UNIVERSIDADE FEDERAL DE GOIÁS
INSTITUTO DE INFORMÁTICA
APRENDIZADO DE MÁQUINA NÃO SUPERVISIONADO
BACHARELADO EM INTELIGÊNCIA ARTIFICIAL

Alunos: Alex Echeverria - Matrícula: 202005469
Heloisy Rodrigues - Matrícula: 202005485
2022-2

Prova II

1- **Explicação do problema**

A classificação de emoção na voz é uma tarefa crucial para compreender como as pessoas se sentem e como esses sentimentos afetam sua comunicação. A capacidade de detectar e entender as emoções humanas é fundamental para uma variedade de aplicações, como análise de chamadas de atendimento ao cliente, avaliação de entrevistas de emprego, análise de diálogos em jogos de voz, entre outros. Em um projeto de pesquisa e desenvolvimento feito pela Sales Holding juntamente com a Universidade Federal de Goiás, deseja-se classificar emoções na voz/ fala espontânea para fazer a geração de arte de acordo com tal sentimento. Sendo assim, um projeto inovador e com uma curva de aprendizado alta.

Um ponto notório é que a classificação de emoção na voz é uma tarefa desafiadora devido à complexidade e à variedade dos sinais de áudio. Isso porque há uma natureza subjetiva das emoções. No entanto, as redes neurais têm mostrado ser uma ferramenta eficaz para a classificação de emoção na voz.

Existem vários tipos de redes neurais que podem ser utilizadas para a classificação de emoção na voz, incluindo redes neurais feedforward, redes neurais recorrentes, redes neurais convolucionais e redes neurais de atenção. A precisão da classificação pode ser aprimorada usando técnicas avançadas, como extratores de características e técnicas de aprendizado profundo. O objetivo em questão deste relatório, é demonstrar como usaremos autoencoders para melhorar a classificação realizada pela rede neural convolucional.

Vale ressaltar que em situações do dia a dia, com um modelo de aprendizado profundo funcionando, não há significativos tratamentos dos dados, visto que geralmente necessita-se de um rápido processamento. Diante disso, é notório que lidaremos com áudios com ruídos dos mais variados tipos, e deve-se estar preparado para isso. Outro ponto importante para o desenvolvimento de uma solução robusta, é que ruídos não acarretam ganho de informação, logo, é desejável que sejam excluídos, a fim de focar na informação correta.

Há algumas abordagens para a remoção de ruídos em áudios, mas neste trabalho focaremos em usar autoencoders para a realização dessa atividade. Essa escolha foi feita pois os autoencoders são modelos de aprendizado de máquina não supervisionado que conseguem

aprender a identificar e eliminar componentes indesejados do sinal de áudio (ruídos) e lidar com dados incompletos e corrompidos.

2- **Abordagem do problema**

Como foi baseado no projeto em desenvolvimento, a abordagem do problema foi mais fácil. Houveram três pilares: construção de um dataset com ruídos, desenvolvimento de um autoencoder e aplicação da classificação com redes neurais. Discorreremos sobre cada uma dessas etapas.

A construção de um dataset com ruídos baseou-se em juntar o dataset multilingual até então utilizado no projeto (contendo os datasets: CREMA, EMODB, EMOVO, IEMOCAP, JL, MOSEI, SAVEE, URDU) com um novo dataset, exclusivamente de ruídos, denominado DEMAND. No dataset de ruídos, há diversos tipos, tais como atividades domésticas, natureza, rua, etc. Essa junção é de fundamental importância tanto para a abordagem do problema em si, quanto para o desenvolvimento do autoencoder. Para a realização do treino, 17294 amostras foram utilizadas e, para o teste, 1144 amostras fora do domínio. É importante dizer também que são 7 sentimentos analisados, sendo eles: felicidade, surpresa, tristeza, nojo, raiva, medo e neutro.

