MUSIC INSTRUMENT RECOGNITION USING MACHINE LEARNING

Presented by Group 1
Diya Dineep - CB.SC.U4AIE24110
VEMURI MONISHA REDDY - CB.SC.U4AIE24157

INTRODUCTION

We use machine learning to automatically identify musical instruments in audio recordings. It enhances music transcription, sound classification, and automatic tagging for streaming services. We are using Support Vector Machines (SVM) to classify instruments using extracted audio features. It is chosen for its efficiency in handling high-dimensional data and strong generalization even with limited training samples.

OBJECTIVES

1. To develop an SVM-based model for musical instrument recognition in monophonic recordings.
2. To train and test the model on monophonic instrument datasets and evaluate performance using accuracy, precision, recall and F1-score.
3. To improve classification accuracy by optimizing SVM kernel functions and hyperparameters.

METHODOLOGY

STEP 1: DATA COLLECTION

Introduction to the Dataset:

The Philharmonia Dataset is a publicly available collection of musical instrument recordings provided by the Philharmonia Orchestra. It contains high-quality, monophonic samples played by professional musicians.

Structure:

- Number of Instruments: 19
- Instrument Families: The dataset covers four major families of musical instruments:

1. Strings: Violin, Viola, Cello, Double Bass, Harp, Guitar
2. Woodwinds: Flute, Oboe, Clarinet, Bassoon, Saxophone
3. Brass: Trumpet, Trombone, French Horn, Tuba
4. Percussion: Timpani, Xylophone, Marimba, Snare Drum

STEP 2: PREPROCESSING THE AUDIO

1. Convert MP3 to WAV

- Since MP3 is compressed, we need to convert it into WAV .
- Use FFmpeg to convert MP3 to WAV.

2.Normalize Volume & Remove Silence

- If some recordings are too loud and some too soft, we normalize them to the same volume level.
- We also remove extra silence at the beginning and end of the audio files.

3.Convert Audio to Mono

- Some recordings in the Philharmonia dataset are in stereo, but instrument classification works best with mono audio.
- Converting stereo to mono simplifies processing and reduces unnecessary complexity.

4.Convert to a Uniform Sample Rate

- Sample rate is how many times per second the computer captures the sound.
- A common sample rate is 16,000 Hz or 22,050 Hz.
- We resample all audio to the same rate so they can be compared fairly.

STEP 3: FEATURE EXTRACTION

MFCC (Mel Frequency Cepstral Coefficients):

- Captures the shape of the sound, like how our ears perceive different frequencies.
- Helps in recognizing different instrument tones.
- Often used in speech and music recognition because it mimics human hearing.

Spectrograms:

- Converts sound into a visual representation (like a heatmap).
- Shows how different frequencies change over time.
- Helps in spotting patterns in musical instruments.

Spectral Centroid:

- Finds the center of mass of the sound’s frequencies.
- If the energy is in high frequencies, the sound is sharp. If it’s in low frequencies, the sound is deep and warm.
- Helps in separating bright instruments (like cymbals) from soft ones (like a bass guitar).

STEP 3: FEATURE EXTRACTION

Zero-Crossing Rate (ZCR):

- Measures how often the sound wave crosses the zero line.
- Helps in distinguishing between percussive (drums) and melodic (flute, violin) instruments.

Chroma Features:

- Identifies the musical notes being played.
- Focuses on pitch and harmony rather than just raw sound.
- Helps in recognizing instruments that play different notes.

STEP 4: TRAIN THE SVM MODEL

Preparing the Dataset:

Before training, we need to split our dataset into two parts:

- Training Set (80%) – This is used to teach the model how different instruments sound.
- Testing Set (20%) – This is used to evaluate if the model can correctly identify instruments.

Training the SVM Model:

- Kernel: Radial Basis Function (RBF). Captures complex non-linear relationships in data.
- C = 1: Regularization parameter controlling the trade-off between maximizing the margin and minimizing classification errors.
- gamma = 'scale': Kernel coefficient; automatically set to 1 / (n_features * X.var()) . Adapts the kernel width based on the data distribution.

STEP 4: TRAIN THE SVM MODEL

Implementation:

- Programming Language: Python
- Operating System
- **Key Libraries:**
- Librosa: Used for audio loading, preprocessing, and feature extraction.
- NumPy: Used for efficient numerical computations and array manipulation.
- Scikit-learn: Used for implementing the SVM classifier, data scaling, and evaluation metrics.
- Soundfile: Used for reading and writing audio files.
- Matplotlib: Used for generating visualizations of results (confusion matrix).

STEP 5: PREDICTION

- Load a new audio file (a recording of an unknown instrument).
- Extract features (MFCC, spectrogram, etc.) from this new file.
- Feed these features into the trained SVM model.
- The model predicts which instrument it is.

