a utilização do processo de restaurante chinês dependente da distância (ddCRP), um modelo não paramétrico Bayesiano capaz

de modelar um número ilimitado de oradores. Especificamente, quando $z_t = 0$, o orador permanece o mesmo. Quando $z_t = 1$, faz-se como mostrado na Equação (A.7).

$$\begin{aligned} P(y_t = k | z_t = 1, y_{1:t-1]}) &\propto N_{k,t-1}, \\ P(y_t = K_{t-1} + 1 | z_t = 1, y_{1:t-1}) &\propto \alpha. \end{aligned} \tag{A.7}$$

Aqui $K_{t-1} := \max y_{1:t-1}$ é o número total de locutores únicos até a $(t-1)$-ésima entrada. Como $z_t = 1$ indica uma mudança de locutor, temos $k \in [K_{t-1}] \setminus \{y_{t-1}\}$. Ademais, considera-se $N_{k,t-1}$ o número de blocos para o locutor k em $y_{1:t-1}$. Um bloco é definido como uma subsequência de comprimento máximo de segmentos contínuos que pertencem a um único locutor. Por exemplo, se $y_{1:6} = (1, 1, 2, 3, 2, 2)$, então existem quatro blocos $(1, 1)|(2)|(3)|(2, 2)$ separados pela barra vertical, com $N_{1,5} = 1, N_{2,5} = 2, N_{3,5} = 1$.

A probabilidade de voltar a um locutor que apareceu anteriormente é proporcional ao número de falas contínuas feitas pelo mesmo. Também existe a chance de mudar para um novo locutor, com uma probabilidade proporcional à constante α. A distribuição conjunta de $\mathbf{Y}$ dado $\mathbf{Z}$ é definida pela Equação (A.8).

$$P(\mathbf{Y}|\mathbf{Z}, \alpha) = \frac{\alpha^{K_T - 1} \prod_{k=1}^{K_T} \Gamma(N_{k,T})}{\prod_{t=2}^{T} (\sum_{k \in [K_{t-1}] \setminus \{y_{t-1}\}} N_{k,t-1} + \alpha)^{\mathbb{1}(z_t = 1)}} \tag{A.8}$$

A.3 Geração de sequência

A suposição básica feita é de que a sequência de observação de *embeddings* $\mathbf{X}$ é gerada por distribuições que são parametrizadas pela saída de uma RNN. Tal RNN possui múltiplas instancias, correspondendo a diferentes locutores, os quais compartilham o mesmo conjunto de parâmetros RNN $\boldsymbol{\theta}$. No trabalho proposto, foi utilizada uma GRU como modelo RNN, de forma a memorizar dependências de longo prazo.

No momento t, define-se $\mathbf{h}_t$ como o estado da GRU correspondente ao locutor y_t, e a Equação (A.9) como a saída da rede por inteiro, sendo que esta pode conter outras camadas.

$$\mathbf{m}_t = f(\mathbf{h}_t | \boldsymbol{\theta}) \tag{A.9}$$

Seja $t' := \max\{0,\ s < t :\ y_s = y_t\}$ a última vez que o locutor y_t foi visto antes de t, então define-se na Equação (A.10) o estado $\mathbf{h}_t$. Pode-se assumir que $\mathbf{x}_0 = \mathbf{0}$ e $\mathbf{h}_0 = \mathbf{0}$, significando que todas as instâncias GRU são inicializadas com o mesmo estado zerado.

$$\mathbf{h}_t = \mathrm{GRU}(\mathbf{x}_{t'}, \mathbf{h}_{t'}|\boldsymbol{\theta}) \tag{A.10}$$

Com base nas saídas GRU, assume-se que os *embeddings* de locutor são modelados pela Equação (A.11), onde $\boldsymbol{\mu}_t = (\sum_{s=1}^{t} \mathbb{1}(y_s = y_t))^{-1} \cdot (\sum_{s=1}^{t} \mathbb{1}(y_s = y_t)\mathbf{m}_s)$ é a saída GRU média para o orador y_t.

$$\mathbf{x}_t|\mathbf{x}_{1:t-1}, y_{1:t} \sim \mathcal{N}(\boldsymbol{\mu}_t, \sigma^2\mathbf{I}). \tag{A.11}$$

