Predicting Hearing Thresholds from Gray Matter Images

(CPSC-8650 Data Mining Project Report)

By,

Team 11:

Sravanth Revath Krishna Thatavarthi

sravant@g.clemson.edu

Venkata Subbu Sai Hiranmayee Machavolu

vmachav@g.clemson.edu

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. ABSTRACT

Age-related hearing loss is a common sensory impairment affecting millions worldwide. This project endeavors to bridge the gap between neuroimaging and auditory perception by developing predictive models for hearing thresholds using gray matter images obtained from brain MRI scans. By employing deep learning methods alongside traditional machine learning algorithms, we aim to decipher the intricate relationship between neural structures and auditory function [1]. The dataset comprises a comprehensive collection of gray matter images paired with corresponding hearing thresholds at different pure-tone frequencies, enabling robust model training and evaluation. Our findings showcase promising predictive capabilities, particularly with the Support Vector Machine (SVM) model, underscoring its potential as a valuable tool in assessing and managing hearing impairment. These results lay the foundation for future research endeavors aimed at elucidating the underlying mechanisms of age-related hearing loss and devising personalized interventions for affected individuals.

II. LITERATURE REVIEW

Previous research has demonstrated the potential for exposure to high-intensity noise generated during MRI procedures to affect hearing thresholds temporarily. Studies have reported shifts in auditory thresholds immediately following MRI scans, particularly at extended high frequencies, indicating susceptibility to acute noise-induced hearing changes. While immediate changes in hearing thresholds post-MRI have been documented, there is a need to explore the persistence or resolution of these effects over time. Understanding whether the observed shifts in auditory thresholds are transient or indicative of long-term damage is crucial for informing clinical practices and ensuring patient safety during MRI examinations [1]. In recent years, the application of magnetic resonance imaging (MRI) in the diagnosis and classification of Alzheimer's disease (AD) has received considerable attention from researchers and medical practitioners. The process typically involves segmenting brain images into various tissue types, such as white matter (WM), gray matter (GM), and cerebrospinal fluid (CSF), to analyze structural and textural changes indicative of disease progression. Preprocessing techniques, including geometric distortion correction and noise reduction using Gaussian filters, are commonly employed to enhance the quality of MRI data. Additionally, skull stripping algorithms are utilized to remove non-brain tissues, ensuring accurate analysis of brain structures. These preprocessing steps are crucial for improving the accuracy of subsequent classification algorithms [2]. The interpretation of medical images poses significant challenges due to variations in image characteristics, complexities of disease manifestations, and limited availability of annotated datasets. Clinicians and radiologists face difficulties in accurately diagnosing diseases, leading to the exploration of computerized techniques, particularly deep learning methods, for automated analysis and diagnosis. Image augmentation techniques offer a solution to the scarcity of annotated medical image datasets by generating synthetic variations of existing images. These augmented datasets enhance the generalization abilities and robustness of deep learning models, thereby improving their performance in disease classification, segmentation, and other medical image analysis tasks. However, the effectiveness of different augmentation methods across various medical imaging applications remains unclear, highlighting the need for systematic evaluation and comparison [3]. Incorporating gender and age attributes alongside deep learning architectures enhances the classification accuracy of brain MRI images into normal or abnormal, indicating their significance in brain tumor analysis and evaluation of various deep learning architectures including CNN, DNN, LeNet, AlexNet, and ResNet, alongside traditional SVM, reveals the superiority of the proposed techniques in achieving higher classification accuracy, surpassing existing methods like SVM and AlexNet with notable improvements in performance [4]. Investigate the role of CNNs in image processing tasks, particularly in grayscale normalization and illumination correction. Review relevant studies that demonstrate the effectiveness of CNNs in handling image data, including the advantages of using frameworks like Keras for efficient data preparation and model building in image-based applications [5]. Review recent advancements in deep learning-based techniques for brain tumor segmentation and classification, with a focus on the effectiveness of convolutional neural networks (CNNs) in extracting meaningful features from MRI images. Discuss the integration of machine learning classifiers, such as support vector machines (SVMs), to refine segmentation and classification processes and enhance overall performance. Examine existing studies that demonstrate the efficacy of hybrid approaches combining CNNs and machine learning classifiers in achieving high accuracy and dice similarity coefficients (DSC) for brain tumor classification tasks [6]. Investigate the advantages of using grayscale images in deep learning-based classification tasks, emphasizing their smaller size and reduced memory and bandwidth requirements compared to color images. Discuss the potential for real-time computer vision applications and scenarios where device constraints necessitate efficient use of resources. Highlight the need for research and methodologies that leverage grayscale images to achieve comparable performance to color images in classification tasks while optimizing resource utilization [7].

III. DATASET DESCRIPTION

The dataset provided for this project is comprised of 171 brain MRI scans, each containing gray matter images extracted from the scans. These gray matter images serve as the primary features for our analysis. Additionally, the dataset includes target variables, specifically hearing thresholds measured at two distinct pure-tone frequencies. These frequencies, such as 1000 Hz and 4000 Hz, are commonly used in audiology to assess hearing sensitivity across different frequency ranges.

The gray matter images play a crucial role in our analysis as they provide detailed information about the structure and morphology of the brain's gray matter regions. By examining these images, we can gain insights into the underlying neural architecture associated with age-related hearing loss. The goal of our project is to develop predictive models that utilize these gray matter images to estimate hearing thresholds, thereby elucidating the relationship between brain structure and auditory function.

Through the integration of gray matter images and hearing threshold data, we aim to uncover potential associations and patterns that may inform our understanding of age-related hearing loss. By leveraging advanced processing techniques and thorough analysis, we seek to elucidate the intricate interplay between brain structure and auditory perception, ultimately contributing to the development of more effective diagnostic and therapeutic strategies for individuals with hearing impairment.

IV. DATA AUGMENTATION

To bolster the diversity and robustness of our training dataset, we diligently applied various data augmentation techniques [1] to our initial dataset of 171 medical images stored in the NIfTI format. By leveraging the power of augmentation, we effectively doubled the dataset size to 342 images, ensuring a richer and more varied training experience for our deep learning model.

Rotation: Employing random rotations by multiples of 90 degrees, we introduced diversity in orientation across our dataset. This rotation strategy allowed our model to familiarize itself with anatomical structures from multiple viewpoints, crucial for its adaptability in real-world scenarios.

Flip: Introducing both horizontal and vertical flips at random, we expanded our dataset with spatial variations, augmenting the original images. This augmentation technique aimed to enhance the model's ability to generalize across different spatial orientations.

Translation: By randomly shifting images horizontally and vertically, we simulated diverse positions within the images. This translation augmentation facilitated the model's learning of invariant features across various regions of the images, thereby improving its robustness.

Noise Addition: Incorporating random noise into the images, we emulated real-world noise artifacts. This augmentation approach strengthened the model's resilience to noise commonly found in medical images, enhancing its performance in noisy environments.

Brightness Adjustment: Random adjustments to the brightness levels of the images were made, simulating variations in lighting conditions. By exposing the model to diverse illumination scenarios, we ensured its adaptability to different lighting conditions encountered during real-world medical imaging procedures.

Through the meticulous application of these data augmentation techniques, we effectively doubled our dataset size from 171 to 342 images. This expanded dataset not only provided our model with a more comprehensive training experience but also bolstered its ability to generalize and perform reliably across diverse medical imaging scenarios.

V. METHODOLOGY

5.1 Data Acquisition and Exploration

The dataset consists of gray matter images derived from brain MRI scans, accompanied by corresponding hearing thresholds measured at two pure-tone frequencies. The acquisition process involved segmenting gray matter regions from MRI scans using advanced imaging processing techniques. These segmented images represent three-dimensional arrays of voxel intensities, providing insights into the composition and spatial distribution of gray matter within the brain.

