Speaker Identification

Using X-Vector Embedding

DATASET

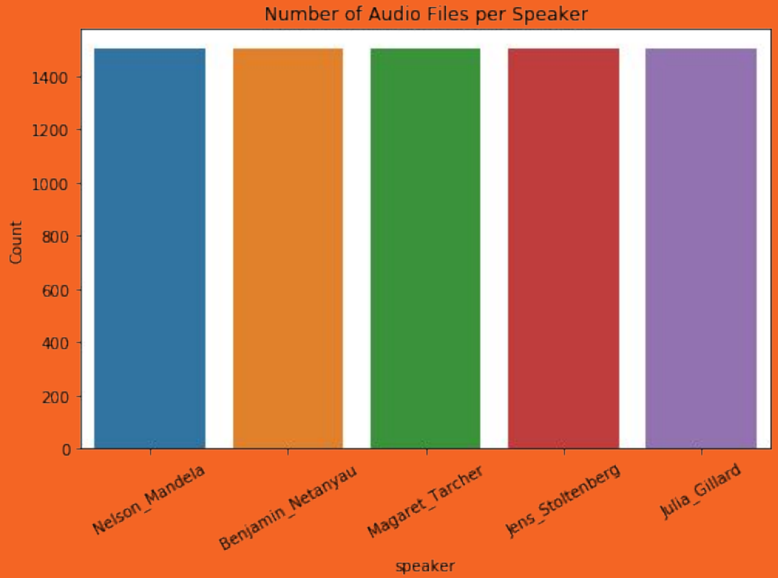
This dataset contains **speech recordings of five world leaders**, with each speaker's audio stored in separate folders:-

- **Benjamin Netanyahu**
- **Jens Stoltenberg**
- **Julia Gillard**
- **Margaret Thatcher**
- **Nelson Mandela**

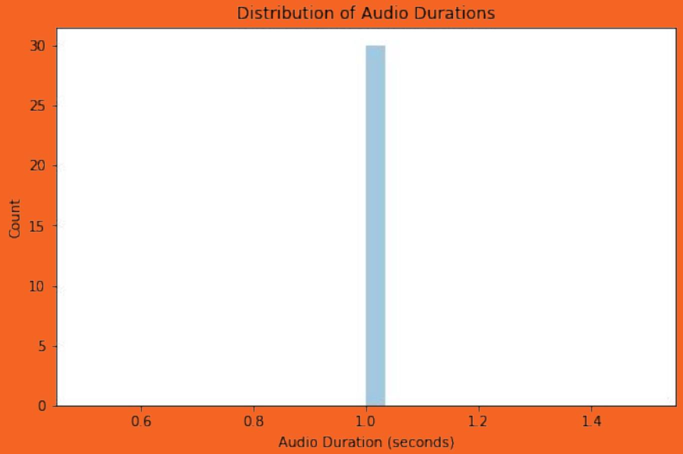
Key Features:

- **Audio Format:** PCM-encoded `.wav` files
- **Sample Rate:** 16 kHz
- **Duration:** Each audio file is **1 second long**
- **Structure:** Originally full-length speeches were (25 min) split into short **1-second chunks** (`0.wav`, `1.wav`, …, `1500.wav`) .
- **Background Noise:** A separate folder `background_noise` contains non-speech audio such as audience clapping, cheering, or laughing, which can be used for data augmentation and robustness testing

