Final Report

Introduction and Background

The challenge of identifying songs from hummed melodies has gained significant attention in the field of music information retrieval. Recent research has made substantial progress in this area, with various approaches yielding promising results. A semi-supervised learning method using the Covers and Hummings Aligned Dataset (CHAD) has shown improved accuracy compared to traditional machine learning models [1]. Another study demonstrated the effectiveness of deep learning applied to polyphonic databases, outperforming models trained on monophonic data [2].

Our project aims to build upon these advancements by developing a robust system for song recognition from hummed tunes. We will utilize two primary datasets: a collection of over 6,000 humming and whistling recordings covering eight songs, and the CHAD dataset, which contains more than 5,000 humming covers for over 300 songs. These datasets provide a diverse range of melodic interpretations, essential for training a versatile model.

Dataset links:

- Hums and Whistles
- CHAD Hummings

By combining these resources and implementing advanced machine learning techniques, we aim to create a system that accurately identifies songs from user-provided humming, addressing the limitations of current music recognition technologies and enhancing the user experience in music discovery and interaction.

on hummed melodies. Traditional systems like Shazam require the original recording, limiting their usefulness when users can only recall a tune. Our goal is to develop a machine learning system that can accurately identify songs from hummed input, overcoming challenges such as variations in pitch, tempo, and vocal quality.

Methods

We implemented a Convolutional Neural Network, CNN, as our supervised learning method for the project. We chose to use CNN as our supervised method because our plan was to convert each song into sine waves and have our model identify the song through its sine wave representation. Hence, using CNN could identify songs through its sine wave visual. In order to preprocess the data, we first had to choose 5 songs for data that our model will be able to detect after training. These songs were Let It Go, Happy Birthday, Here Comes the Sun, Party Rock, and I Got a Feeling. For preprocessing, we did three methods. These were cropping the training data into the 3 most popular 10 second segments, converting the audio into sine waves, and then creating variations of these audio files. First, we cropped each song into the 3 most popular 10 second segments and hummed all these segments. After humming, we had to convert each audio files to ensure consistent digital audio segments. We did this by converting the hummed audio files into sine waves as this would ensure that the audio files would be in the simplest form and our CNN would only focus on the melodic part of the songs. Finally, we created variations of each audio clip by speeding up, slowing down, altering the pitch speed, and rhythm of the hummed parts and converted those variations into sine waves. The reason for creating many versions for these audio segments was to take into account

data into the CNN.

Results and Discussion

```
Image: HBDJosh1.png → HappyB
Image: HBDJosh2.png → HappyB
Image: HarishBirthday 1.png → HappyB
Image: HarishFrozen1 1.png → Frozen
Image: HarishSun1 1.png → Sun
Image: LetItGoJosh1.png → Frozen
Image: LetItGoJosh2.png → Frozen
Image: RonitFrozen3 1.png → Frozen
Image: RonitHappy3 1.png → HappyB
Image: RonitHappy4.png → HappyB
Image: RonitParty2.png → PartyRock
Image: RonitParty3.png → PartyRock
Image: RonitSun2.png → Sun
Image: RonitSun3 1.png → Sun
Image: SunJosh1.png → HappyB (True: Sun)
Image: SunJosh2.png → Sun
Image: TestHappy.png → HappyB
Image: TestParty.png → PartyRock
Image: TestPartyJosh1.png → PartyRock
Image: TestPartyJosh2.png → PartyRock
Image: TestSun.png → Sun

Test Accuracy: 95.2%
```

Figure 1: Accuracy of our CNN without PCA

```
Training Loss (Cross Entropy): 0.0733
Validation Loss (Cross Entropy): 0.0052
Test Loss (Cross Entropy): 0.0057
```

Figure 2: Quantitative Metrics for CNN without PCA

This approach involves preprocessing the training data for a music recognition system by converting train audio into sine waves and applying augmentation techniques like speed and pitch variations, as well as normalization, to enrich the dataset. This meticulous preprocessing aims to capture the fundamental audio characteristics needed for accurate melody recognition. By avoiding Principal Component Analysis (PCA), all audio features are preserved, which enhances the precision of the recognition model, allowing it to effectively differentiate between similar melodies. The final model, a Convolutional Neural Network (CNN) without PCA, achieved an impressive training accuracy of 98.70%, a perfect validation accuracy of 100.00%, and a test accuracy of 95.24%, demonstrating its robustness and reliability in recognizing hummed melodies. We also implemented the use of a Random Forest algorithm for processing and predicting outcomes based on preprocessed train data. In this approach, train audio is first converted into sine waves, and then augmented through speed, pitch, and variation adjustments, followed by normalization. The Random Forest algorithm is chosen for its ability to handle complex datasets robustly by combining multiple decision trees, thereby enhancing the model's precision and reducing overfitting. This makes it particularly effective for managing the intricacies of audio data. However, after thoroughly training and testing the results, we discovered that the CNN model without PCA outperformed the Random Forest model in terms of both accuracy and efficiency. Consequently, we opted for the CNN model without PCA.