Vale ressaltar que os áudios foram pré-processados de forma a garantir que as amostras tenham sempre a taxa de amostragem esperada, apenas 1 único canal (mono) e o mesmo tamanho / duração. Isso tudo é garantido a partir de funções presentes no dataloader dos sinais. Assim, todos contém o shape de (1, 64000), o que corresponde a 4 segundos de áudio, dada a taxa de amostragem de 16 kHz.

Em relação à estrutura da rede autoencoder, utilizamos um modelo convolucional de 1 dimensão para o aprendizado de características latentes de cada áudio. Como foram utilizados áudios com um grande shape unidimensional, teve-se preocupação com a quantidade de dimensões latentes que deveriam ser representadas. Por isso, treinou-se diversas redes com diferentes tamanhos e formatos.

A primeira rede utilizada foi construída com o encoder possuindo 4 camadas convolucionais, em que os números de canais iam de 1 a 64, aumentando gradualmente. Também possui 1 camada de flattening, em que se converte os tensores em 64 canais para um tensor com quantidade fixa de elementos e unidimensional. Em seguida, há 1 camada fully connected, que admite o vetor flattened e o leva ao gargalo máximo da rede, um espaço de dimensões pré definidas pelo usuário pela variável “encoded_space_dim”. Não linearidades do tipo “ReLU” foram utilizadas após cada camada linear. A rede Decoder contém a mesma quantidade de camadas com as mesmas características, apenas na ordem inversa. Por fim, há a camada não-linear “tanh” (tangente hiperbólica), em que há a garantia que os áudios fiquem restritos ao intervalo [-1, 1].

Já uma segunda versão da rede é mais compacta, contendo apenas 1 camada convolucional tanto no encoder quanto no decoder. Há apenas essa camada, com não linearidade do tipo “ReLU”, em que o áudio é levado a 64 canais pelo encoder, depois trago de volta ao seu canal único pelo decoder.

Outros detalhes de implementação e hiperparâmetros podem ser conferidos no código da rede, em “/Autoencoder/Rede.py”.

Quanto à classificação dos sentimentos, os áudios são transformados em mel espectrograma, para serem passados na rede neural. Isso porque fazemos o uso de rede neural convolucional para extrair os padrões da imagem do mel espectrograma.

O problema é resolvido quando, de posse de um conjunto de dados ruidosos, fornecemos eles como entrada para o autoencoder, que processa os dados e remove ruídos, para então fornecermos os dados à rede de classificação de sentimentos, no intuito de melhorar a eficiência de classificação, aumentando sua assertividade.

3- **Resultados obtidos**

Após o treinamento das redes autoencoders, observou-se resultados horrorosos com a rede complexa. Não aprendeu absolutamente nada da distribuição dos dados, como pode-se observar na figura 1, que representa a curva de Loss nos dados de treino e validação ao longo do treinamento. Além disso, podemos observar na figura 2 o péssimo desempenho em reconstruir o sinal de forma adequada.

Já com a rede compacta, os resultados foram interessantes. A começar pela curva de loss dentro do esperado de um treinamento, como pode ser visto na figura 3. Além disso, é notável o quão bem ela consegue reconstruir o sinal original, como pode-se observar na figura 4. Foi o melhor resultado que obtivemos em relação ao autoencoder.

Para atingir esse resultado, treinou-se por 20 épocas com um batch size de 512 áudios a cada step. Demorou cerca de 3 horas para realizar o treinamento, em uma RTX 3060 mobile.

Mesmo assim, ao comparar-se manualmente um resultado, não percebemos redução de ruídos a partir da saída do autoencoder e o áudio original sem ruídos.