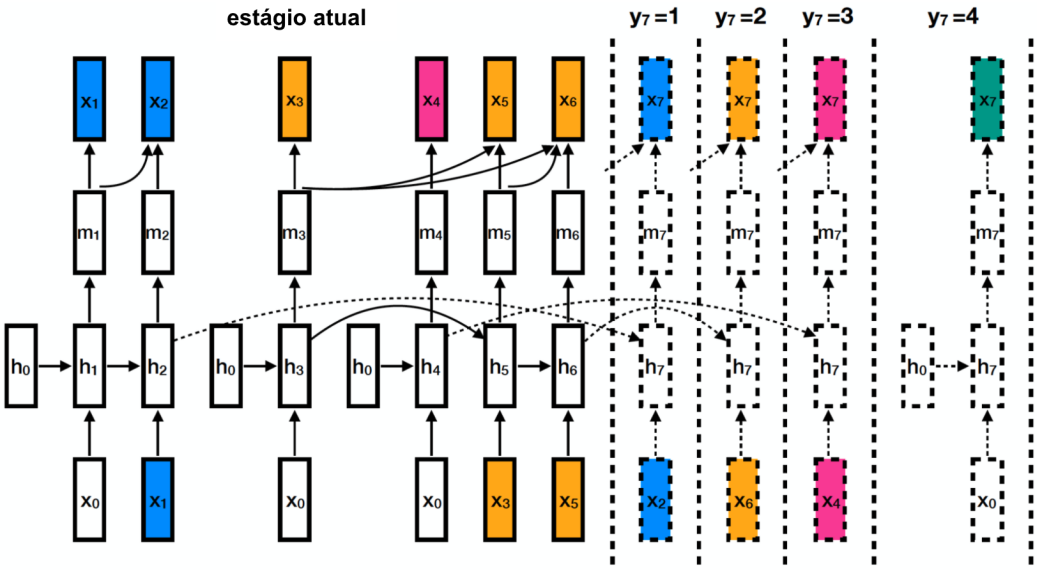
A.4 Resumo do modelo

Em resumo, a UIS-RNN é uma rede de parâmetros compartilhados que modela cada locutor individualmente a partir da sequência de *embeddings* de entrada, que representam características distintas dos locutores. Este processo ocorre enquanto os estados (*states*) da RNN para diferentes locutores se intercalam (*interleave*) no domínio do tempo. Para a atribuição de locutores, baseia-se no processo de restaurante chinês dependente de distâncias (ddCRP) definido por Blei e Frazier (2011), de forma a acomodar um número desconhecido de falantes (*unbounded*).

Como todos os componentes são representados por modelos treináveis, a UIS-RNN pode ser treinada de maneira supervisionada ao encontrar parâmetros que maximizam $\log P(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ sobre os dados de treino. E seu processo de predição é conduzido buscando pelo $\mathbf{Y}$ que maximiza $\log P(\mathbf{X}, \mathbf{Y})$.

Na Figura 16 está ilustrada uma representação da UIS-RNN, onde $\mathbf{Z}$ e $\boldsymbol{\lambda}$ são omitidos para uma demonstração simplificada. No estágio atual (mostrado em linhas sólidas) $y_{1:6} = (1, 1, 2, 3, 2, 2)$. Existem quatro opções para y_7: $1, 2, 3$ (locutores existentes) e 4 (um novo locutor). A probabilidade de gerar uma nova observação $\mathbf{x}_7$ (mostrada em linhas tracejadas) depende tanto da sequência anterior de atribuição de rótulos $y_{1:6}$ quanto da sequência anterior de observações $\mathbf{x}_{1:6}$.

Figura 16 – Processo generativo da UIS-RNN



Fonte: Zhang et al. (2019).

Nota: As cores indicam os rótulos de cada locutor por segmento. Existem 4 opções para y_7 dado $\mathbf{x}_{1:6}, y_{1:6}$.

ANEXO B – CONJUNTO DE DADOS

B.1 *VoxCeleb*

O *VoxCeleb* consiste em um conjunto de dados dividido em dois estágios, denominados *VoxCeleb1* (NAGRANI et al., 2020) e *VoxCeleb2* (CHUNG; NAGRANI; ZISSERMAN, 2018). Nestas compilações, os dados foram extraídos de vídeos disponíveis na plataforma *YouTube* e possuem sinais de fala de diversas celebridades, tais como atores e empresários. O conjunto de dados é equilibrado em relação a gênero, sotaque, profissão, idade e etnia dos locutores.

Os cenários dos vídeos incluem entrevistas, discursos, gravações em estúdios e também ao ar livre. Dessa maneira, a qualidade do áudio e do respectivo equipamento de captação são variáveis. Assim, todos os vídeos possuem elementos e ruídos do cotidiano, tais como risadas, conversas de fundo e sobreposição de fala.

O *VoxCeleb1* contém mais de 100.000 falas de 1.251 locutores; enquanto o *VoxCeleb2* possui mais de 1 milhão de falas de 6.000 locutores. As estatísticas mais relevantes deste conjunto de dados estão compiladas na Tabela 8.

