Universidade Tecnológica Federal do Paraná – UTFPR
ELTD1 S11 Introdução À Modelagem e Aprendizado
Semestre 2023/2
Prof. Heitor Silvério Lopes

Projeto Embarcado:

Classificação de emoções
a partir da prosódia da fala

Daniel A. P. de Castro
danielcastro@alunos.utfpr.edu.br - (41) 99777-1037

Gustavo F. Armênio
gustavofardoarmenio@alunos.utfpr.edu.br - (49) 99810-7140

1 Introdução

Desde os princípios da computação, a necessidade da interação da máquina com um usuário humano é predominante para uma alta gama de aplicações. A comunicação de dados entre um humano e a máquina é possível em diversos níveis de abstração e modalidades - comunicação numérica, textual, visual, sonora, etc. O conceito de Interface Humano-Computador (IHC) foi popularizado por Card, Moran e Newell (1983) [1] e desde então se consolidou como um campo de estudo que visa ao desenvolvimento meios novos e aprimorados para a interação entre uma máquina e seu usuário.

A voz é o meio primordial de comunicação entre seres humanos, a partir da linguagem natural, em conjunto com a comunicação não-verbal. Para a máquina, a linguagem natural do ser humano não é trivial, e por isso é geralmente codificada a partir de uma sintaxe simplificada, já que há uma alta complexidade na morfologia e na semântica da linguagem natural humana.

Com os avanços tecnológicos atuais, as máquinas têm a capacidade de explorar o universo da linguagem natural humana. Por meio do reconhecimento de fala, conseguem converter o sinal sonoro da voz em texto, utilizando ferramentas como o Whisper [2]. Através de técnicas de Processamento de Linguagem Natural (NLP), também conseguem processar textos e extrair contextos, palavras-chave e até mesmo emoções. Nesse processo, no entanto, uma das componentes que carregam semântica - muitas vezes determinante para o significado - é a entonação com a qual se pronuncia a fala.

A entonação de uma frase pode ser essencial para a definição do sentido que se propõe comunicar, e está muito conectada com a emoção. Por exemplo, ao analisar a frase "Você já terminou o seu trabalho?"com uma entonação de surpresa, podemos concluir que o trabalho foi finalizado antes do esperado ou até mesmo uma dúvida se o trabalho foi ou não finalizado; ao analisarmos com uma entonação de raiva, podemos inferir que o trabalho está atrasado.

Dessa maneira, a Classificação de Emoções a partir da prosódia da fala - análise da entonação, ritmo, tonalidade das sílabas, etc. - se mostra útil para

promover uma compreensão mais robusta e complexa da informação fornecida a partir da voz humana.

2 Dados

Em uma busca extensa, foram encontrados três datasets que possuem gravações de frases ditas por seres humanos, cada uma associada e demarcada com uma emoção, na Língua Portuguesa. O CORAA SER v1 [3] é composto por 50 minutos de áudios, com frases classificadas entre uma emoção neutra ou não-neutra. O emoUERJ [4] contém cerca de 400 amostras, gravadas por 4 atores e 4 atrizes, para as emoções felicidade, raiva, tristeza e neutralidade. O VERBO [5] possui 1167 amostras de frases gravadas por 6 atores e 6 atrizes para as seguintes emoções: Felicidade, Nojo, Medo, Neutro, Raiva, surpresa e Tristeza.

2.1 Dataset emoUERJ

O dataset escolhido para os testes foi o emoUERJ [4]. Em contrapartida ao CORRA SER v1 [3], o emoUERJ possui três emoções bem definidas além da neutralidade. Por mais que o VERBO [5] possua mais dados, para ainda mais emoções, não foi possível obter acesso a eles a tempo.

A distribuição dos dados do emoUERJ é quase uniforme, mas ainda sim desbalanceada. Como pode ser visto na Figura 1, a distribuição das amostras de cada classe é 92, 100, 91 e 94 para, respectivamente, neutro, tristeza, felicidade e raiva. Assim, o dataset possui 377 amostras originais.