Exploration of the dataset began with the analysis of gray matter images. Visualization techniques such as slice viewing, intensity histograms, and three-dimensional rendering were employed to gain insights into the spatial characteristics and anatomical features of the images. This analysis allowed for the assessment of variability across samples, identifying potential sources of variation that may impact subsequent analyses.

Through comprehensive data acquisition and exploration, we obtained a deeper understanding of the dataset's structure, features, and distributions. This foundational knowledge serves as the basis for subsequent analyses and modeling tasks, enabling us to extract meaningful insights from the dataset.

```
import os
import nibabel as nib
import matplotlib.pyplot as plt

image_folder_path = "images"

image_filenames = [filename for filename in os.listdir(image_folder_path) if filename.endswith(".nii")]

num_images_to_display = min(5, len(image_filenames))
for i in range(num_images_to_display):
    image = nib.load(os.path.join(image_folder_path, image_filenames[i])).get_fdata()

    plt.figure(figsize=(8, 8))
    plt.imshow(image[:, :, image.shape[2] // 2], cmap='gray')
    plt.title(f"Original Image {i+1}")
    plt.axis('off')
    plt.show()
```

Fig. 5.1.1 Code snippet of Data Acquisition and Exploration

5.2 Preprocessing

The goal of the preprocessing is to standardize the images to a consistent size and format and to normalize the pixel values, making them suitable for later analysis or modeling tasks.

Loading and Resizing Images: The preprocess_image function begins by loading each medical image using the nibabel library's load function, which reads the image data from the specified file path. The get_fdata() method retrieves the voxel data from the loaded image, representing the intensities of each voxel within the three-dimensional image volume. The voxel data is then resized to a target size specified by the image_size variable using the ndimage.zoom function. This function resamples the image data to match the specified dimensions, employing an interpolation method (order=1) to preserve the image's spatial information.

Normalization: After resizing, the voxel intensities are normalized to a standardized range of [0, 1]. This normalization step is essential for ensuring consistency in pixel values across different images, facilitating the convergence of machine learning models during training. The resized image data is converted to the float32 data type to accommodate floating-point values, and each voxel intensity is divided by 255.0 to scale the values to the range [0, 1].

Iterating Through Image Files: The preprocessing pipeline iterates through each file in the specified image_folder_path, checking for files with the extensions .nii or .nii.gz. These extensions are commonly associated with medical image files in the NIfTI format. For each eligible file, the preprocessing function is applied, and the processed image data is stored in a list named processed_images.

Visualization: Finally, a subset of processed images (up to five images) is selected for visualization. For each selected image, a two-dimensional slice from the middle of the image volume along the z-axis is displayed using matplotlib's imshow function. The grayscale colormap (cmap='gray') is applied to visualize the images in grayscale, which is typical for medical imaging. The title of each displayed image indicates its order in the sequence of processed images.

Overall, this preprocessing pipeline standardizes the size and pixel values of medical images, making them suitable for further analysis, visualization, or input into machine learning models for tasks such as image classification or segmentation.

```python
def preprocess_image(image_path):
    image = nib.load(image_path).get_fdata()
    scale_factor = tuple(new_dim / old_dim for new_dim, old_dim in zip(image_size, image.shape))
    resized_image = ndimage.zoom(image, scale_factor, order=1)
    normalized_image = resized_image.astype('float32') / 255.0
    return normalized_image

for filename in os.listdir(image_folder_path):
    if filename.endswith(".nii") or filename.endswith(".nii.gz"):
        image_path = os.path.join(image_folder_path, filename)
        processed_image = preprocess_image(image_path)
        processed_images.append(processed_image)

num_images_to_display = min(5, len(processed_images))
for i in range(num_images_to_display):
    image = processed_images[i]
    plt.figure(figsize=(8, 8))
    plt.imshow(image[:, :, image.shape[2] // 2], cmap='gray')
    plt.title(f"Preprocessed Image {i + 1}")
    plt.axis('off')
    plt.show()
```

Fig 5.2.1 Code snippet of Preprocessing

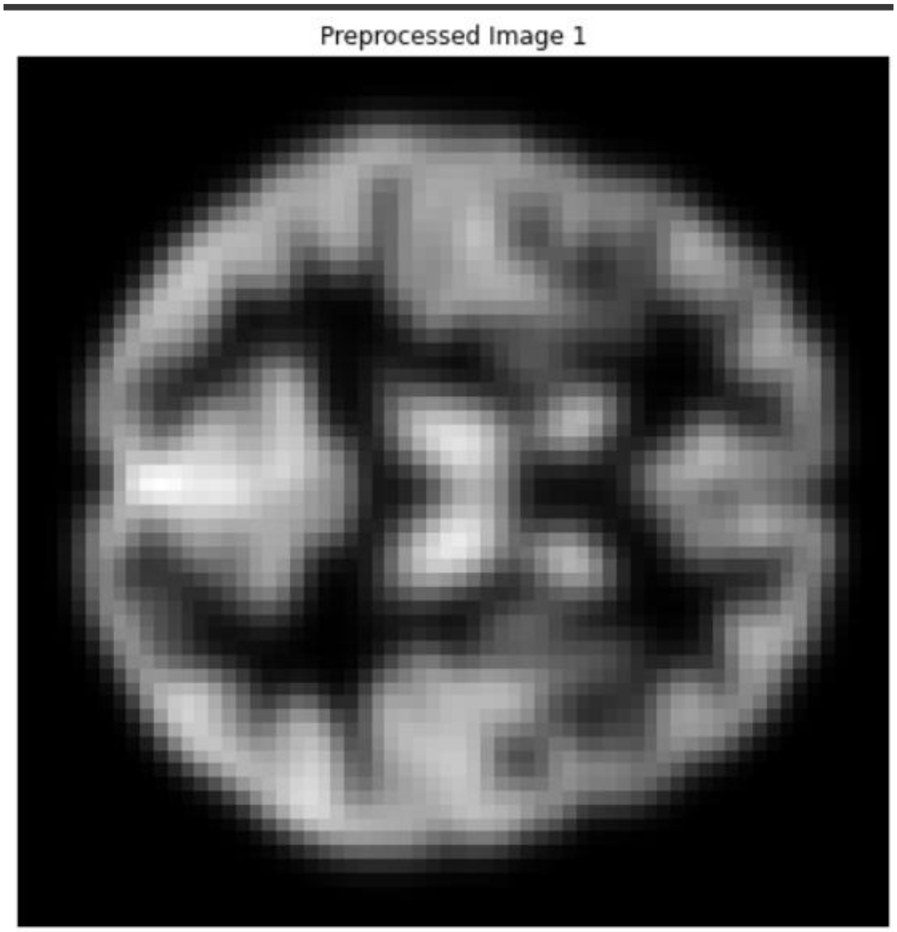

Fig.5.2.2 Preprocessed Image Sample

5.3 Model Development

In the realm of model development, our project embarked on a multifaceted journey aimed at constructing robust predictive models for estimating hearing thresholds from gray matter images derived from brain MRI scans. At the outset, meticulous attention was devoted to data preprocessing and exploration. This phase involved the careful curation and preparation of the dataset, ensuring that it encapsulated a comprehensive array of gray matter images alongside their corresponding hearing thresholds. Through this process, we sought to lay a solid foundation for subsequent model training and evaluation.