Visualizando os resultados da rede de classificação de sentimentos na voz, no nosso treino, obtivemos 63% de acurácia e na validação 55% de acurácia. Partindo para os testes, fizemos com um teste das 1144 amostras com ruído. Isso porque depois queremos analisar como são os resultados com os dados fornecidos pelo autoencoder. Vale ressaltar que no pouco tempo para o desenvolvimento do autoencoder mediante a prova, conseguimos diferenças na classificação dos sentimentos com o resultado dos áudios vindos do autoencoder. Em dois breves testes vistos na pasta “/modelos/”, com uma mudança acarretou em melhora de 2% de acurácia, como apresentado na tabela 1, no resultado de classificação de sentimentos. Com mais tempo, é possível fazer com que o autoencoder melhore ainda mais, e como consequência a classificação de sentimentos também melhore.

Modelo	Acurácia	F1-Score
—	0.26	0.22
Autoencoder-Segundo_tanh	0.13	0.07
Autoencoder-Terceiro_20epocas	0.15	0.08

Tabela 1: Métricas obtidas na classificação

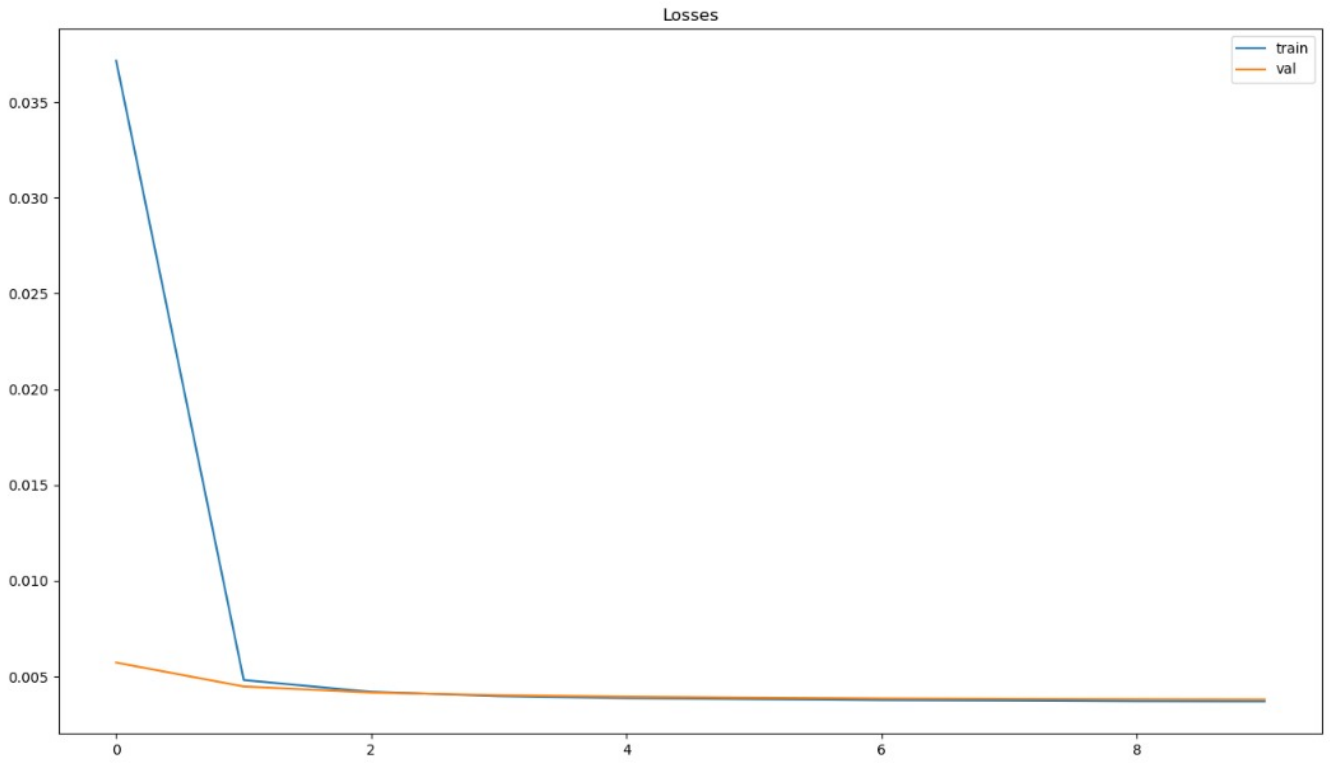

Figura 1: Loss de treinamento rede complexa.