Esses dados estão disponíveis como trilhas de áudio em formato .wav, com em média 5 segundos de duração. As frases recitadas foram escolhidas por cada dublador entre uma lista de frases pré-definidas:

- Não importa quem está certo.
- Você perde tempo demais com a Internet.
- A garrafa está na geladeria.
- Eu estou me sentindo doente hoje.
- Eu estou um pouco atrasado.
- Nos fins de semana, eu sempre ia para a casa dele(a).
- De quem são essas malas que estão debaixo da mesa?
- Ele volta na quarta-feira.
- Já chega! Eu vou tomar um banho e ir para a cama.
- Você poderia arrumar a mesa, por favor?

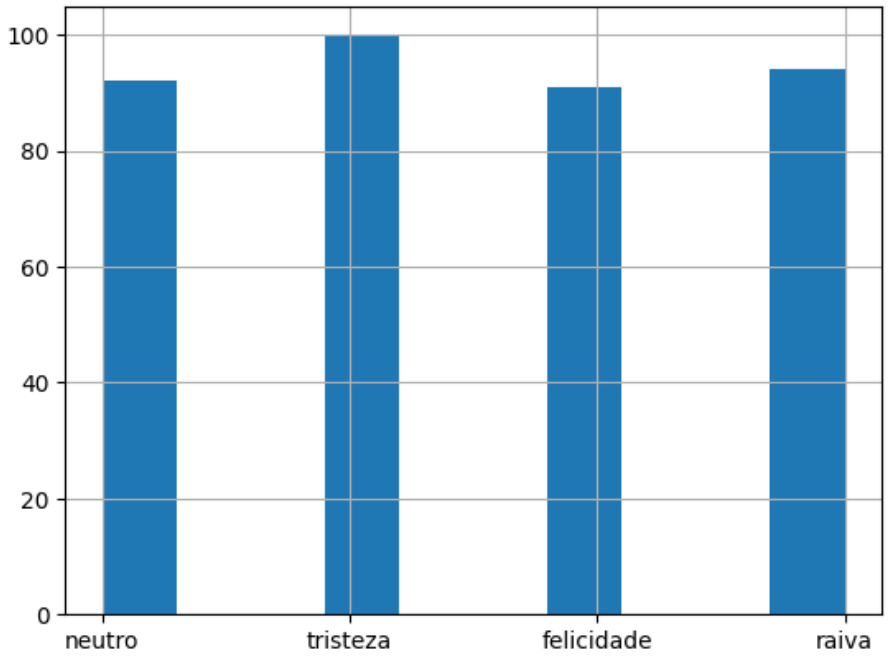

Figura 1: Distribuição de amostras para cada classe no emoUERJ [4].

3 Métodos

O projeto foi realizado em linguagem *Python* e *Jupyter Notebook*, utilizando várias bibliotecas para pré-processamento, manipulação de dados, extração de features, construção dos modelos de classificação, extração de métricas, etc.

3.1 Pré-processamento dos dados

Disponíveis em arquivos .wav, os dados devem ser transformados em sinais para poderem ser manipulados. Para isso, foi aplicada a biblioteca Librosa [6], que permite obter um sinal temporal e uma taxa de amostragem a partir do arquivo .wav. O sinal de uma amostra pode ser visualizado na Figura 2, com amplitude normalizada.

3.1.1 Data Augmentation

Com o intuito de expandir o número de amostras, foram aplicadas técnicas de Data Augmentation, sabendo que os modelos de classificação podem ter seu desempenho muito beneficiado pelo aumento na quantia de dados. Utilizando a biblioteca Audiomentations [7], três processamentos foram aplicados a cada amostra, baseado em Ottoni e Cerqueira (2023) [8]:

1. Ruído gaussiano com fator de amplificação aleatória entre 0.001 e 0.01.

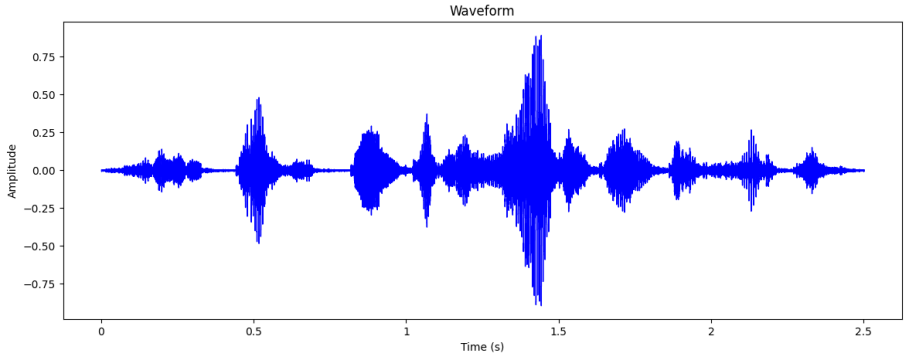

Figura 2: Sinal obtido pela biblioteca Librosa [6] a partir de uma das amostras em .wav.

2. Distorção de tempo (Time Stretch) aleatória entre 0.8 e 1.25 vezes.
3. Deslocamento de tom (Pitch Shift) com variação aletória mínima de -2 semitons e máxima de 2 semitons.

Um exemplo de amostra gerada por Data Augmentation pode ser vista na Figura 3, a partir da amostra vista na Figura 2.

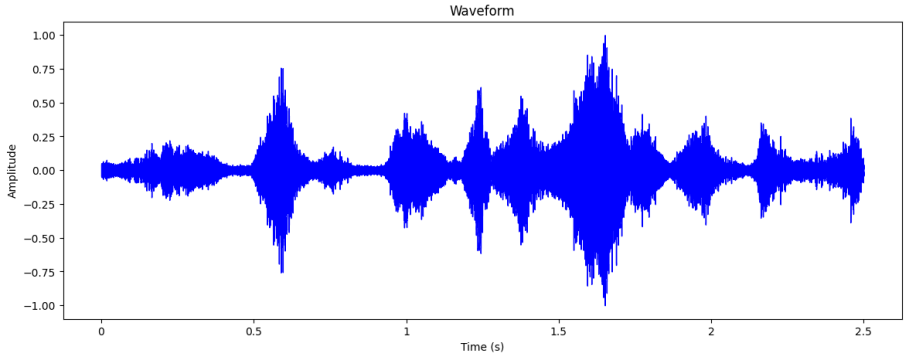

Figura 3: Sinal com Data Augmentation a partir do sinal da Figura 2

Para poder verificar o impacto de diversos passos de Data Augmentation, cada amostra foi processada 2 vezes, resultando em um dataset completo com o triplo do tamanho, 1131 amostras.

3.2 Extração de features

Sendo um passo essencial e determinante no desempenho dos modelos de classificação, a escolha do método de extração de features foi feita avaliando uma série de possibilidades. Como o objetivo é inferir uma classe para todo o trecho de áudio, a extração foi realizada de forma a transformar cada amostra do dataset em um único vetor de características. Entre os métodos avaliados estão 2

vistos na disciplina - as bibliotecas TSFEL [9] e pyaudioanalysis [10] -, 2 previamente testados por Ottoni e Cerqueira (2023) - MFCC e Mel Spectrogram - [8] e 1 utilizado por Marcancini (2022) [3] - o Praat. A avaliação de cada método não contou somente com os resultados finais dos modelos de classificação, mas também o tempo médio de extração para uma amostra, pois a aplicação em um sistema embarcado exige uma inferência mais próxima do tempo real quanto for possível.

3.2.1 TSFEL

Com a biblioteca TSFEL [9], é possível extrair 778 características estatísticas, no domínio do tempo e no domínio da frequência. Muitas características podem ser úteis para o modelo, contudo, aumentam substancialmente o tempo de extração, treinamento e inferência, ou até podem introduzir ruídos. Experimentalmente, determinou-se um tempo médio de extração de 20,38s. Por esse motivo, o TSFEL foi desclassificado e não prosseguiu para as etapas seguintes.