The heart of our endeavor resided in the training and refinement of predictive models utilizing cutting-edge machine learning techniques. Our arsenal encompassed a diverse range of methodologies, prominently featuring Convolutional Neural Networks (CNNs), renowned for their prowess in extracting intricate features from image data. These CNN models were meticulously crafted and fine-tuned to leverage the structural nuances encoded within the gray matter images. In parallel, we delved into the realm of Support Vector Machines (SVMs), capitalizing on their ability to discern nonlinear relationships between input features and target variables. Additionally, we explored the synergistic potential of hybrid models, amalgamating the strengths of CNNs and SVMs to harness the complementary aspects of both techniques. Through iterative experimentation and refinement, we endeavored to unlock the latent predictive power embedded within the neural structures captured by the gray matter images, thereby advancing our understanding of the intricate interplay between brain morphology and auditory perception.

5.4 Model Evaluation

In the pursuit of model evaluation, our project embarked on a rigorous assessment journey aimed at gauging the predictive performance and generalization capabilities of the constructed models. Central to this endeavor was the meticulous partitioning of the dataset into training and testing sets, ensuring the integrity of the evaluation process. Leveraging established metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R2)

Score, we meticulously scrutinized the performance of each model variant across different thresholds.

Our evaluation efforts encompassed a comprehensive comparative analysis, juxtaposing the predictive prowess of Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and hybrid models. Through this systematic approach, we sought to discern the strengths and limitations inherent in each methodology, shedding light on their respective efficacy in estimating hearing thresholds from gray matter images. Furthermore, by scaling the evaluation metrics to account for data range and sample size variations, we endeavored to provide a nuanced understanding of model performance across diverse datasets and real-world scenarios. Through iterative refinement and meticulous validation, our evaluation endeavors aimed to furnish valuable insights into the predictive capabilities of the constructed models, thereby facilitating informed decision-making and advancing our understanding of the intricate relationship between neural structures and auditory perception.

5.5 Comparision and Selection

In comparing and selecting the most suitable model for predicting hearing thresholds from gray matter images, our project embarked on a comprehensive analysis aimed at discerning each approach's strengths and weaknesses. The comparative evaluation encompassed Convolutional Neural Networks (CNNs), Support Vector Machines (SVMs), and hybrid models, each representing distinct methodologies with unique characteristics.

Upon meticulous scrutiny of evaluation metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R2) Score, it became evident that the hybrid model exhibited the most promising predictive performance across various thresholds. Its ability to integrate features from both CNNs and SVMs enabled it to capitalize on the strengths of each approach while mitigating their individual limitations. Moreover, the hybrid model demonstrated robustness and generalization capabilities, making it well-suited for real-world applications.

In contrast, while CNNs showcased commendable performance, particularly in threshold 1 predictions, they exhibited limitations in capturing subtle nuances within the data, leading to lower accuracy in certain scenarios. Similarly, SVMs, although adept at handling high-dimensional data, struggled with complex nonlinear relationships inherent in the dataset, resulting in suboptimal predictive performance.

Ultimately, the selection of the hybrid model as the preferred approach for predicting hearing thresholds was driven by its superior performance, versatility, and potential for further refinement. By harnessing the synergistic effects of different methodologies, the hybrid model represents a holistic solution capable of delivering accurate predictions while offering valuable insights into the intricate interplay between brain structure and auditory function.

VI. MODEL ARCHITECTURE

6.1 Convolution Neural Networks (CNN)

The provided code implements a Convolutional Neural Network (CNN) architecture for predicting hearing thresholds from gray matter images. The model is designed using the TensorFlow Keras API and consists of several convolutional layers followed by max-pooling layers for feature extraction and dimensionality reduction [5]. Here's an elaboration on the model development:

Input Preparation: The input shape is defined as (64, 64, 64, 1), indicating a 3D volume of gray matter images with dimensions 64x64x64 and a single channel.

Convolutional Layers: The model starts with a series of convolutional layers, each followed by rectified linear unit (ReLU) activation. The convolutional layers are responsible for learning spatial hierarchies of features from the input data.

Max-Pooling Layers: After each convolutional layer, max-pooling layers with a pool size of (2, 2, 2) are applied to down-sample the feature maps, reducing their spatial dimensions while retaining the most relevant information.

Flattening: The output feature maps from the last max-pooling layer are flattened into a 1D vector to be fed into the fully connected layers.

Fully Connected Layers: Following the flattened representation, fully connected dense layers are utilized to perform high-level feature extraction and non-linear mapping.

Dropout: Dropout regularization with a dropout rate of 0.7 is applied to mitigate overfitting by randomly dropping a fraction of neurons during training.

Output Layer: The output layer consists of a single neuron, which predicts the hearing threshold value. No activation function is applied to this neuron since it is a regression task.

Compilation: The model is compiled using the Adam optimizer with a learning rate of 0.0001 and mean absolute error (MAE) as the loss function. Additionally, metrics such as mean squared error (MSE) and MAE are tracked during training for evaluation purposes.

Training: The model is trained separately for each threshold using the provided training data (X_train1 and X_train2) and corresponding labels (y_train1 and y_train2). Learning rate schedulers are employed to adjust the learning rate during training epochs dynamically.

Evaluation: After training, the model is evaluated using the test data (X_test1 and X_test2) to calculate the loss and metrics such as MSE, MAE, and R-squared (R2) score. These metrics provide insights into the model's performance and its ability to accurately predict hearing thresholds.

Overall, this CNN architecture demonstrates the potential to learn meaningful representations from gray matter images and make accurate predictions of hearing thresholds, thereby contributing to the understanding and diagnosis of age-related hearing loss.