RESULTS

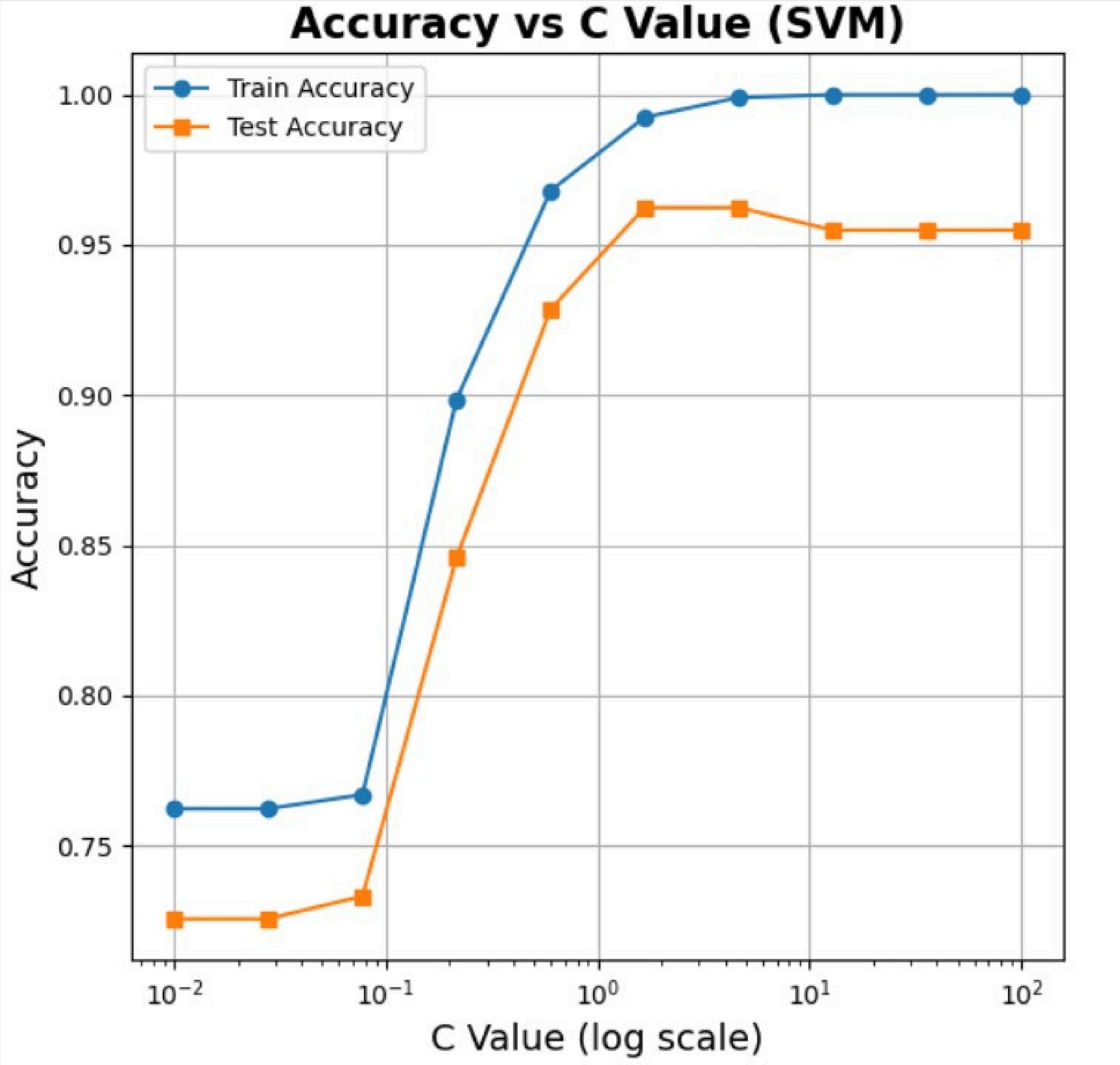
Accuracy, Precision, Recall, F1-Score

```
Accuracy: 95.11%


Overall Precision: 95.37%
Overall Recall:    95.11%
Overall F1 Score:  95.13%
```