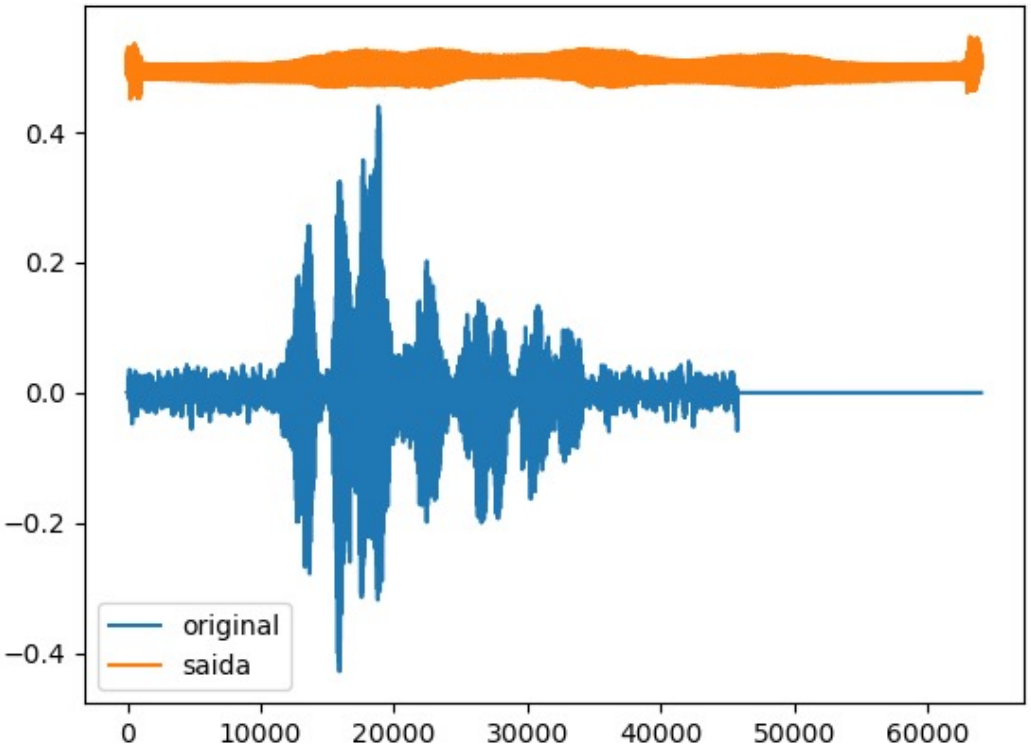

Figura 2: Sinal original e sua reconstrução pelo autoencoder complexo.