Tabela 8 – Estatísticas do conjunto de dados *VoxCeleb*

Conjunto de dados	***VoxCeleb1***	***VoxCeleb2***
Locutores	1.251	6.112
Locutores masculinos	690	3.761
Total de vídeos extraídos	22.496	150.480
Total de horas	352	2.442
Total de uterâncias	153.516	1.128.246
Média de vídeos por locutor	18	25
Média de uterâncias por locutor	116	185
Duração média (s) das uterâncias	8,2	7,8

Fonte: Chung, Nagrani e Zisserman (2018).

B.2 *VoxConverse*

O *VoxConverse* é um conjunto de dados que apresenta cenários ainda mais desafiadores do que o *VoxCeleb*. O que diferencia este conjunto de dados é a quantidade de locutores por vídeo: em média, cada vídeo possuí de 4 a 6 falantes. Os vídeos foram extraídos

da plataforma *YouTube* e incluem debates políticos, discussões, entrevistas, *talk shows*, segmentos de notícias e de comédia. Os vídeos possuem um som ambiente dinâmico, apresentando ruídos como risadas e aplausos (CHUNG et al., 2020b).

O conjunto de desenvolvimento consiste em 216 vídeos, totalizando 1.218 minutos e 8.268 mudanças de locutor. Já o conjunto de teste contém 232 vídeos, atingindo o total de 2.612 minutos. As estatísticas mais relevantes do conjunto de dados *VoxConverse* está compilado na Tabela 9.

Tabela 9 – Estatísticas do conjunto de dados VoxConverse. Dados apresentados em mínimo / média / máximo.

Conjunto	**Locutores**	**Duração (s)**	**Fala (%)**	**Sobreposição (%)**
Dev	1 / 4,5 / 20	22 / 338,2 / 1.097,4	10,7 / 93,2 / 99,8	0 / 3,8 / 28,7
Teste	1 / 6,5 / 21	26 / 675,6 / 1.200,0	46,9 / 89,6 / 100	0 / 3,1 / 29,8

Fonte: Chung et al. (2020b).

B.3 VCTK *Corpus*

A base de dados VCTK foi elaborada pelo CSTR (*Centre for Speech Technology Research*), na universidade de Edinburgh. Este conjunto consiste em gravações de 110 locutores da língua inglesa, abrangendo uma variedade de sotaques. Em cada gravação, o locutor lê cerca de 400 frases selecionadas de jornais e de exercícios fonoaudiológicos. Os áudios foram coletados por meio de um microfone omnidirecional em uma câmara semi anecoica, suprimindo ruídos externos.

Este conjunto de dados se apresenta em duas versões diferentes. A primeira consiste na versão *RAW*, isto é, sem compressão, com 96kHz de frequência de amostragem e quantização de 24 bits. Já a segunda versão possui frequência de amostragem de 48kHz, quantização de 16 bits, sem compressão (YAMAGISHI et al., 2019).

B.4 *LibriSpeech*

O *LibriSpeech* é um conjunto de dados montado a partir de *audiobooks* de domínio público, originalmente extraídos da plataforma *LibriVox*. Totalizam-se cerca de 1.000 horas de fala, amostradas em 16kHz. O *Librispeech* foi compilado com intuito de ser utilizado para treinamento e avaliação de sistemas de reconhecimento de fala (PANAYOTOV et al., 2015).

O conjunto de dados foi segmentado com diferentes características para as etapas de treinamento, teste e desenvolvimento. Para o treinamento, o conjunto de dados foi segmentado em trechos de 25 a 30 minutos, com boa qualidade de áudio. Já as etapas de teste e desenvolvimento apresentam dados mais desafiadores, com trechos de 8 a 10 minutos.

Além disso, as etapas são subclassificadas de acordo com a discrepância entre as palavras da transcrição automática e do texto original do livro. Este índice é chamado de *Word Error Rate* ou WER. Quanto menor a discrepância, mais "limpo"("*clean*") é considerado o áudio. Esta classificação está apontada na Tabela 17.

Figura 17 – Estatísticas da base de dados *LibriSpeech*

Conjunto	**Horas**	**Minutos por Locutor**	**Locutores Femininos**	**Locutores Masculinos**	**Total de Locutores**
Dev-Clean	5,4	8	20	20	40
Test-Clean	5,4	8	20	20	40
Dev-Other	5,3	10	16	17	33
Test-Other	5,1	10	17	16	33
Train-Clean-100	100,6	25	125	126	251
Train-Clean-360	363,6	25	439	482	921
Train-Other-500	496,7	30	564	602	1166

Fonte: Panayotov et al. (2015).