3.2.2 pyaudioanalysis

Aplicando a extração com a classe ShortTermFeatures[1], o sinal é dividido em quadros menores e para cada um calcula-se 64 features a partir de 11 elementos distintos do sinal (Zero Crossing Rate, Energia, Centroide Espectral, etc.). Por fim, para obter os valores para um único vetor, é calculada a média de cada feature. O tempo médio de extração para uma amostra foi 0,85s.

3.2.3 MFCC

Para extrair características MFCC - Mel-Frequency Cepstrum Coefficients - da fala, inicia-se dividindo o sinal em quadros menores, aplicando uma DFT a cada quadro, e calculando um Banco de Filtros Mel-Escalados (MSFB) [8]. Este banco de filtros reflete a percepção de frequência do ouvido humano e resulta em 26 valores que representam a energia de cada quadro. Em seguida, são calculadas energias logarítmicas e aplicada a DCT - Transformada Cosseno Discreta - para gerar os coeficientes MFCCs [8]. Um exemplo de MFCCs para uma das amostras está apresentado na Figura 4. Utilizando a biblioteca Librosa [6], basta definir o número de coeficientes a ser calculados a partir do sinal. O número de coeficientes escolhido foi 13 e para transformar os quadros em um único vetor, foram calculadas a média, o desvio padrão o mínimo e o máximo, resultando em 51 características. O tempo médio de extração foi 0,047s.

3.2.4 Mel Spectrogram

O Mel Spectrogram analisa o sinal no domínio da frequência. Aplicando uma FFT em janelas de tempo, divide o espectro de frequência em frequências de es-

[1] `https://github.com/tyiannak/pyAudioAnalysis/wiki/3.-Feature-Extraction`

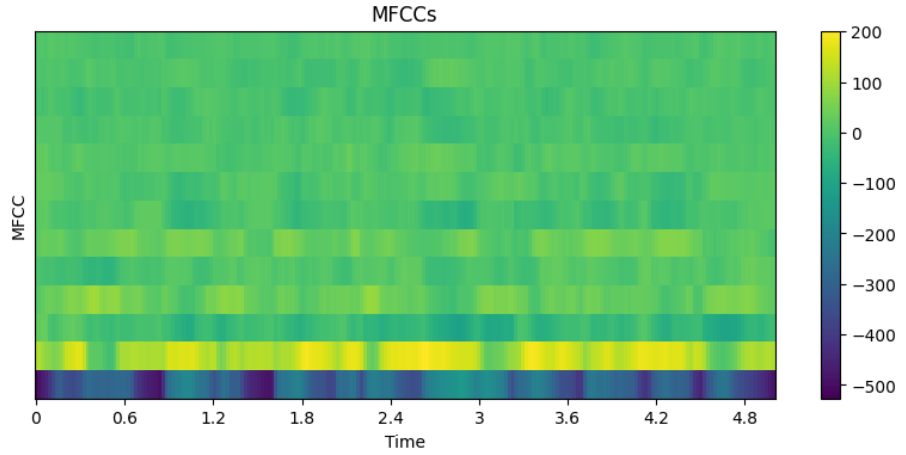

Figura 4: MFCCs obtidos do sinal da Figura 2

cala Mel espaçadas uniformemente, produzindo um Mel Spectrogram para cada janela. Em seguida, decompõe a magnitude do sinal em componentes correspondentes às frequências na escala Mel que, como dito anteriormente, reflete a percepção de frequência do ouvido humano. Um exemplo de Mel Spectrogram para uma das amostras está apresentado na Figura 5. Aqui também foram calculadas a média, desvio padrão, mínimo e máximo para as janelas. Com a biblioteca Librosa [6], o resultado foi 512 características, com tempo médio de 0,100s.