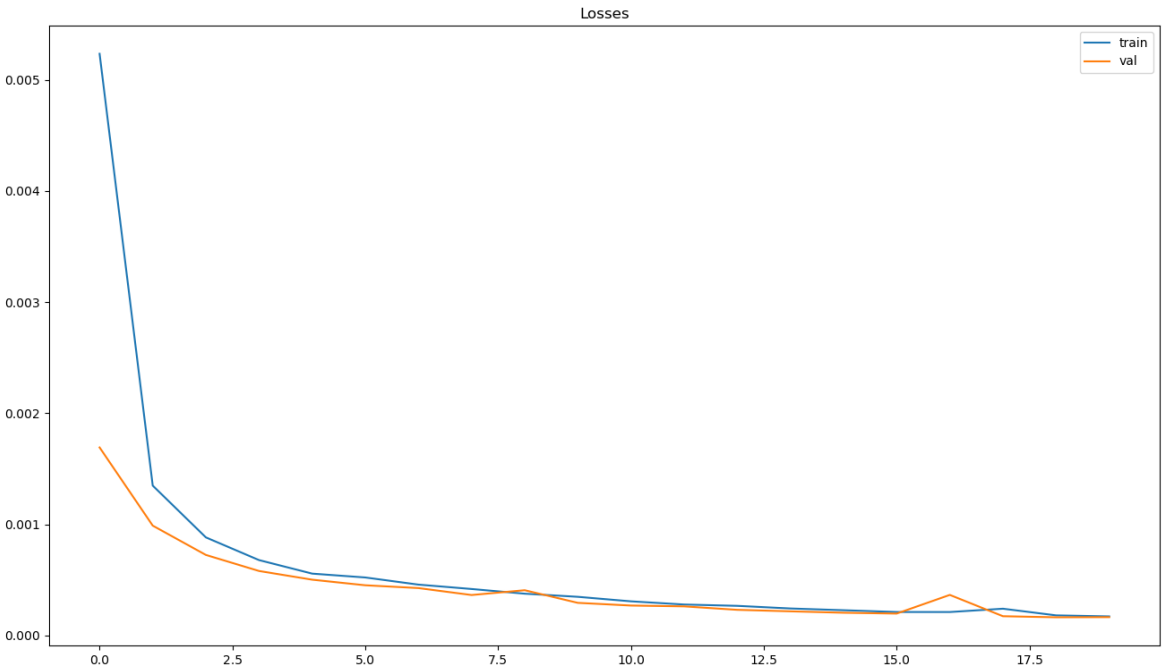

Figura 3: Curvas de Loss treinamento da rede compacta

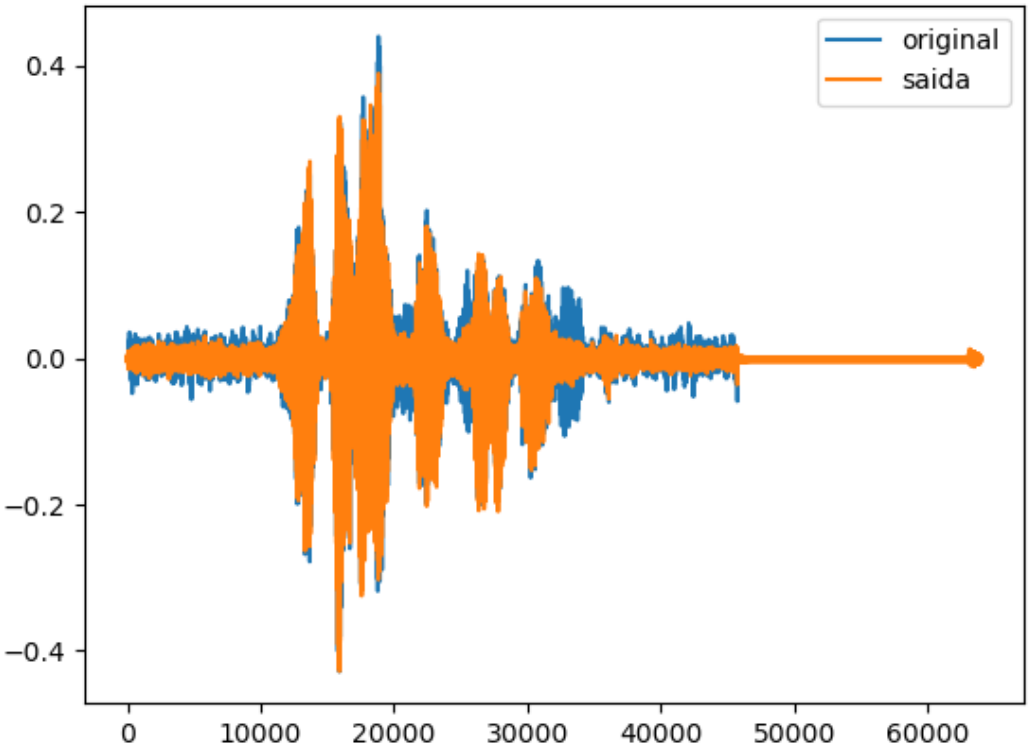

Figura 4: Reconstrução de um áudio pela rede compacta.