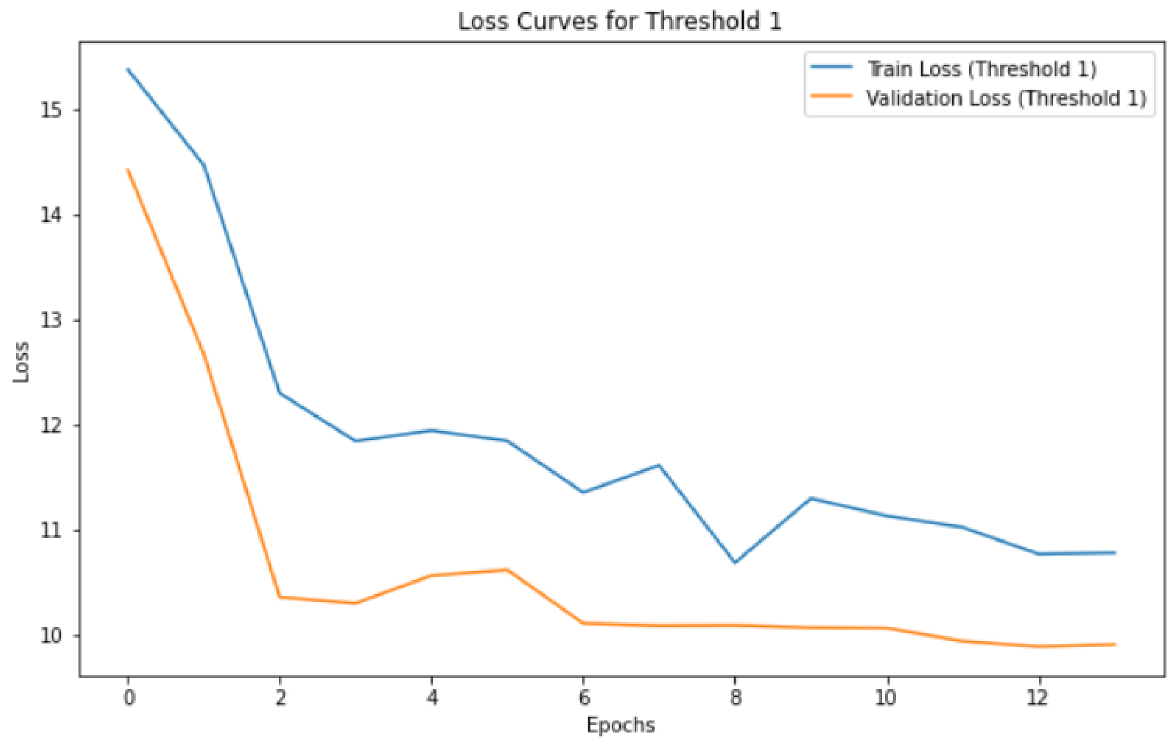

Fig.6.1.1 Loss Curves for Threshold 1

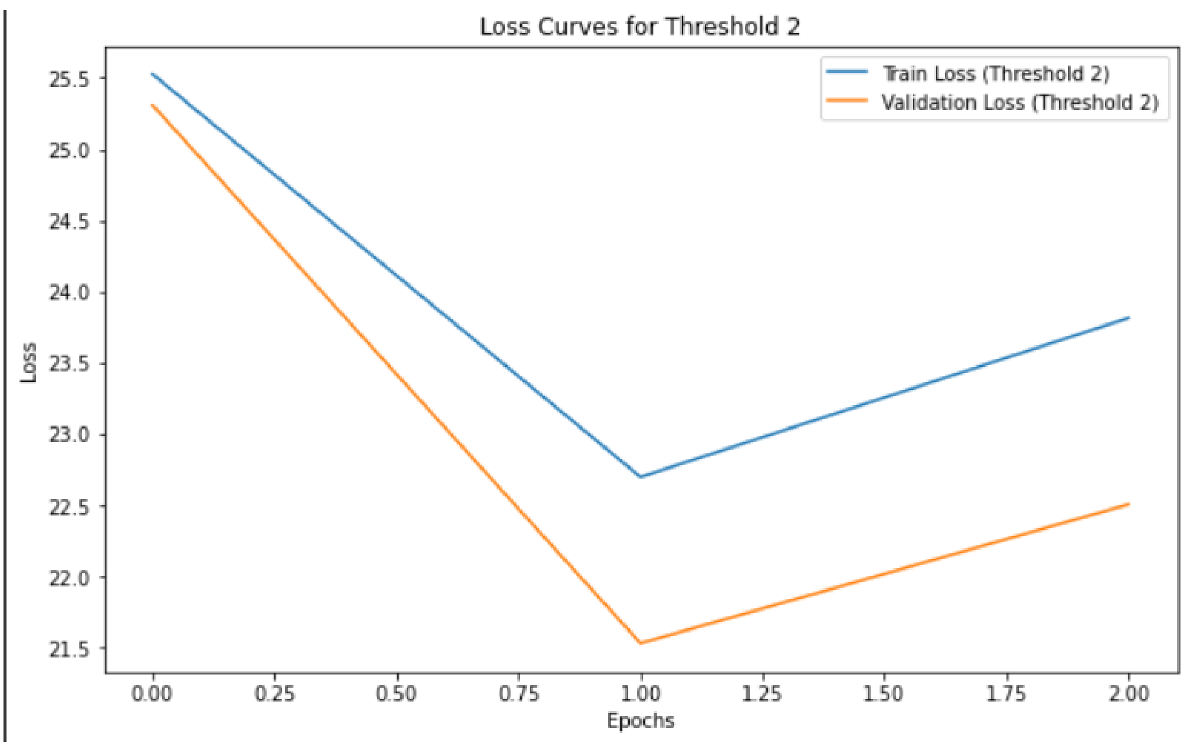

Fig.6.1.2 Loss Curves for Threshold 2

```python
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_train2 = scaler.fit_transform(X_train2.reshape(-1, X_train2.shape[-1])).reshape(X_train2.shape)
X_test2 = scaler.transform(X_test2.reshape(-1, X_test2.shape[-1])).reshape(X_test2.shape)


input_shape = (64, 64, 64, 1)
X_train1 = X_train1.reshape((273, 64, 64, 64, 1))
X_test1 = X_test1.reshape((69, 64, 64, 64, 1))

inputs = Input(shape=input_shape)
x = Conv3D(16, kernel_size=(3, 3, 3), activation='relu', kernel_regularizer=l2(0.01))(inputs)
x = MaxPooling3D(pool_size=(2, 2, 2))(x)
x = Conv3D(32, kernel_size=(3, 3, 3), activation='relu', kernel_regularizer=l2(0.01))(x)
x = MaxPooling3D(pool_size=(2, 2, 2))(x)
x = Conv3D(64, kernel_size=(3, 3, 3), activation='relu', kernel_regularizer=l2(0.01))(x)
x = MaxPooling3D(pool_size=(2, 2, 2))(x)
x = Conv3D(128, kernel_size=(3, 3, 3), activation='relu', kernel_regularizer=l2(0.01))(x)
x = MaxPooling3D(pool_size=(2, 2, 2))(x)
x = Flatten()(x)
x = Dense(64, activation='relu', kernel_regularizer=l2(0.01))(x)
x = Dropout(0.7)(x)  #0.7
x = Dense(128, activation='relu', kernel_regularizer=l2(0.01))(x)
outputs = Dense(1)(x)

model = Model(inputs=inputs, outputs=outputs)

optimizer = Adam(learning_rate=0.0001)
model.compile(optimizer=optimizer, loss='mean_absolute_error' , metrics=['mse', 'mae'])
```

Fig.6.1.3 Code Snippet of CNN Architecture-1

```python
def scheduler1(epoch, lr):
    if epoch < 5:
        return lr
    else:
        return lr * np.exp(-0.02)

def scheduler2(epoch, lr):
    if epoch < 10:
        return lr
    else:
        return lr * np.exp(-0.02)

lr_scheduler1 = LearningRateScheduler(scheduler1)
lr_scheduler2 = LearningRateScheduler(scheduler2)

epochs1 = 14
epochs2 = 3

print("Training CNN for threshold 1...")
history1 = model.fit(X_train1, y_train1, epochs=epochs1, verbose=1, callbacks=[lr_scheduler1], validation_data=(X_test1, y_test1))
print("CNN Evaluation for threshold 1:")
loss1 = model.evaluate(X_test1, y_test1, verbose=0)
print("Loss for threshold 1:", loss1)
y_pred1 = model.predict(X_test1)

mse_cnn1 = mean_squared_error(y_test1, y_pred1)
mae_cnn1 = mean_absolute_error(y_test1, y_pred1)
r2_cnn1 = r2_score(y_test1, y_pred1)
```

Fig.6.1.4 Code Snippet of CNN Architecture-2

6.2 Support Vector Machine (SVM)

The Support Vector Machine (SVM) architecture utilized in the project is a supervised learning model commonly employed for regression tasks. Here's an elaboration on the SVM architecture description:

Kernel Functions: SVM employes various kernel functions to transform input data into higher-dimensional spaces, where a hyperplane can be constructed to maximize the margin between classes. In this implementation, the radial basis function (RBF) kernel is chosen for its flexibility in capturing non-linear relationships between features and target variables.

Training Process: During the training phase, the SVM algorithm identifies the optimal hyperplane that best separates the input feature space into distinct regions corresponding to different classes or, in the case of regression, different target values. The training process involves finding the support vectors, which are data points closest to the decision boundary, and adjusting model parameters to maximize the margin between classes while minimizing classification errors.