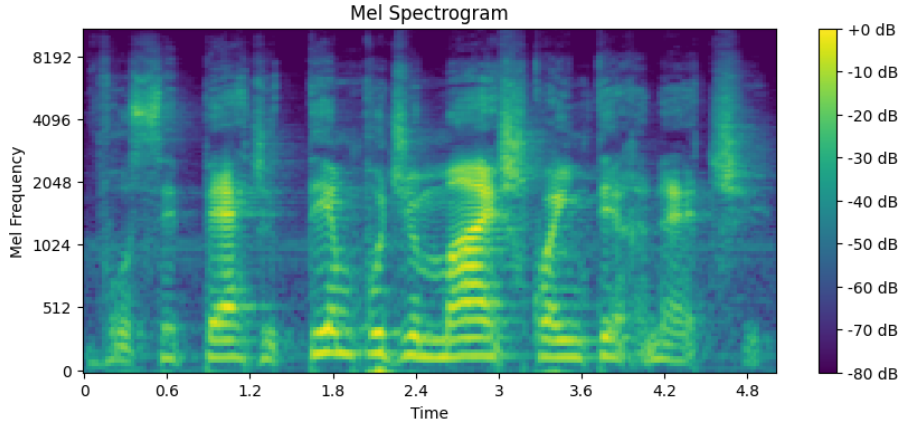

Figura 5: Mel Spectrogram obtido do sinal da Figura 2

3.2.5 Praat

A biblioteca Parselmouth [11] implementa em Python diversas funcionalidades do software Praat [12] para a análise de trilhas de áudio. Entre os atributos possíveis de extrair na análise vocal - prosódia - estão os de intensidade, tom, variação da frequência (jitter) e amplitude (shimmer), as ressonâncias específicas na resposta vocal (formante), a razão harmônicos-ruído e a razão glotal-ruído. No total, foram 56 características extraídas em um tempo médio de 3,1s.

3.3 Modelos de Classificação

Considerando a razão entre tempo de treinamento/inferência e desempenho, foram implementados dois modelos, um SVM e uma Rede Neural.

3.3.1 Support Vector Machine (SVM)

O classificador SVM foi construído a partir da biblioteca scikit-learn [13] - a partir da classe SVC[2], com parâmetro de regularização C=1.0 e kernel linear. O treinamento contou com uma etapa anterior de normalização das features.

3.3.2 Rede Neural

Para elaborar a arquitetura da rede neural, a biblioteca de Deep Learning Keras [14] foi primordial. A arquitetura utilizada está apresentada na Tabela 1.

O treinamento foi realizado com batch size de 8 amostras e 100 epochs. O otimizador aplicado foi o *adam* e a perda foi calculada a partir da métrica *sparse_categorical_crossentropy*. O controle da taxa de aprendizado é feito dinamicamente, com taxa mínima de 0.0001, de forma a reduzí-la caso o aprendizado fique estagnado. Por fim, o modelo para antecipadamente caso a perda passe a variar muito pouco por muitas épocas. Assim como para o SVM, os vetores de característica foram normalizados antes do treinamento.

4 Resultados

De forma a comparar os resultados com diferentes combinações de métodos e condições, foi avaliado o f1_score médio de 5 folds em validação cruzada para as 4 técnicas de extração de features - Praat, MFFC, Mel Spectrogram e pyaudioanalysis -, para o SVM e para a rede neural, considerando zero (377 amostras), um (754 amostras) e dois casos de Data Augmentations (1131 amostras). Os scores para o SVM estão apresentados na Tabela 2 e para a rede neural na Tabela 3.