4- **Desafios e dificuldades**

Foi uma constante briga contra o tempo para treinar os autoencoders e achar uma rede que conseguisse reconstruir de fato os áudios.

Começamos treinando a rede complexa, com muitas camadas convolucionais e deconvolucionais, mas a rede não estava aprendendo o esperado e não sabíamos o que estava errado. Isso gerou uma busca por tentar fazer a rede aprender algo.

A primeira tentativa de melhorar foi mudar a arquitetura da rede, que foi quando surgiu a rede compacta. Mesmo diminuindo absurdamente a quantidade de camadas, ainda assim não estava aprendendo. Dessa forma, ficamos desesperados.

Em seguida, percebemos que a última camada da rede Decoder era uma não linearidade do tipo sigmoide, o que, após analisarmos de perto um áudio qualquer, percebemos que não fazia sentido, pois os áudios estão contidos no intervalo [-1, 1] e a imagem da função sigmoide está contida no intervalo [0, 1].

Após isso, removeu-se a não linearidade. Esperávamos por melhoras, no entanto ainda assim não houve melhora. Usamos a rede complexa e também não houveram alterações. Sendo assim, julgamos importante a não linearidade ao fim do processo de decodificação. Colocamos assim, a função tangente hiperbólica para executar essa função, já que sua imagem está contida justamente no intervalo dos áudios [-1, 1].

Após essa alteração, testamos a rede compacta, que funcionou perfeitamente e fez o papel de autoencoder em reconstruir o sinal.

Não foi possível testar com a rede complexa, pois o tempo da prova está em seus limites finais.

5- **Conclusão e próximos passos**

Como observamos nos resultados, o experimento não teve êxito em retirar ruídos e haver melhora na eficiência de classificação de sentimento presente na voz. No entanto, ao ouvirmos manualmente um áudio, percebemos que há uma criação de um dado sintético, ou seja, é algo extremamente positivo para o projeto de classificação de sentimentos, dada a escassez de bons datasets abertos.

Com isso, percebe-se a importância desse estudo, pois podemos incorporar o autoencoder gerado ao pipeline de dados do projeto real, ajudando a gerar novos dados, recurso essencial para o desenvolvimento dos modelos de classificação.

Para o seguimento deste estudo, convém aumentar a complexidade do autoencoder até refinarmos a ponto de extinguir o ruído presente nos datasets.

6 - **Referências bibliográficas**

THIEMANN, J.; ITO, N.; VINCENT, E. DEMAND: a collection of multi-channel recordings of acoustic noise in diverse environments. Disponível em: <https://zenodo.org/record/1227121#.W2wUVNj7TUI>. Acesso em: 22 jan. 2023.

BUSSO, C. et al. IEMOCAP: interactive emotional dyadic motion capture database. Language Resources and Evaluation, v. 42, n. 4, p. 335–359, 5 nov. 2008.

COSTANTINI, G. et al. EMOVO Corpus: an Italian Emotional Speech Database. Disponível em: <https://aclanthology.org/L14-1478/>. Acesso em: 22 jan. 2023.

CMU-MOSEI Dataset | MultiComp. Disponível em: <http://multicomp.cs.cmu.edu/resources/cmu-mosei-dataset/>. Acesso em: 22 jan. 2023.

BANK, D.; KOENIGSTEIN, N.; GIRYES, R. Autoencoders. arXiv:2003.05991 [cs, stat], 3 abr. 2021.

BALDI, P. Autoencoders, Unsupervised Learning, and Deep Architectures. v. 27, p. 37–50, 2012.