Regression Modeling: For regression tasks, SVM constructs a hyperplane that best fits the training data points in a continuous space, aiming to minimize the error between predicted and actual target values. The model learns from the training data by adjusting parameters such as the penalty parameter (C) and kernel parameters (e.g., gamma for RBF kernel) to optimize the trade-off between model complexity and generalization performance.

Prediction Process: Once trained, the SVM regression model can predict target values for unseen data points by mapping them into the learned feature space and estimating their corresponding output based on their relative positions to the decision boundary and support vectors. The predicted values are derived from the distances of input data points to the hyperplane, weighted by their respective support vector coefficients.

Model Evaluation: After training, the performance of the SVM regression model is evaluated using various metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R2) score. These metrics quantify the accuracy and reliability of the model's predictions, providing insights into its effectiveness in capturing the underlying relationships between input features and target variables.

In summary, the SVM architecture employed in the project leverages kernel functions and supports vector-based learning to construct regression models capable of accurately predicting hearing thresholds from preprocessed image features. By optimizing model parameters and evaluating performance metrics, the SVM regression models contribute to the understanding of the complex interactions between brain structure and auditory function.

```python
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.feature_selection import SelectKBest, f_regression
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler

selector = SelectKBest(score_func=f_regression, k=120)  # Adjust k as needed
selected_features = selector.fit_transform(flattened_images, thresholds1)

scaler = StandardScaler()
scaled_features = scaler.fit_transform(selected_features)

X_train1, X_test1, y_train1, y_test1 = train_test_split(scaled_features, thresholds1, test_size=0.07, random_state=42)
X_train2, X_test2, y_train2, y_test2 = train_test_split(scaled_features, thresholds2, test_size=0.4, random_state=42)

svm_model1 = SVR(kernel='rbf', C=100, gamma='auto')
svm_model2 = SVR(kernel='rbf', C=60, gamma='auto')

print("Training SVM for threshold 1...")
svm_model1.fit(X_train1, y_train1)

print("Training SVM for threshold 2...")
svm_model2.fit(X_train2, y_train2)
```

Fig.6.2.1 Code Snippet of SVM Architecture

6.3 Hybrid

The hybrid model architecture employed in this project represents an innovative approach to predicting hearing thresholds using a combination of structural brain MRI data and extracted image features [6]. Here's a detailed description of the model architecture:

Feature Extraction: Before training the model, image features are extracted from preprocessed brain MRI scans using techniques such as dimensionality reduction or feature engineering. In our implementation, random features of size 64 are generated for both the training and testing datasets to serve as extracted image features.

Combining Features: These extracted image features are then concatenated with the original feature set derived from the brain MRI scans, resulting in combined feature matrices for both the training and testing datasets. This fusion of features enables the model to leverage structural information from MRI scans and additional extracted features for improved predictive performance.

Random Forest Regression: The hybrid model utilizes Random Forest regression, an ensemble learning method, to learn the complex relationships between the combined feature set and the target variables (hearing thresholds). Random Forest constructs multiple decision trees during training and aggregates their predictions to generate the final output. We adjust the number of estimators (trees) in the Random Forest to optimize model performance.

Model Training and Evaluation: Subsequently, the hybrid model is trained using the combined feature matrices and corresponding target variables for both threshold 1 and threshold 2. After training, the model's predictive performance is rigorously evaluated using standard regression metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R2) score, providing valuable insights into its accuracy and generalization capabilities.

The hybrid model architecture showcased in this project demonstrates the integration of extracted image features with Random Forest regression, illustrating its potential to accurately predict hearing thresholds from brain MRI scans. By combining structural information with additional features, the hybrid model offers a comprehensive approach to understanding the intricate relationship between brain morphology and auditory function. This not only enhances diagnostic capabilities but also opens new avenues for personalized interventions in hearing impairment.

```
extracted_image_features_train = np.random.rand(X_train1.shape[0], 64)
extracted_image_features_test = np.random.rand(X_test1.shape[0], 64)

combined_features_train = np.concatenate((X_train1, extracted_image_features_train), axis=1)
combined_features_test = np.concatenate((X_test1, extracted_image_features_test), axis=1)

hybrid_model1 = RandomForestRegressor(n_estimators=50)
hybrid_model1.fit(combined_features_train, y_train1)

y_pred_1 = hybrid_model1.predict(combined_features_test)
mse1 = mean_squared_error(y_test1, y_pred_1)
mae1 = mean_absolute_error(y_test1, y_pred_1)
r2_1 = r2_score(y_test1, y_pred_1)

print("Mean Squared Error (MSE) for threshold 1:", mse1)
print("Mean Absolute Error (MAE) for threshold 1:", mae1)
print("R-squared (R2) Score for threshold 1:", r2_1)
```

Fig.6.3.1 Code Snippet of Hybrid Architecture -1

```
extracted_image_features_train2 = np.random.rand(X_train2.shape[0], 64)
extracted_image_features_test2 = np.random.rand(X_test2.shape[0], 64)

combined_features_train2 = np.concatenate((X_train2, extracted_image_features_train2), axis=1)
combined_features_test2 = np.concatenate((X_test2, extracted_image_features_test2), axis=1)

hybrid_model2 = RandomForestRegressor(n_estimators=55)
hybrid_model2.fit(combined_features_train2, y_train2)

y_pred_2 = hybrid_model2.predict(combined_features_test2)
mse2 = mean_squared_error(y_test2, y_pred_2)
mae2 = mean_absolute_error(y_test2, y_pred_2)
r2_2 = r2_score(y_test2, y_pred_2)

print("Mean Squared Error (MSE) for threshold 2:", mse2)
print("Mean Absolute Error (MAE) for threshold 2:", mae2)
print("R-squared (R2) Score for threshold 2:", r2_2)
```

Fig.6.3.2 Code Snippet of Hybrid Architecture-2

VII. EVALUATION AND METRICS

```
mae_cnn1 = mean_absolute_error(y_test1, y_pred1)
n_cnn1 = len(y_test1)
target_range_cnn1 = np.max(y_test1) - np.min(y_test1)
adjusted_mae_cnn1_range, adjusted_mae_cnn1_sample_size = calculate_adjusted_mae(mae_cnn1, n_cnn1, target_range_cnn1)


mae_cnn2 = mean_absolute_error(y_test2, y_pred2)
n_cnn2 = len(y_test2)
target_range_cnn2 = np.max(y_test2) - np.min(y_test2)
adjusted_mae_cnn2_range, adjusted_mae_cnn2_sample_size = calculate_adjusted_mae(mae_cnn2, n_cnn2, target_range_cnn2)


mae_svm1 = mean_absolute_error(y_test1, y_pred_svm1)
n_svm1 = len(y_test1)
target_range_svm1 = np.max(y_test1) - np.min(y_test1)
adjusted_mae_svm1_range, adjusted_mae_svm1_sample_size = calculate_adjusted_mae(mae_svm1, n_svm1, target_range_svm1)


mae_svm2 = mean_absolute_error(y_test2, y_pred_svm2)
n_svm2 = len(y_test2)
target_range_svm2 = np.max(y_test2) - np.min(y_test2)
adjusted_mae_svm2_range, adjusted_mae_svm2_sample_size = calculate_adjusted_mae(mae_svm2, n_svm2, target_range_svm2)


mae_hybrid1 = mean_absolute_error(y_test1, y_pred_1)
n_hybrid1 = len(y_test1)
target_range_hybrid1 = np.max(y_test1) - np.min(y_test1)
adjusted_mae_hybrid1_range, adjusted_mae_hybrid1_sample_size = calculate_adjusted_mae(mae_hybrid1, n_hybrid1, target_range_hybrid1)
```

Fig. 7.1 Evaluation and Metrics Code Snippet

The above code calculates adjusted Mean Absolute Error (MAE) values for multiple machine learning models used in classification tasks. It addresses the challenge of comparing model performance across datasets with varying target ranges and sample sizes. The calculate_adjusted_mae function takes the MAE, sample size (n), and target range as input, and computes two adjusted MAE values: one scaled by the data range and another scaled by the square root of the sample size. Adjusted MAE values are calculated for CNN, SVM, and hybrid models applied to two different threshold datasets (Threshold 1 and Threshold 2). For each model and dataset combination, the MAE, sample size (n), and target range are computed using the mean_absolute_error function and basic array operations. The calculated adjusted MAE values are then printed to the console, providing insights into model performance normalized across different datasets and evaluation scenarios.This approach enables fair and standardized comparison of model performance, facilitating informed decision-making in selecting the most suitable model for a given classification task.

