Language Recognition from Audio for Pan Indian Languages

End Project Report : CSE343 Machine Learning, Monsoon 2024, IIIT-Delhi

Ritika Thakur 2022408 | Saksham Singh 2022434 | Sarthak Gupta 2022451 | Sidhartha Garg 2022499

Abstract

India's rich linguistic diversity, with 23 official languages and countless dialects, presents unique challenges for language recognition systems. This project aims to create an advanced system for recognizing Pan Indian languages by leveraging classical machine learning techniques. Starting with an extensive dataset of audio recordings, we focus on extracting rich features like MFCCs and Spectral to capture unique acoustic features. By applying SVM and exploring different kernels, our approach demonstrates significant potential in accurately classifying a wide range of Indian languages.

Link: *GitHub Repository*

1. Introduction

In a country as diverse as India, effective language identification from audio is essential for improving communication technologies and making digital platforms more accessible in multilingual environments.

We want to develop a language recognition system for Pan Indian languages using classical machine learning techniques on a comprehensive dataset of audio recordings. By analyzing acoustic features like MFCCs and spectral properties, we aim to build models capable of accurately classifying languages from audio samples.

2. Literature Survey

- **Speech Recognition using MFCC** by S. Suksri, T. Yingthawornsuk.: This paper explores speech recognition using MFCC for feature extraction. SVM outperforms Maximum Likelihood (ML) classifiers, especially with larger MFCC samples, in recognizing spoken words.

- **Speaker Gender Recognition via MFCCs and SVMs** by Ernest Fokoue, Zichen Ma.: An algorithm involving MFCCs and SVMs is provided to perform speaker gender recognition and the RBF kernel is compared with polynomial kernel and considered as a better kernel function in this gender recognition task.

- **SVM-based audio scene classification** by Hongchen Jiang, Junmei Bai: This paper presents an approach that uses support vector machine (SVM) for audio scene classification, which classifies audio clips into one of five classes: pure speech, non-pure speech, music, environment sound, and silence.

- **Spoken Language Identification using Gaussian Mixture Model-Universal Background Model in Indian Context** by Sreedhar Potla, Vishnu Vardhan B.: The study identifies one of twenty-three Indian languages using Mel-Frequency Cepstral Coefficient (MFCC) features, showing improved accuracy with a GMM-UBM model compared to a standard GMM.

We experimented with several models, including SVM, GMM, GMM-UBM, and HMM. For each model, we evaluated performance using our best feature set as well as a subset of the features, specifically the MFCCs. This approach allowed us to compare how each model performs with different feature configurations and determine the most effective method for the task.

3. Dataset Details and Pre-processing

We have used a dataset, created by Chaitanya Bharadwaj H B on Kaggle, containing approximately 257K five-second audio samples evenly distributed across 10 Indian languages: Bengali, Gujarati, Hindi, Kannada, Malayalam, Marathi, Punjabi, Tamil, Telugu, and Urdu. The audio samples were sourced from regional videos on YouTube.

3.1. Dataset Details

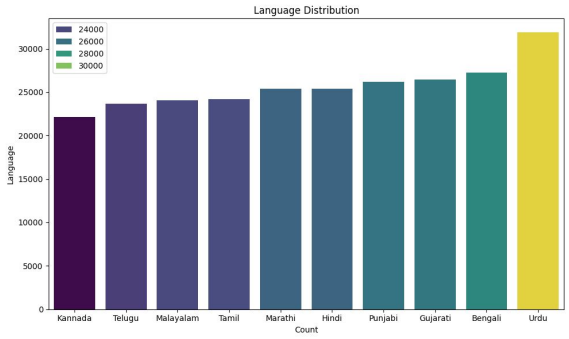

Figure 1. Classes are mostly balanced throughout the dataset

Dataset contains audio samples (around 5 seconds long) sampled at the rate of 22050 bits/sec. There are ten classes ranging from 22k samples/class to 31k samples/class. We removed class 'Punjabi' was removed due to its corrupted audio samples.