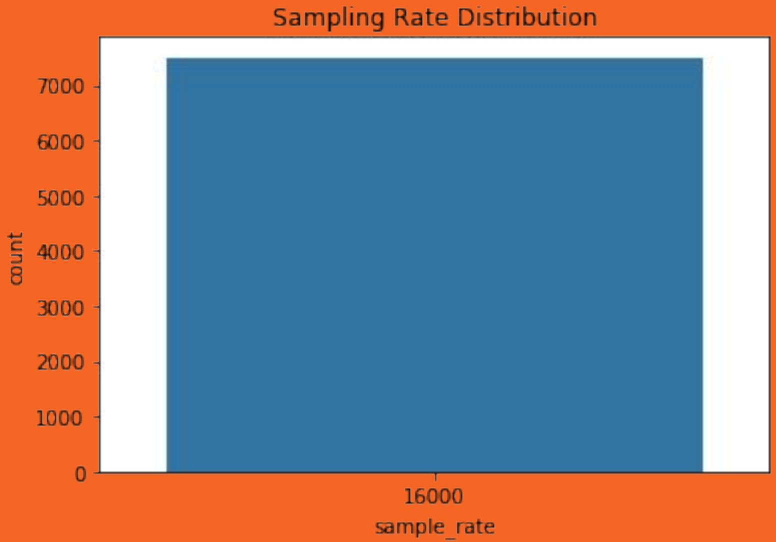
DATASET



Number of Audio Files per Speaker
Count
1400
1200
1000
800
600
400
200
0
Nelson_Mandela
Benjamin_Netanyau
Magaret_Tarcher
Jens_Stoltenberg
Julia_Gillard
speaker



Distribution of Audio Durations
Count
30
25
20
15
10
5
0
0.6
0.8
1.0
1.2
1.4
Audio Duration (seconds)



Sampling Rate Distribution
count
7000
6000
5000
4000
3000
2000
1000
0
16000
sample_rate

Feature Extraction for X-Vector

Audio Preprocessing

- Load audio at **16kHz sample rate** (standard for X-vector).
- Remove silence using energy threshold.

Step 2: Extract MFCCs

- Use `librosa.feature.mfcc()` with parameters:
  - `n_mfcc = 23` → 23 MFCC coefficients.
  - `n_fft = 512` → FFT window size(32 ms).
  - `hop_length = 160` → 10 ms stride.
  - Shape=(97,23)

Step 3: Create Fixed-Length Segments

- Segment length: **400 frames (~2.5s)**.
- Step size: **200 frames (~1.25s overlap)**.
- If audio < 400 frames → **pad with zeros**.
- Each segment shape: **(400, 23)** → ready for X-vector input.

Step 4: Build Dataset

- All 5 speakers processed.
- **1 audio file** → **1 segment** (for short audio).
- Dataset summary:
  - Total files: **7501**
  - Total segments: **7501 # all files are of >=1 sec**
  - X shape: `(7501, 400, 23) #Input to model(feature)`
  - y shape: `(7501,)# label i.e speaker ID's`
  - Balanced classes across speakers.

Dataset Splitting

Total dataset: 7,501 audio segments (from 5 speakers).

Split Ratio

- **Training:** 5789 segments (~77%)
- **Validation:** 1336 segments (~18%)
- **Test:** 376 segments (~5%)

Processing

Labels converted to **categorical (one-hot)** format.

y_train_cat: (5789, 5) #for each speaker a sample of 5 D Vector

Outcome

Balanced representation of all 5 speakers in **Train/Val/Test** sets.

Ready dataset for deep learning model training.

Training the X-Vector Model

Training Configuration

- **Dataset:** 7,501 segments (5 speakers)
- **Train / Val / Test:** 5789 / 1336 / 376
- **Input Shape:** (400 frames × 23 MFCCs)
- **Batch Size:** 64
- **Loss function:**-Categorical Cross-entropy (for multi-class classification)
- **Epochs:** 50
- **Learning Rate:** 0.001 → adaptive

So training pair
X_train: (5789, 400, 23)
y_train_cat: (5789, 5)

Training Results

- **Final Training Accuracy: 99.8%**
- **Final Validation Accuracy: 93.56%**
- **Best Validation Accuracy: 93.56% (epoch 40)**
- **Stopped at:** Epoch 34 (early stopping)

Model Architecture:

- **Input_shape = (400, 23)**
- **Frame-Level (TDNN Layers):**

Conv1D(512, kernel_size=5, dilation_rate=1) # tdnn1 #(396,512) padding =valid

Conv1D(512, kernel_size=3, dilation_rate=2) # tdnn2 #(392,512)

Conv1D(512, kernel_size=3, dilation_rate=3) # tdnn3

Conv1D(512, kernel_size=1) # tdnn4

Conv1D(1500, kernel_size=1) # tdnn5

  - 5 Conv1D layers capture temporal contex
  - Batch Normalization after each layer.