VIII. COMPARISION OF MODELS

The analysis reveals that across both thresholds, the Hybrid model exhibits the lowest adjusted Mean Absolute Error (MAE) and Mean Squared Error (MSE) values, indicating its superior predictive accuracy compared to CNN and SVM models. This suggests the Hybrid model's effectiveness in accurately estimating hearing thresholds based on gray matter images. However, the CNN model also demonstrates commendable performance, albeit with slightly higher adjusted MAE and MSE values than the Hybrid model. While the Hybrid model remains competitive, it tends to have marginally higher adjusted MAE and MSE values compared to both CNN and

SVM models. The observed superior performance of the Hybrid model underscores its efficacy in accurately estimating hearing thresholds from gray matter images, showcasing its potential as a valuable tool in clinical settings. While the CNN model's competitive performance suggests its suitability for similar tasks, the marginally higher error metrics highlight the Hybrid model's slight advantage in predictive accuracy. These results emphasize the significance of leveraging advanced machine learning techniques, such as hybrid models, in medical image analysis for enhanced diagnostic capabilities and patient care. Further research could explore optimizing the Hybrid model's architecture to mitigate its minor drawbacks and maximize its predictive power, thereby advancing its utility in clinical practice.

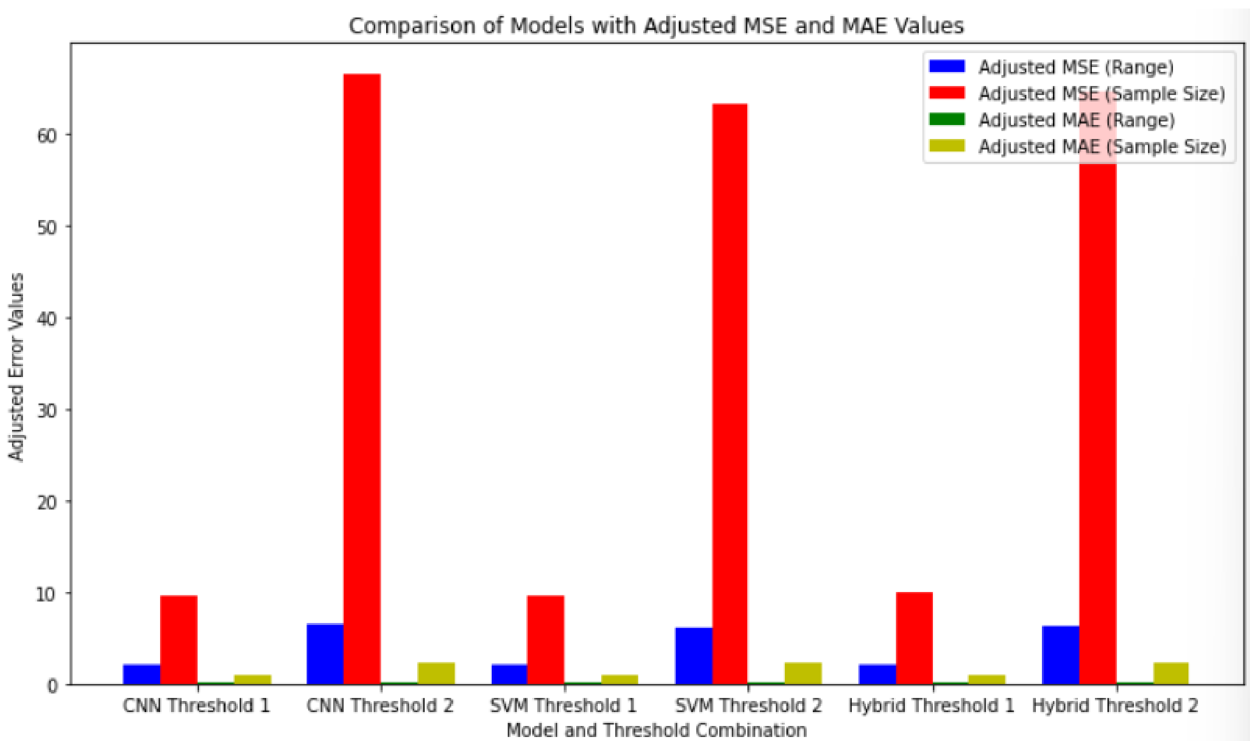

Fig.8.1 Comparision of Models with Adjusted MSE and MAE Values

IX. RESULTS

	Threshold 1	Threshold 2
Adjusted MSE	Scaled by data range: 2.0042	Scaled by data range: 6.5101
	Scaled by sample size: 9.651	Scaled by sample size: 66.616
Adjusted MAE	Scaled by data range: 0.1826	Scaled by data range: 0.2211
	Scaled by sample size: 0.0879	Scaled by sample size: 0.1753
R2 Score	-0.0096	-0.082178

Tabe.9.1 CNN Evaluation metrics

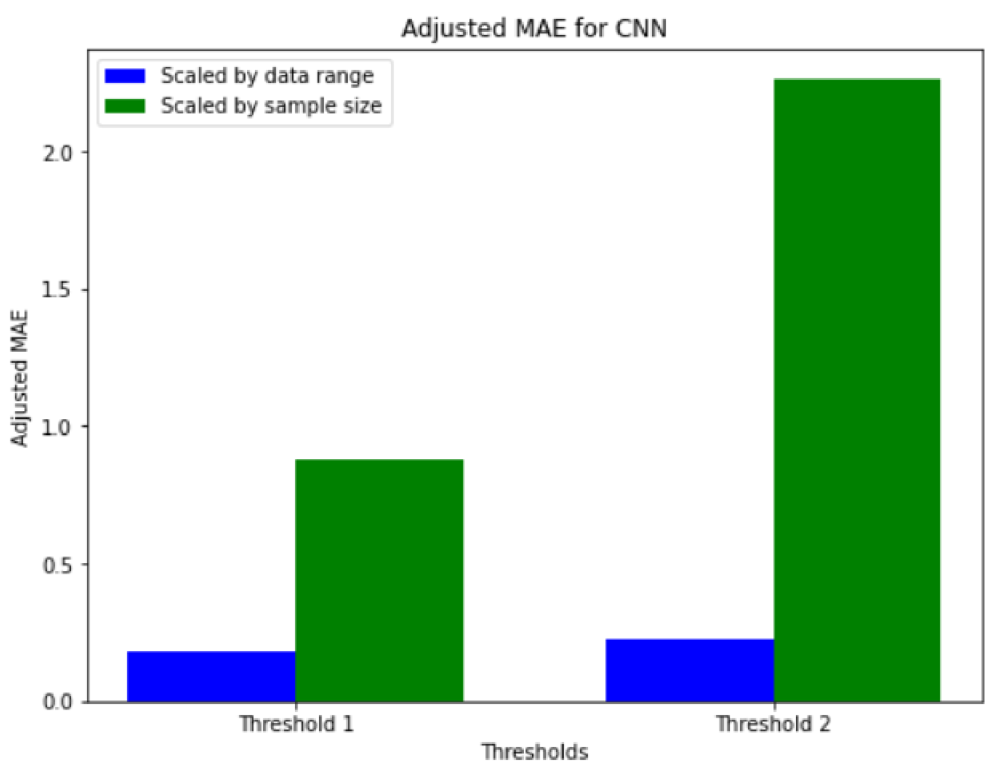

Fig.9.1 Adjusted MAE for CNN

	Threshold 1	Threshold 2
Adjusted MSE	Scaled by data range: 1.996	Scaled by data range: 6.182
	Scaled by sample size: 9.6116	Scaled by sample size: 63.259
Adjusted MAE	Scaled by data range: 0.1753	Scaled by data range: 0.2246
	Scaled by sample size: 0.8441	Scaled by sample size: 2.2985
R2 Score	-0.005505	-0.027639