RESULTS

Accuracy Graph



RESULTS

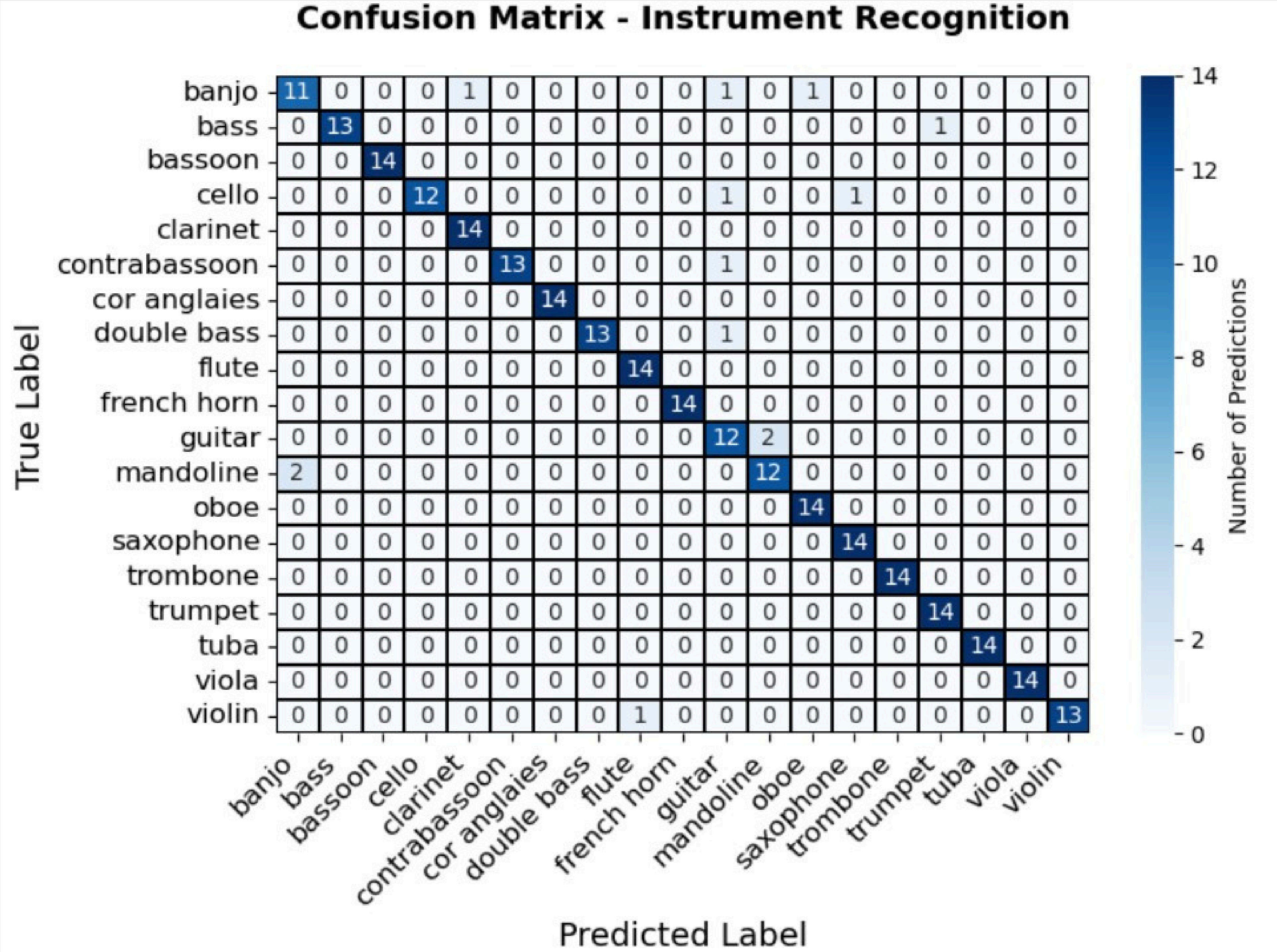
Classification Report

Classification Report (in %):

Label	Precision	Recall	F1-Score	Support
banjo	84.62	78.57	81.48	14
bass	100.00	92.86	96.30	14
bassoon	100.00	100.00	100.00	14
cello	100.00	85.71	92.31	14
clarinet	93.33	100.00	96.55	14
contrabassoon	100.00	92.86	96.30	14
cor anglaies	100.00	100.00	100.00	14
double bass	100.00	92.86	96.30	14
flute	93.33	100.00	96.55	14
french horn	100.00	100.00	100.00	14
guitar	75.00	85.71	80.00	14
mandoline	85.71	85.71	85.71	14
oboe	93.33	100.00	96.55	14
saxophone	93.33	100.00	96.55	14
trombone	100.00	100.00	100.00	14
trumpet	93.33	100.00	96.55	14
tuba	100.00	100.00	100.00	14
viola	100.00	100.00	100.00	14
violin	100.00	92.86	96.30	14
macro avg	95.37	95.11	95.13	266
weighted avg	95.37	95.11	95.13	266

RESULTS

Confusion Matrix



DISCUSSION

Strengths:

- High overall accuracy: Achieved a 95.11% classification accuracy rate.
- Effective feature extraction: Extracted key features to represent 19 instruments effectively.
- Appropriate Method: Used appropriate Support Vector Machine.
- Balanced performance: Maintained balanced precision and recall.

Weaknesses & Limitations:

- Potential to misclassify instruments: Limited the ability to classify instruments effectively.
- Challenging conditions: Performance might be changed due to the noisy environment and recording issues.
- Overfitting Possibilities: Overfitting may be a possibility due to limited training data.

CONCLUSION

The SVM-based system effectively recognizes musical instruments with 95.11% accuracy, demonstrating balanced performance through engineered features and highlighting both strengths and challenges in classifying specific instruments. This contributes to automated music analysis with applications in education and content retrieval.

FUTURE WORK

- Feature Enhancement: Add temporal & instrument-specific features for better accuracy.
- Hybrid Modeling: Combine SVM with CNNs to leverage different strengths.
- Polyphony Handling: Develop methods to recognize instruments in multi-instrument recordings.
- Real-World Applications: Focus on noise robustness and real-time use.
- Scalability: Expand instrument set and testing conditions for wider applicability.

THANK YOU