- **Statistics Pooling Layer:**Pooling: `(3000, )` (mean + std concat)
  - Computes **mean & std** over time frames → fixed-length embeddings.
- **Segment-Level Layers:**
  - Dense + Dropout layers
  - **Embedding layer:** 512-dimensional X-Vector
  - Softmax for speaker classification

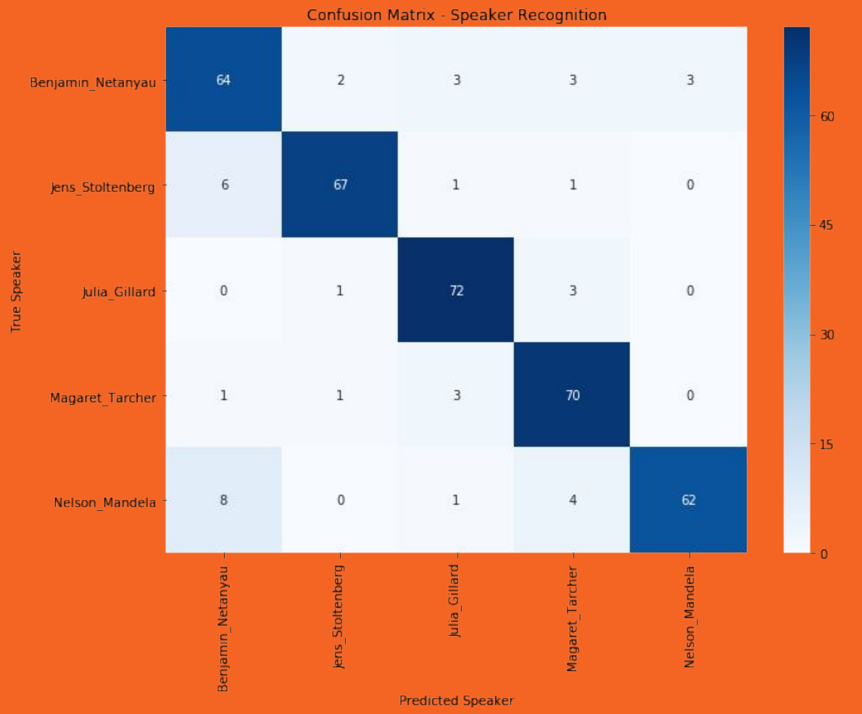
Model Evaluation on Test data

Model Evaluation

- Test Accuracy: **91%**

- Test Loss: **0.34**

- Per-speaker Accuracy:-
  - Benjamin_Netanyau: 90.67% (75 samples)
  - Jens_Stoltenberg: 90.67% (75 samples)
  - Julia_Gillard: 97.37% (76 samples)
  - Magaret_Tarcher: 89.33% (75 samples)
  - Nelson_Mandela: 88.00% (75 samples)



Speaker Prototypes (Mean Embeddings)