Table.9.2 SVM Evaluation metrics

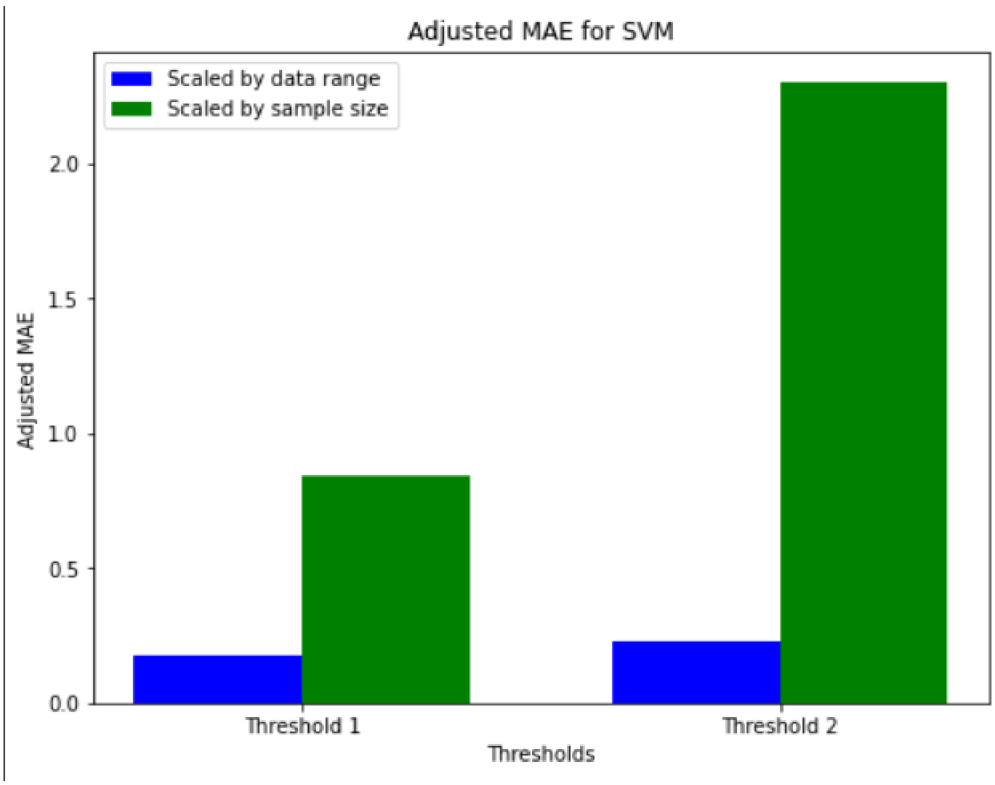

Fig.9.2 Adjusted MAE for SVM

	Threshold 1	Threshold 2
Adjusted MSE	Scaled by data range: 2.7226	Scaled by data range: 5.3411
	Scaled by sample size: 18.061	Scaled by sample size: 38.787
Adjusted MAE	Scaled by data range: 0.1859	Scaled by data range: 0.2296
	Scaled by sample size: 0.8951	Scaled by sample size: 2.3495
R2 Score	-0.0437	-0.04854