3.2. Data Pre-Processing

Data Cleaning

The files with complete or partial silent segments (less than 20 db) and files of length 0 seconds were removed, and files less or more than five seconds were padded with zeroes and trimmed down respectively. All samples were normalised and noisy samples were removed by calculating the SNR(signal to noise ratio), with a noise threshold of 0.02,

Feature Extraction

We started with three feature sets. Feature Set 1 is the most comprehensive, consisting of 15 MFCC coefficients, their delta and delta2 derivatives, along with Chroma STFT, Spectral, ZCR, RMS, and other temporal features. Feature Set 2 retains the same MFCC and Chroma STFT features as Feature Set 1 but excludes higher-order MFCC derivatives, focusing on more essential spectral and temporal components. Feature Set 3 is the most minimalistic, incorporating only 13 MFCC coefficients, Chroma STFT, Spectral, ZCR, and a single Tonnetz feature, which focuses on basic spectral structure and rhythmic patterns, offering a simpler approach to speech recognition.

Feature set-2 comprising of 25 MFCC, 25 MFCC Delta, 12 Chroma STFT and 5 Spectral Features showed the best accuracy with all the models. This feature set also contains MFCC's which work well with GMM and HMM models we try on. Hence we proceeded with feature set-2 for our project further.

4. Methodologies and Model details

4.1. Random Forest Model

Random Forest (RF) is an ensemble learning method that builds multiple decision trees during training and aggregates their predictions to improve accuracy and robustness. It operates by creating trees on randomly sampled subsets of the dataset and features, which enhances generalization and prevents overfitting. In this case, it was used as a baseline model with default hyperparameters. These default settings allowed for a straightforward comparison with more complex models.

4.2. Support Vector Machine

After preprocessing and cleaning the dataset, we performed undersampling to create a balanced subset of 45,000 files, with 5,000 files per class, to facilitate efficient grid search. Following the split into an 80:20 train-test ratio, we standardized the data to ensure uniform scaling. Among the models tested, the SVM with an RBF kernel achieved the best results on the undersampled dataset, with a **regularization parameter C=10**. To validate the robustness of these findings, we employed 5-fold cross-validation, further confirming the best performance of these hyperparameters.

4.3. Gaussian Mixture Models

Gaussian Mixture Model (GMM) is a probabilistic model that represents data as a combination of multiple Gaussian distributions, each defining a cluster through its mean and covariance. For this task, **n_components=9** was chosen to identify a small number of clusters, **covariance_type='tied'** assumes a shared covariance matrix across clusters to reduce complexity, and **reg_covar=1e-3** ensures numerical stability by regularizing the covariance matrices. A grid search was employed to fine-tune these hyperparameters, optimizing the model for clustering accuracy.

4.4. GMM-UBM

GMM-UBM extends Gaussian Mixture Models (GMM) for tasks like speaker and language recognition, where a Universal Background Model (UBM) captures general patterns from diverse datasets. The model uses **n_components=512** to handle complex distributions, **covariance_type='diag'** for computational efficiency, **reg_covar=1e-6** for detailed modeling, and **relevance_factor=16** to balance adaptation from the UBM via Maximum A Posteriori (MAP) techniques. These hyperparameters were fine-tuned using grid search to optimize recognition performance.

4.5. Hidden Markov Models Models

Hidden Markov Models (HMMs) are probabilistic models used to represent systems with hidden states that evolve over time, where each state generates observable outputs. Key components of an HMM include the start matrix (initial state distribution), transition matrix (probabilities of state changes), emission matrix (probabilities of observed outputs given a state), and the hidden states (unobservable states influencing the outputs). HMMs are widely used for time series and sequential data tasks.

A GaussianHMM models emission probabilities with Gaussian distributions, where each hidden state is associated with a unique Gaussian distribution. The n components hyperparameter specifies the number of hidden states, while covariance type determines how the covariance matrix of the Gaussian distributions is structured: **"full" (full covariance), "diag" (diagonal covariance), and "spherical" (shared variance across features).**

4.6. Multilayer Perceptron model

Multilayer Perceptron (MLP) is a type of neural network that consists of an input layer, one or more hidden layers, and an output layer. It uses forward propagation and backpropagation to learn complex patterns in the data, with each hidden layer learning to represent increasingly abstract features of the input.