Loaded speaker prototypes!
Total speakers: 5

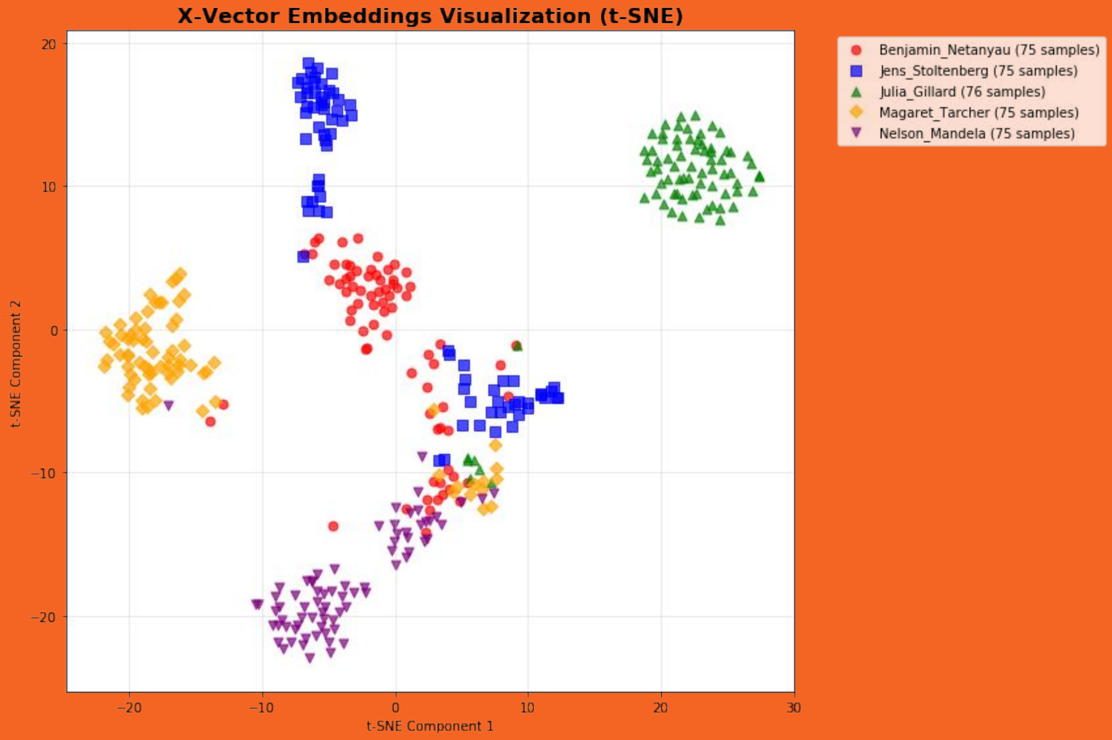
Benjamin_Netanyau → shape: (512,)
Example values: [0.02851644 0.5527304   0.14894848 0.9204143   0.135088 ……..]
Jens_Stoltenberg → shape: (512,)
Example values: [0.01786437 0.11141498 2.6988244   0.21255979 0.54119664 ……..]
Julia_Gillard → shape: (512,)
Example values: [0.35999376 2.5317929   0.01885039 0.01771628 1.8704544 ……..]
Magaret_Tarcher → shape: (512,)
Example values: [2.1499817   0.10702317 0.0085087   0.01160683 0.1295776 ……..]
Nelson_Mandela → shape: (512,)
Example values: [0.02459505 0.3224634   0.00601586 2.097342   0.01683762……..]

All prototype dimensions: [(512,), (512,), (512,), (512,), (512,)]

X-Vector Embedding Visualization & Analysis

1. Dimensionality Reduction & Visualization

- PCA reduced embeddings (512D → 50D).
- t-SNE projected into 2D for visualization.
- Clear speaker clusters observed in scatter plot.



Prediction/Inference

Audio File
↓
MFCC Extraction (n_mfcc=23)
↓
Segmentation (400 frames / 200 step)
↓
X-Vector Model (embedding layer)
↓
Average segment embeddings → 512-D vector
↓
Cosine similarity with speaker prototypes
↓
Predicted Speaker

```
audio_path = "/kaggle/input/speaker-recognition-dataset/16000_pcm_speeches/Jens_Stoltenberg/1001.wav"
predict_speaker(audio_path, speaker_prototypes)
```

```
Cosine similarities:
  Benjamin_Netanyau: 0.3121
  Jens_Stoltenberg: 0.4196
  Julia_Gillard: 0.2948
  Magaret_Tarcher: 0.3835
  Nelson_Mandela: 0.3495

Predicted Speaker → Jens_Stoltenberg (score: 0.4196)
('Jens_Stoltenberg',
 {'Benjamin_Netanyau': 0.31205845,
  'Jens_Stoltenberg': 0.41963243,
  'Julia_Gillard': 0.29484367,
  'Magaret_Tarcher': 0.3835154,
  'Nelson_Mandela': 0.34951988})
```

Audio Augmentation

Apply augmentations:

- Noise → simulate background sounds
- Echo → simulate room reflections
- Reverb → simulate large spaces

Final Prediction with Augmentation:
Speaker: Magaret_Tarcher, Confidence: 95.7%
All probabilities:
  Magaret_Tarcher: 95.7%
  Nelson_Mandela: 4.3%
  Benjamin_Netanyau: 0.0%
  Jens_Stoltenberg: 0.0%
  Julia_Gillard: 0.0%

Results:-

- Original audio: `Magaret_Tarcher` (Confidence: 99%)
- Augmented audio: `Magaret_Tarcher` (Confidence: 95.7%)
- Slight drop in confidence, but correct identification → **robust embeddings**

1. **Input Audio** (raw `.wav` file)

2. **Feature Extraction** → MFCC (23 coefficients)

3. **Segmentation** → Break audio into overlapping windows

4. **Model Prediction** → Pre-trained DNN/CNN

5. **Aggregation** → Average predictions across all segments

6. **Final Output** → Speaker nae + probabilities