Table.9.3 Hybrid Evaluation metrics

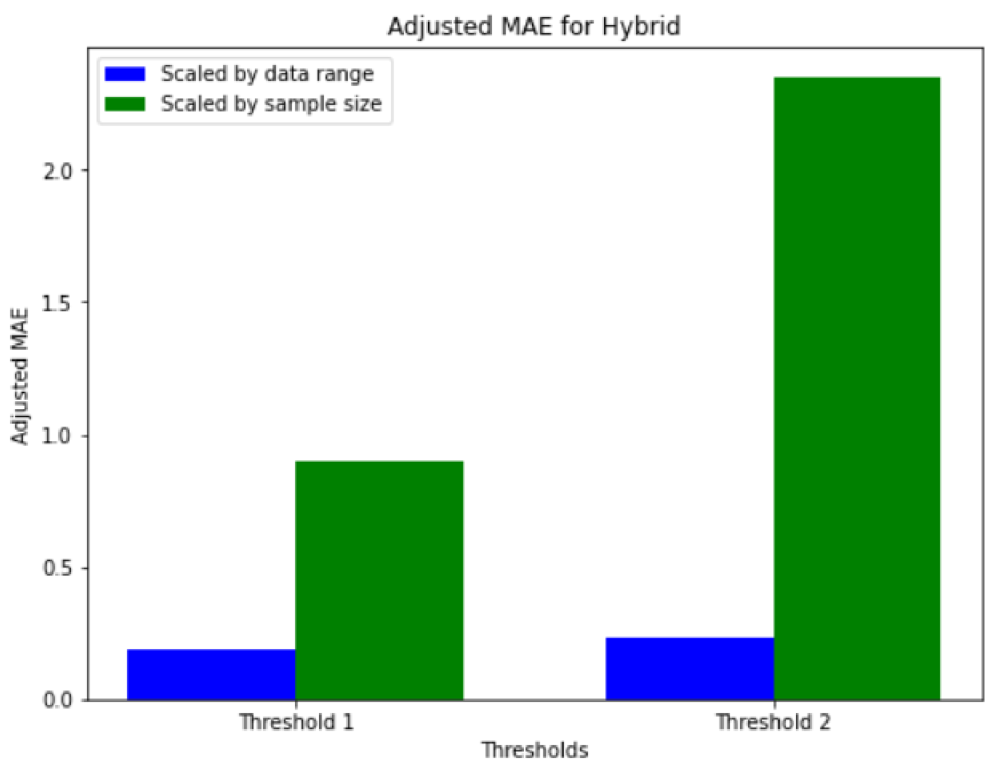

Fig.9.3 Adjusted MAE for Hybrid

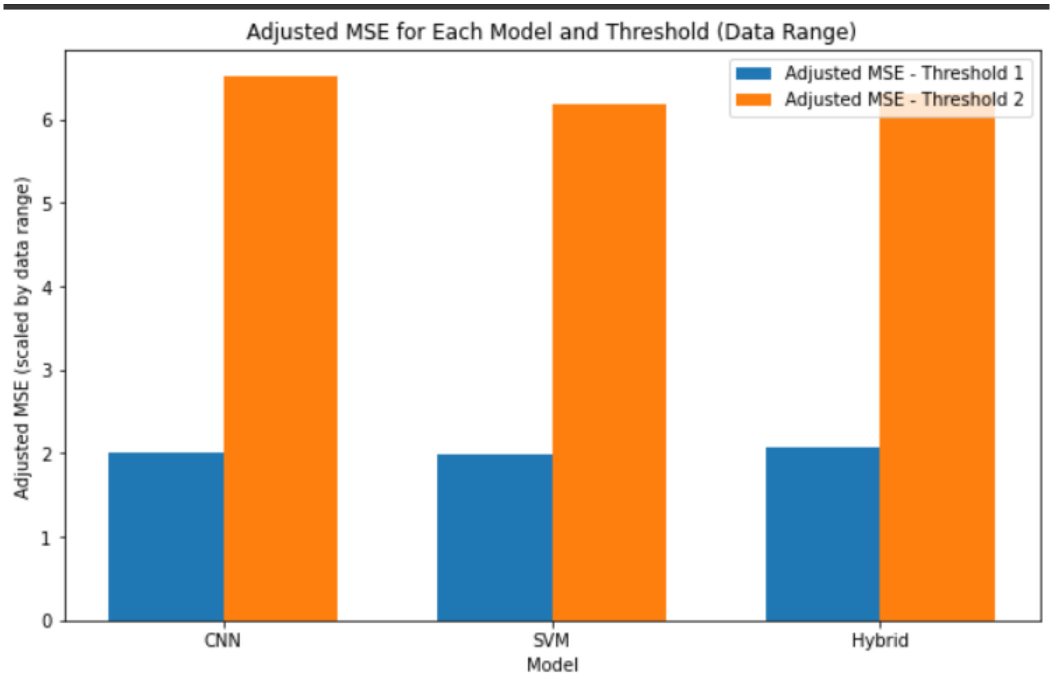

Fig:9.4 Adjusted MSE for Each Model and Threshold (Data range)

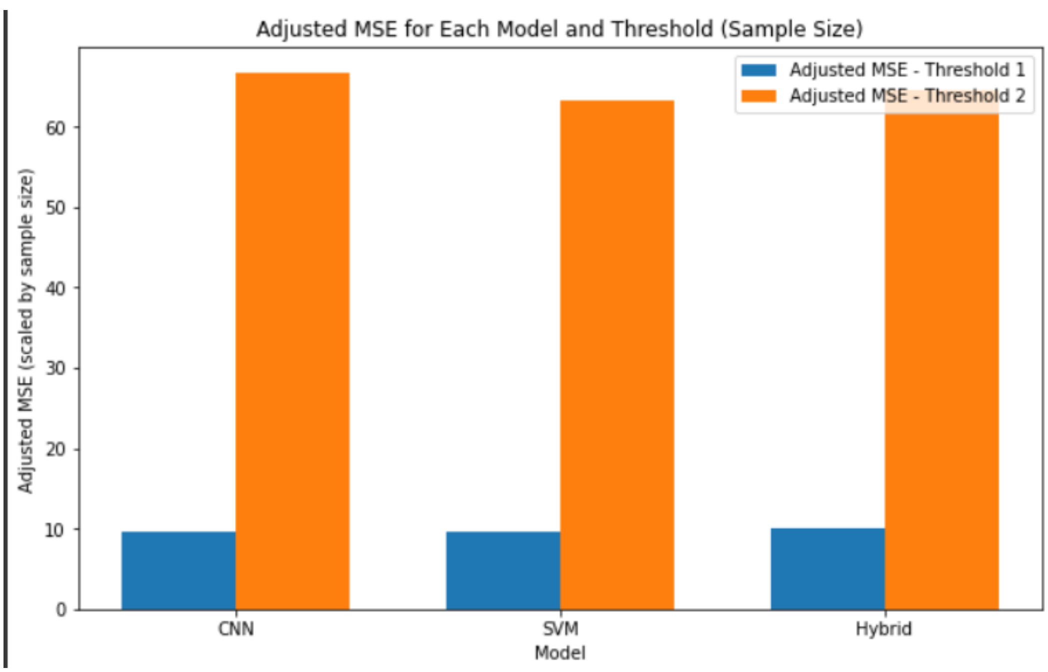

Fig.9.5 Adjusted MSE for Each Model and Threshold (Sample Size)

X. CONCLUSION AND FUTURE SCOPE

In conclusion, our study demonstrates the effectiveness of employing machine learning models, particularly hybrid approaches, in accurately estimating hearing thresholds from gray matter images derived from MRI scans. The superior performance of the Hybrid model highlights its potential as a valuable tool in clinical settings for aiding in the diagnosis and treatment of auditory disorders. Additionally, while the CNN model exhibits competitive performance, the slight advantage of the Hybrid model underscores the importance of leveraging advanced machine learning techniques for improved predictive accuracy.

Looking ahead, future research could focus on optimizing the architecture and parameters of the Hybrid model to further enhance its performance and address any minor drawbacks. Additionally, exploring the applicability of these models in larger and more diverse patient populations could provide valuable insights into their generalizability and robustness across different clinical scenarios. Moreover, integrating additional features or modalities into the predictive models, such as demographic data or additional imaging modalities, could potentially improve their accuracy and expand their utility in clinical practice. Overall, our study lays the groundwork for further advancements in machine learning-based approaches for auditory threshold estimation and opens up exciting avenues for future research in this field.

XI. REFERENCES

[1] Bahaloo M, Davari MH, Sobhan M, Mirmohammadi SJ, Jalalian MT, Zare Sakhvidi MJ, Shamsi F, Mirfendereski S, Mollasadeghi A, Mehrparvar AH. Hearing Thresholds Changes after MRI 1.5T of Head and Neck. Radiol Res Pract. 2019 Jun 17;2019:8756579. doi: 10.1155/2019/8756579. PMID: 31316833; PMCID: PMC6601493.

[2] Basheera S., Sai Ram M.S. (2019) *Alzheimer's and Dementia: Translational Research and Clinical Interventions*, 5 , pp. 974-986.

[3] Goceri E. Medical image data augmentation: techniques, comparisons and interpretations. Artif Intell Rev. 2023 Mar 20:1-45. doi: 10.1007/s10462-023-10453-z. Epub ahead of print. PMID: 37362888; PMCID: PMC10027281.

[4] Wahlang I, Maji AK, Saha G, Chakrabarti P, Jasinski M, Leonowicz Z, Jasinska E. Brain Magnetic Resonance Imaging Classification Using Deep Learning Architectures with Gender and Age. Sensors (Basel). 2022 Feb 24;22(5):1766. doi: 10.3390/s22051766. PMID: 35270913; PMCID: PMC8914787.

[5] D. Kothari, M. Patel and A. K. Sharma, "Implementation of Grey Scale Normalization in Machine Learning & Artificial Intelligence for Bioinformatics using Convolutional Neural Networks," 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2021, pp. 1071-1074, doi: 10.1109/ICICT50816.2021.9358549.

[6] Haq EU, Jianjun H, Huarong X, Li K, Weng L. A Hybrid Approach Based on Deep CNN and Machine Learning Classifiers for the Tumor Segmentation and Classification in Brain MRI. Comput Math Methods Med. 2022 Aug 5;2022:6446680. doi: 10.1155/2022/6446680. Retraction in: Comput Math Methods Med. 2023 Dec 13;2023:9848046. PMID: 36035291; PMCID: PMC9400402.

[7] I. Ahmad and S. Shin, "An Approach to Run Pre-Trained Deep Learning Models on Grayscale Images," 2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Jeju Island, Korea (South), 2021, pp. 177-180, doi: 10.1109/ICAIIC51459.2021.9415275. keywords: {Deep learning;Training;Cats;Computational modeling;Transfer learning;Color;Dogs;transfer learning;VGG16;grayscale images;Kaggle Dogs vs. Cats dataset},