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`

Camada	Especificações
[INPUT] Conv1D (+ relu)	filters = 64, kernel_size = 3
MaxPooling1D	pool_size=2
BatchNormalization	
Dropout	rate=0.25
Conv1D (+ relu)	filters = 128, kernel_size = 3
MaxPooling1D	pool_size=2
BatchNormalization	
Dropout	rate=0.25
Conv1D (+ relu)	filters = 256, kernel_size = 3
MaxPooling1D	pool_size=2
BatchNormalization	
Dropout	rate=0.25
Flatten	
Dense (+ relu)	units=512
BatchNormalization	
Dropout	rate=0.5
[OUTPUT] Dense (+ softmax)	units=4 (número de classes)

Tabela 1: Arquitetura da Rede Neural implementada com o Keras [14]

SVM F1 Média	Praat	MFCC	Mel Spectrogram	pyaudioanalysis
original	0.79309	0.82147	0.72944	0.70234
1 augment	0.77360	0.83445	0.71440	0.72476
2 augment	0.77294	**0.85533**	0.72705	0.76127

Tabela 2: Resultados da média de F1 Score para validação cruzada com 5 folds para o SVM.

RN F1 Média	Praat	MFCC	Mel Spectrogram	pyaudioanalysis
original	0,83993	0,84514	0,82642	0,81392
1 augment	0,85652	0,85641	0,84599	0,81812
2 augment	**0,90611**	0,89175	0,88888	0,86142

Tabela 3: Resultados da média de F1 Score para validação cruzada com 5 folds para a Rede Neural.

Primeiramente, é evidente perceber a diferença entre os resultados do SVM e da Rede Neural, com uma diferença média de quase 10% em vantagem da Rede Neural. Também, é perceptível o efeito do Data Augmentation: para a Rede Neural, aumentou o F1 score em todos os casos; para o SVM, o MFCC e o pyaudioanalysis foram beneficiados, já o Praat e o Mel Spectrogram sofreram decréscimo no desempenho.

O melhor resultado obtido para o dataset emoUERJ foi com o método de ex-

tração Praat, com 2 Data Augmentations, na Rede Neural - 90,611% de F1 Score médio. Em comparação, o melhor resultado para o SVM foi com o MFCC e 2 Data Augmentations - 85,533%. Para o melhor resultado, a evolução do F1 Score pode ser visualizada na Figura 6 e a Matriz de Confusão na Figura 7.

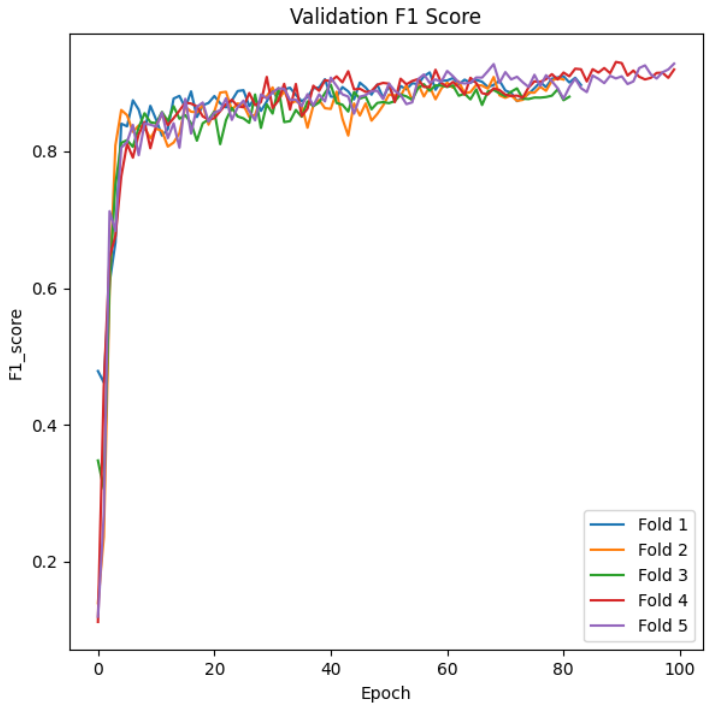

Figura 6: Evolução do F1 Score no melhor treinamento (Praat + 2 DA + RN)

5 Captura de Áudio e Inferência no Sistema Embarcado

Inicialmente, a ideia era que a implementação final do modelo ocorreresse em uma Raspberry Pi 1 Model B. Contudo, conforme os experimentos foram sendo realizados, percebeu-se a incompatibilidade do microcontrolador com a complexidade da tarefa, de forma que a inferência ocorra em quase tempo-real. Por isso, o sistema embarcado escolhido passou a ser a Jetson Nano, um computador compacto da NVIDIA que possui um sistema operacional Linux e uma placa gráfica que permite a aceleração gráfica, podendo reduzir o tempo de inferência de redes neurais.

