

# *Análise Comparativa de Verificação de Locutor com $k$ -NN e SVM*

- Rosângela D'Avilla



# Da Voz aos Dados

“Para que um computador entenda uma voz, ela precisa ser transformada em uma “assinatura vocal” numérica.”

Neste projeto

## MFCC

*(Mel-Frequency Cepstral Coefficients)*

Áudio A



=

Áudio B





Modelos Modernos

# MFCC

(Mel-Frequency Cepstral Coefficients)

## Espectrogramas

## Modelos Transformers

tipo de  
arquitetura

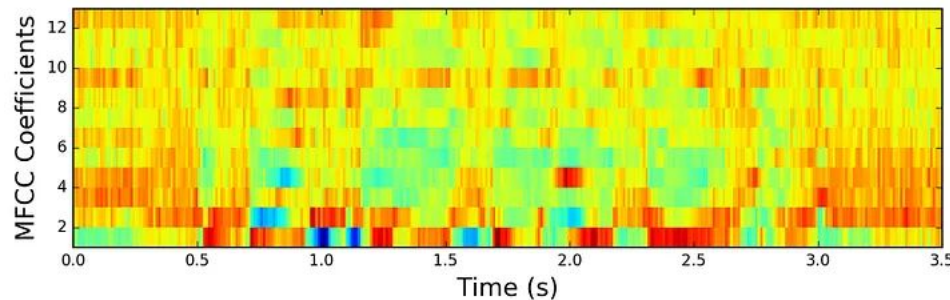


# Redes Neurais

Neste Projeto

# MFCC

(Mel-Frequency Cepstral Coefficients)





## PEAS

### **P (Desempenho)**

**Verificar se duas amostras de áudio são do mesmo locutor.**

- Medir a acurácia do sistema em acertar "Sim" ou "Não";
- Matriz de confusão.

### **E (Ambiente)**

- Pares de áudios reais, com diferentes locutores, qualidade, ruído.

### **A (Autuadores)**

- Processar um par de áudios;
- Produzir uma saída final: **1 (Sim)** ou **0 (Não)**.

### **S (Sensores)**

- Áudios brutos em WAV/MP3;
- Extração de MFCCs.

# 1. Escolha do Dataset

Version	Date	Size	Recorded Hours	Validated Hours	License	Number of Voices	Audio Format
✓ Common Voice Corpus 22.0	6/24/2025	4.74 GB	226	185	CC-0	3,759	MP3

## Common Voice

moz://a

**Dataset (Common Voice)**  
[Arquivo .tsv com metadados]  
[Pasta com arquivos .mp3]

### Splits (Age and Sex)

36% 20 - 29  
24% No information  
18% 30 - 39  
13% 40 - 49  
4% 60 - 69  
3% 50 - 59  
2% < 20  
0% 70 - 79  
0% 80 - 89  
0% 90 - 99

68% Male/Masculine  
27% No Information  
6% Female/Feminine  
0% Transgender  
0% Non-Binary  
0% Don't Wish To Say



## 2. Seleccionando Locutores Válidos

```
print(f"Filtrando locutores com no mínimo {MIN_CLIPS_PER_SPEAKER} áudios...")  
speaker_counts = df['client_id'].value_counts()  
valid_speakers = speaker_counts[speaker_counts >= MIN_CLIPS_PER_SPEAKER].index  
df_valid = df[df['client_id'].isin(valid_speakers)].copy()
```

CLIENT\_ID

Filtrando locutores com no mínimo 4 áudios...  
Encontrados 93 locutores válidos.





### 3. Criando os pares de áudios

Em vez de perguntar ao modelo "Quem é este locutor?", pensei em ensinar ele a responder uma pergunta mais simples:

**"Estes dois áudios pertencem à mesma pessoa? (Sim/Não)"**



#### Par Positivo (Label = 1)

Peguei dois áudios **diferentes** que pertencem ao **mesmo locutor**. Isso ensina ao modelo como é a "assinatura vocal" de uma pessoa, mesmo em gravações que são diferentes.

 Áudio A +  Áudio B

#### Par Negativo (Label = 0)

Peguei um áudio de um locutor e o comparei com um áudio de um **locutor diferente**, escolhido aleatório. Isso ensina ao modelo a reconhecer as diferenças entre pessoas.

 Áudio A +  Áudio C



## 4. Resultado do Pré-processamento



### Como a Tabela foi Criada

1. Para cada par de áudios, extraí os **coeficientes MFCC**, que representam a “assinatura vocal” de cada um.
2. Calculei a **diferença absoluta** entre duas matrizes MFCC (valor positivo da subtração).
3. Essa diferença virou a **feature de entrada** para o modelo.
4. A tabela final contém essas diferenças e a **label** indicando se os áudios são da **mesma pessoa (1)** ou de **pessoas diferentes (0)**.

Feature 1	Feature 2	Feature 3	...	Feature 19	Feature 20	Label
27.595	43.916	18.324	...	0.910	2.145	1
7.365	5.180	24.221	...	3.604	3.145	1
122.929	11.341	6.703	...	0.607	3.128	0





Grupo	Proporção	Nº de Amostras
Dados de Treinamento	80%	1004
Dados de Teste	20%	252

KNN

```
→ n_neighbors = 5  
→ scikit-learn (classe  
KNeighborsClassifier)
```

SVM

```
→ kernel = 'rbf'  
→ scikit-learn (classe  
SVC)
```



**Matriz de Confusão: k-NN**  
Acurácia: 82.9%

Valor Real	Diferente (Neg)	Mesmo Locutor (Pos)
	Diferente (Neg)	Mesmo Locutor (Pos)
Diferente (Neg)	VN 157	FP 30
Mesmo Locutor (Pos)	FN 13	VP 52

Valor Predito

**Matriz de Confusão: SVM**  
Acurácia: 88.5%

Valor Real	Diferente (Neg)	Mesmo Locutor (Pos)
	Diferente (Neg)	Mesmo Locutor (Pos)
Diferente (Neg)	VN 176	FP 11
Mesmo Locutor (Pos)	FN 18	VP 47

Valor Predito