Speech Commands Classification Task

1. Summary of the Paper

The "Speech Commands" paper introduces a dataset designed for training and evaluating keyword spotting systems. The dataset contains over 100,000 one-second audio utterances of 35 common words recorded by 2,618 speakers. It aims to provide a standardized collection to build and benchmark small models that detect when a specific word is spoken, minimizing false positives from background noise or unrelated speech.

2. Data Analysis

After downloading and extracting the dataset, the following analyses were conducted:

- **Commands and Counts:** We identified a list of all available commands and counted the number of .wav files for each command to understand the dataset's distribution.
- **Unique Speakers:** Using regular expressions to extract speaker IDs from the filenames, we determined the total number of unique speakers in the dataset: **2,618 speakers**.
- **Audio Duration:** The duration of each audio file was calculated using the number of frames and sample rate. The average duration of the audio files was found to be approximately **1 second**, with a minimum and maximum duration also around **1 second**, as expected given the dataset's design.

3. Modelling

We built a convolutional neural network (CNN) to classify the spoken commands. The audio files were pre-processed into a uniform format (16,000 samples per second) and split into training, validation, and testing datasets.

- **Training Data:** 70% of the data.
- **Validation Data:** 15% of the data.
- **Testing Data:** 15% of the data.

The CNN model consisted of multiple layers:

- **Input Layer:** Handles 1D audio inputs.
- **Three Convolutional Layers:** Extracts features from the audio data.
- **Pooling Layers:** Reduces the dimensionality and computational complexity.
- **Global Average Pooling Layer:** Aggregates the feature maps.
- **Dense Layers:** Fully connected layers with ReLU and SoftMax activation for classification.

4. Model Training and Performance

The model was compiled using the Adam optimizer, with sparse categorical cross-entropy as the loss function and accuracy as the metric. The model was trained for **6 epochs**, and the performance was evaluated on the test data.

5. Classification Report and Confusion Matrix

The classification report provided detailed precision, recall, and F1-score metrics for each command, showing the model's performance across all classes. The confusion matrix visualized the number of correct and incorrect predictions, highlighting where the model performed well and where it struggled.

6. Fine-Tuning with Personal Voice Data

To adapt the model for personalized voice data, 10 samples of each command were recorded in the user’s voice, creating a new dataset. The model was fine-tuned on this new dataset to improve its performance on the user-specific voice.

7. Conclusion

The speech command classification task was successfully completed with a robust model that demonstrated reasonable accuracy on both the standard dataset and the user-specific dataset. Fine-tuning with personalized data further improved performance, indicating the model's adaptability to different voices.