Para a entrada de áudio e o controle do usuário, foi incorporado um microfone e um teclado. O fluxograma do processo completo de captura e inferência está detalhado na Figura 8.

Realizando experimentos com as nossas vozes, o modelo não obteve resultados tão bons, o que pode ter ocorrido por uma série de motivos. A questão

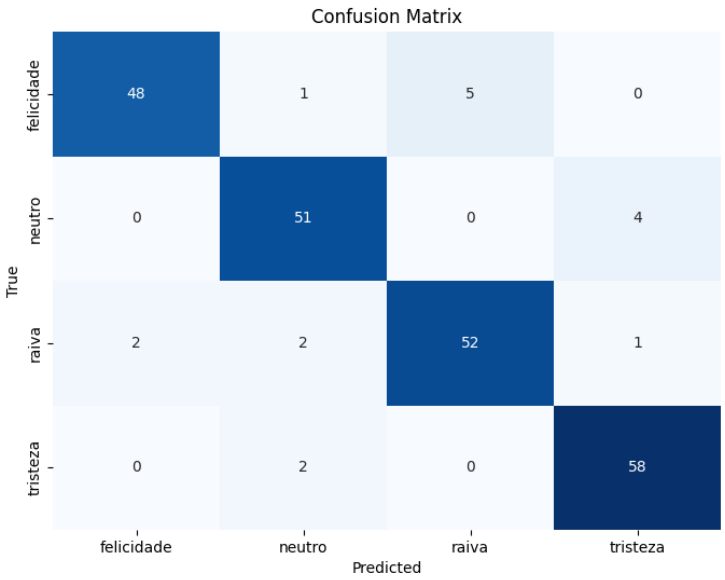

Figura 7: Matriz de confusão para o melhor treinamento (Praat + 2 DA + RN)

principal que devemos levar em consideração é a subjetividade do problema, visto que a emoção expressa por uma determinada fala pode ser interpretada distintamente por pessoas diferentes. Além disso, é necessário ressaltar que o dataset foi elaborado com a voz de dubladores profissionais, que sabem emular emoções em suas vozes de forma consciente. Por fim, é evidente que um dataset com poucas amostras como o emoUERJ não consegue representar uma gama vasta de timbres e formas de se falar e, portanto, não é extremamente capaz de promover uma alta capacidade de generalização.

6 Conclusões

Com o desenvolvimento deste projeto, foi possível aprender sobre as particularidades da análise de sinais temporais, mais especificamente áudios, e como as características destes variam a partir da fala, com a aplicação das quatro técnicas de extração de características - em especial o Praat. Além disso, evidenciou-se a superioridade das Redes Neurais em comparação com o SVM para tarefas mais complexas de classificação. Os resultados obtidos para o dataset emoUERJ foram muito satisfatórios - com o melhor resultado sendo 90,611% de F1 Score médio para extração Praat, Rede Neural e 2 passos de Data Augmentation - e percebeu-se a grande vantagem de se aplicar técnicas de Data Augmentation. A implementação do sistema em um sistema embarcado se provou desafiadora, exigindo a troca de uma Raspberry Pi 1 Model B por uma placa NVIDIA