In this study, we experimented with two MLP architectures for language classification, where the input consisted of **67 features**, and the output layer represented **9 possible language labels**. The first architecture had three hidden layers with 50, 30, and 20 neurons, respectively, and various activation functions (ReLU, identity, tanh, and sigmoid) applied to these layers. The Adam optimizer was used for training. The second architecture followed a similar structure with three hidden layers but used **134, 268, and 134 neurons**, with a **dropout probability of 0.3** applied after each hidden layer to prevent overfitting. **ReLU activation was applied to all hidden layers, and Adam was used for optimization.**

Performance Metrics

For each model that we train, we are generating a classification report using the scikit learn library which gives us the precision, recall, f1-score, accuracy, weighted avg and macro avg. Accuracy is used as the final metric to measure the performance of each model.

5. Result and Analysis

5.1. Support Vector Machine

We observe the RBF kernel worked the best with c = 10.

Kernel Type	Accuracy	C Value
RBF SVM	0.94	0.1
	0.97	1
	0.98	10
	0.98	100

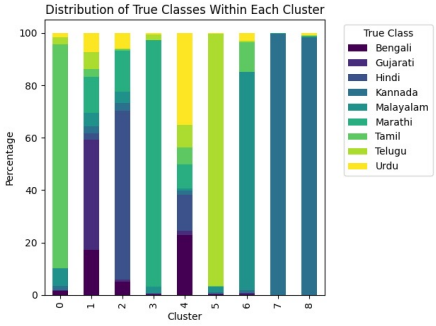

Figure 2. True Class Purity

5.2. Gaussian Mixture Models

Figure 3 shows that clusters 1, 6, and 7 have high purity, dominated by a single language (Hindi, Malayalam, Telugu). Clusters 3 and 4 show more overlap, indicating lower purity. This suggests GMM works well for languages with distinct features but struggles with languages that have similar characteristics.

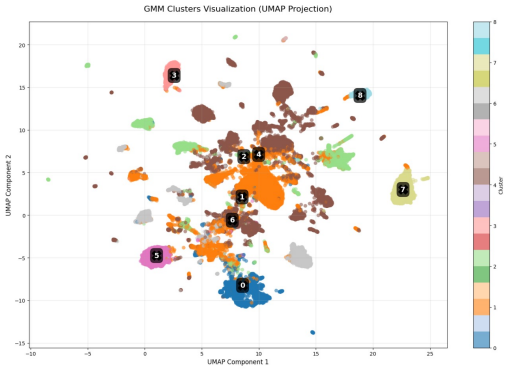

Figure 3. Cluster labeling using GMM

Figure 4 shows that clusters 0, 1, and 7 are well-separated, matching the high purity seen in Figure 3. Clusters 2, 3, and 4 overlap, indicating poor separation, particularly for languages like Gujarati and Kannada, suggesting the need for better feature differentiation.
The accuracy of the **GMM model is 59%**, while the accuracy of the **GMM-UBM model is 44%.**

Key Insights: GMM performs well for languages with distinct acoustic features but faces challenges with similar ones. Improved feature engineering or a hybrid clustering approach could enhance results.

5.3. Hidden Markov Model

n comp	full	diag	spherical
1	0.93	0.73	0.68
2	0.56	0.51	0.48
3	0.45	0.42	0.42
4	0.44	0.42	0.41

The above table is for only 25 MFCC features. When taking the best parameters from above but using all 67 features we get the accuracy of **94.97**.

5.4. Multilayer Perceptron

Architecture	ReLU	Sigmoid	Identity	Tanh
50,30,20	98.67%	98.30%	88.61%	98.40%
134,268,134	99.12%	–	–	–

5.5. Comparative Analysis of models

The results show that the Multilayer Perceptron (MLP) with three hidden layers (134, 268, 134) and a dropout rate of 0.3 outperforms all other models, achieving an accuracy of **99.12**. This suggests that MLP is highly effective for this classification task, likely due to its ability to learn complex non-linear relationships in the data. The Support Vector Machine (SVM) with an RBF kernel and a C value of 10 also performs well, with an accuracy of **99.03**, indicating its effectiveness in handling high-dimensional data.