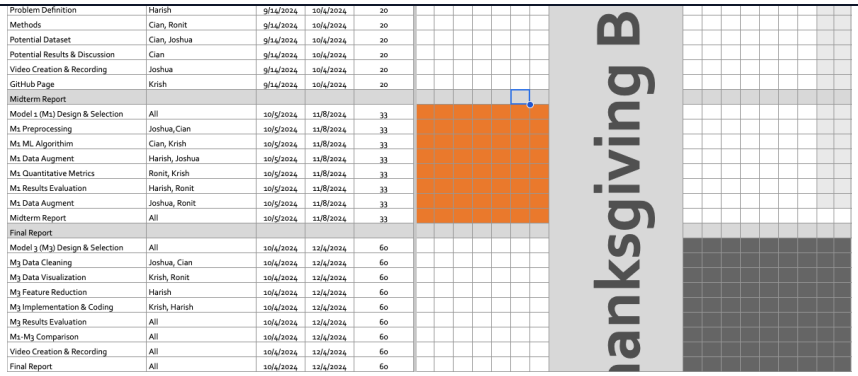
Fine-tuning the model with additional diverse datasets can help it better generalize across different humming styles and conditions. Exploring alternative machine learning algorithms or neural network architectures that might capture the nuances of hummed melodies more effectively is also beneficial. Improving the preprocessing steps to incorporate variations in speed and rhythm can further refine the input data quality. Utilizing transfer learning by leveraging pre-trained CNN models, such as those trained on ImageNet or Inception ResNet, and fine-tuning them for specific tasks can significantly enhance performance, especially when training data is limited. Additionally, incorporating temporal information through the use of recurrent neural networks (RNNs) or long short-term memory (LSTM) networks, in combination with CNNs, can better capture the sequential nature of melodies, potentially leading to more precise recognition outcomes.

References

[1] A. Amatov et al., A SEMI-SUPERVISED DEEP LEARNING APPROACH TO DATASET COLLECTION FOR QUERY-BY-HUMMING TASK.

[2] Sun, J., & Lee, S.-P. (n.d.). (rep.). Query by Singing/Humming System Based on Deep Learning (Vol. 12). International Journal of Applied Engineering Research.

[3] B. L. Pham, H. H. Huong Hoang Luong, T. P. Tran, H. P. Ngo, H. V. Nguyen, and T. Thinh, "An Approach to Hummed-tune and Song Sequences Matching," An Approach to Hummed-tune and Song Sequences Matching, Nov. 20, 2020. https://link.springer.com/chapter/10.1007/978-981-19-8069-5_49 (accessed Oct. 02, 2024).



Problem Definition	Harish	9/14/2024	10/4/2024	20
Methods	Cian, Ronit	9/14/2024	10/4/2024	20
Potential Dataset	Cian, Joshua	9/14/2024	10/4/2024	20
Potential Results & Discussion	Cian	9/14/2024	10/4/2024	20
Video Creation & Recording	Joshua	9/14/2024	10/4/2024	20
GitHub Page	Krish	9/14/2024	10/4/2024	20
Midterm Report				
Model 1 (M_1) Design & Selection	All	10/5/2024	11/8/2024	33
M_1 Preprocessing	Joshua,Cian	10/5/2024	11/8/2024	33
M_1 ML Algorithim	Cian, Krish	10/5/2024	11/8/2024	33
M_1 Data Augment	Harish, Joshua	10/5/2024	11/8/2024	33
M_1 Quantitative Metrics	Ronit, Krish	10/5/2024	11/8/2024	33
M_1 Results Evaluation	Harish, Ronit	10/5/2024	11/8/2024	33
M_1 Data Augment	Joshua, Ronit	10/5/2024	11/8/2024	33
Midterm Report	All	10/5/2024	11/8/2024	33
Final Report				
Model 3 (M_3) Design & Selection	All	10/4/2024	12/4/2024	60
M_3 Data Cleaning	Joshua, Cian	10/4/2024	12/4/2024	60
M_3 Data Visualization	Krish, Ronit	10/4/2024	12/4/2024	60
M_3 Feature Reduction	Harish	10/4/2024	12/4/2024	60
M_3 Implementation & Coding	Krish, Harish	10/4/2024	12/4/2024	60
M_3 Results Evaluation	All	10/4/2024	12/4/2024	60
M_1-M_3 Comparison	All	10/4/2024	12/4/2024	60
Video Creation & Recording	All	10/4/2024	12/4/2024	60
Final Report	All	10/4/2024	12/4/2024	60

Contribution Chart

Name	Proposal Contributions
Cian Thomas	Preprocessing, Quantitative Metrics, Model Testing, Optimizing
Krish Sharma	Implemented CNN without PCA
Ronit Sehgal	Implemented Random Forest
Joshua George	Added PCA to our model
Harish Tirumalai	Website, Presentation, Data creation