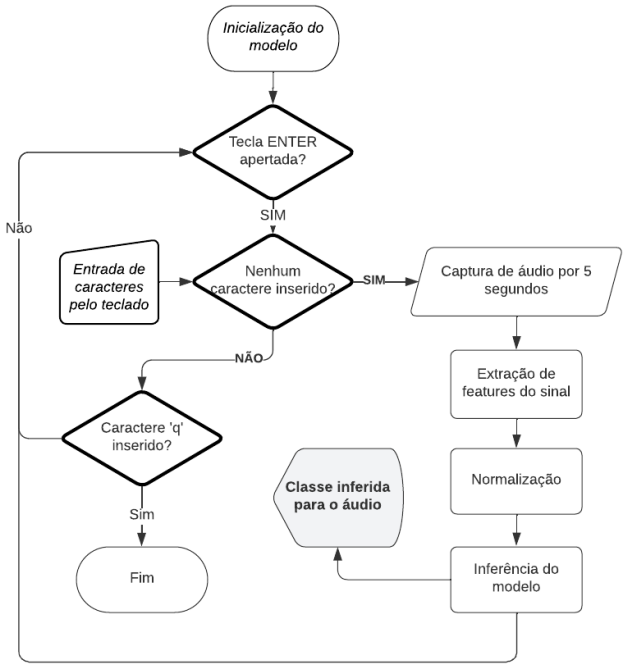

Figura 8: Fluxograma do funcionamento do projeto embarcado, desde a captura até a inferência.

Jetson Nano, capaz de realizar aceleração gráfica. Assim, foi possível obter um sistema que captura o áudio em tempo-real e retorna a inferência da classe poucos segundos depois. Os testes realizados no sistema embarcado não foram tão satisfatórios quanto no dataset por alguns possíveis motivos que incluem a subjetividade inerente ao problema e a generalização insuficiente para o domínio de vozes exterior ao dataset. Para o aumento da capacidade de generalização do modelo, é evidente que seria necessário uma quantia maior e mais variada de dados.

Referências

[1] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction*. CRC Press, 5 2018.

[2] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. 12 2022.

[3] Ricardo Marcacini. Coraa ser v1. Disponível em: `https://github.com/rmarcacini/ser-coraa-pt-br`, 2022.

[4] Rodrigo Gregory Bastos Germano, Michel Pompeu Tcheou, Felipe da Rocha Henriques, and Sergio Pinto Gomes Junior. emouerj: an emotional speech database in portuguese, 2021.

[5] José R. Torres Neto, Geraldo P.R. Filho, Leandro Y. Mano, and Jó Ueyama. Verbo: Voice emotion recognition database in portuguese language. *Journal of Computer Science*, 14(11):1420–1430, November 2018.

[6] Brian Mcfee, Colin Raffel, Dawen Liang, Daniel P W Ellis, Matt Mcvicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in python. *PROC. OF THE 14th PYTHON IN SCIENCE CONF*, 2015.

[7] Iver Jordal. audiomentations. Disponível em: `https://github.com/iver56/audiomentations`, 2023.

[8] Lara Toledo Cordeiro Ottoni and Jes de Jesus Fiais Cerqueira. Optimizing speech emotion recognition: Evaluating combinations of databases, data augmentation, and feature extraction methods. In *XVI Brazilian Conference on Computational Intelligence (CBIC 2023)*, Salvador, October 8th to 11th 2023.

[9] Marília Barandas, Duarte Folgado, Letícia Fernandes, Sara Santos, Mariana Abreu, Patrícia Bota, Hui Liu, Tanja Schultz, and Hugo Gamboa. Tsfel: Time series feature extraction library. *SoftwareX*, 11:100456, 2020.

[10] Theodoros Giannakopoulos. pyaudioanalysis: A python library for audio feature extraction, classification, segmentation and applications. Disponível em: `https://github.com/tyiannak/pyAudioAnalysis`, 2023.

[11] Yannick Jadoul. Parselmouth - praat in python, the pythonic way. Disponível em: `https://github.com/YannickJadoul/Parselmouth/tree/stable`, 2023.

[12] Paul Boersma and David Weenink. Praat: doing phonetics by computer [Computer software]. `http://www.praat.org/`, 2023.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] François Chollet et al. Keras. *GitHub repository*, 2015.