However, the Gaussian Mixture Models (GMM) and Hidden Markov Models (HMM) underperform compared to the MLP and SVM, with the GMM model achieving only 59. accuracy and the HMM model showing a lower accuracy for smaller feature sets. The GMM struggles with languages that have similar acoustic characteristics, while the HMM, though more accurate with all features, still lags behind the MLP and SVM models. Overall, the MLP stands out as the most robust model, while the SVM also shows competitive performance, particularly when optimized with appropriate hyperparameters. Further feature engineering and model tuning may help improve the GMM and HMM results, particularly for more challenging language classifications.

6. Conclusion

Learnings

From this study, we learned several important insights that contributed to the success of our models. First, feature engineering played a crucial role, with the inclusion of MFCC, spectral features, and other derived features such as Chroma STFT and ZCR significantly improving model performance. Second, the choice of model proved essential—while MLP emerged as the most effective, traditional machine learning models such as SVM and GMM still provided valuable benchmarks and could serve as complementary methods in certain contexts. Additionally, hyperparameter tuning, especially the adjustment of the C value for SVM and architecture settings for MLP, significantly impacted performance, reinforcing the need for careful model optimization. Finally, distinguishing between languages with similar acoustic features remains a major challenge, as seen in the lower accuracy of GMM and HMM for such cases.

Challenges

Several challenges were encountered during this study. One significant hurdle was data imbalance, with certain languages having disproportionately large or small amounts of training data, which led to biased model predictions favoring the majority languages. This imbalance affected the overall classification accuracy and required strategies such as oversampling or under-sampling to mitigate its impact. Additionally, feature selection was a non-trivial task, as some features, like ZCR and Tempogram, showed minimal improvement in performance, demanding extensive analysis and experimentation to determine the optimal feature set. Computational complexity also posed a challenge, particularly when training deep learning models like MLP on large datasets, which required considerable time and resources. Lastly, the overlap in linguistic features between some languages, especially in GMM and HMM models, created difficulties in accurate classification, pointing to the need for more advanced feature extraction techniques or hybrid models to improve performance in these cases.

Contribution

- Ritika Thakur - Pre-processing, Feature Selection (Feature Set 2), Model Training (RF, Linear SVM, SVM-RBF), Report, GMM-UBM, PPT
- Saksham Singh - Pre-processing, Feature Selection (Feature Set 1), Model Training (RF, Linear SVM, SVM-RBF, SVM-Poly), Grid Search, MLP, Report, PPT
- Sarthak Gupta - Pre-processing, Feature Selection (Feature Set 3), Model Training (RF, Linear SVM, SVM-RBF, SVM-Poly), HMM, PPT
- Sidhartha Garg - Pre-processing, Feature Selection (Feature Set 1), Model Training (Linear SVM, SVM-RBF, SVM-Poly), GMM, Grid Search, Report, PPT

7. References

- S. Suksri and M. Yingthawornsuk, "Speech Recognition using MFCC," *Semantic Scholar*. Available: https://www.semanticscholar.org/paper/Speech-Recognition-using-MFCC-Suksri-Yingthawornsuk.
- P. Fokoue and W. Ma, "Speaker Gender Recognition via MFCCs and SVMs," *Semantic Scholar*. Available: https://www.semanticscholar.org/paper/Speaker-Gender-Recognition-via-MFCCs-and-SVMs-Fokoue-Ma.
- D. Jiang and H. Bai, "SVM-based audio scene classification," *Semantic Scholar*. Available: https://www.semanticscholar.org/paper/SVM-based-audio-scene-classification-Jiang-Bai.
- A. R. Mishra and N. Sharma, "Spoken Language Identification using Gaussian Mixture Model-Universal Background Model in Indian Context," *International Journal of Applied Engineering Research*, vol. 13, no. 5, pp. 4002–4008, 2018. Available: https://www.ripublication.com/ijaer18/ijaerv13n5$_7$9.*pdf*.
- C. Bharadwaj, "Audio Dataset with 10 Indian Languages," *Kaggle*. Available: https://www.kaggle.com/datasets/hbchaitanyabharadwaj/audio-dataset-with-10-indian-languages.
- *Scikit learn Documentation*, "SVM," Available: https://scikit-learn.org/1.5/modules/svm.html.
- *Librosa Documentation*, "Librosa Library Python," Available: https://librosa.org/doc/latest/index.html.
- *Scikit-learn*, "Scikit-learn for model training, metric calculations, and feature scaling," Available: https://scikit-learn.org/stable/.