MelTrim: Coarse-to-Fine Data Pruning for Speech Classification

Shaobo Wang[1]*, Tianle Niu[1]*, Xuan Ouyang[1]*, Xintong Li[1], Zhengkun Ge[1,2], Yue Min[1], Xiaoqian Liu[1,2], Hankun Wang[1], Linfeng Zhang[1]†
[1]EPIC Lab, Shanghai Jiao Tong University
[2]School of Computer Science and Engineering, Northeastern University
{shaobowang1009, linfengzhang}@sjtu.edu.cn,
turturtur250@gmail.com

Abstract

Dataset Pruning (DP) aims to construct a coreset that achieves performance comparable to the original, full dataset. However, few studies have explored DP in the context of Speech Classification (SC) tasks. Unlike image or text classification, SC is particularly challenging due to the difficulty in capturing the acoustic, semantic, and contextual representations. In this study, we propose a novel dataset pruning method for speech datasets, termed MelTrim, which uses a two-step coarse-to-fine framework designed to address these challenges. Specifically, in Step 1, MelTrim coarsely filters *utterance-level* redundant samples using DBSCAN clustering on Mel-Frequency Cepstral Coefficients (MFCC) features, which are first flattened and then reduced in dimensionality using UMAP. In Step 2, we perform *frame-level* redundancy pruning for each utterance via utility pruning, which aims to eliminate irrelevant frames within each utterance. To the best of our knowledge, this is the first dataset pruning approach designed for Speech Classification tasks, demonstrating outstanding performance compared to classical general DP methods. Notably, for the Speech Emotion Recognition, our method achieves up to a 49.5% improvement in WA (Weighted Accuracy) on the MEAD dataset. For the Speaker Identification tasks, it results in a 41.9% reduction in EER (Equal Error Rate) on the VoxCeleb1 dataset. Our implementation is publicly available at https://github.com/turturturturtur/MelTrim.

1 Introduction

In recent years, the rapid advancement of speech foundation models such as HuBERT (Hsu et al., 2021), Wav2Vec 2.0 (Baevski et al., 2020), and Whisper (Radford et al., 2023), along with the evolution of large-scale speech and multimodal models (Chu et al., 2024; Hu et al., 2024), has positioned speech as a critical modality in intelligent perception and interaction systems. These

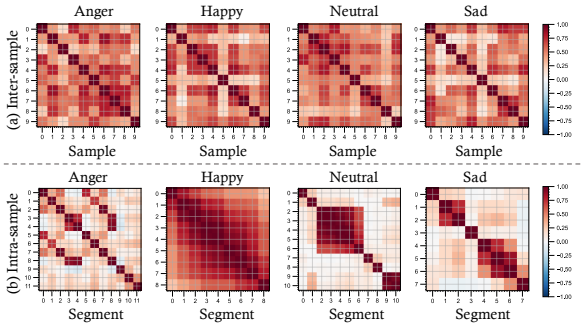

Figure 1: (a) Inter-sample (utterance-level) and (b) Intra-sample (frame-level) redundancy phenomenon in speech recognition. We evaluated the cosine similarity within and across samples based on syllable-level features extracted from M3ED using Sylber (Cho et al., 2024).

models, typically trained on *thousands of hours* of raw audio, aim to capture rich acoustic, semantic, and contextual representations that generalize across downstream tasks such as speech translation (ST) (Liu et al., 2024), speaker identification (SID) (Jung et al., 2022a), speech emotion recognition (SER) (Ma et al., 2024b,a) and automatic speech recognition (ASR) (An et al., 2024). However, speech data is inherently high-dimensional, temporally dense, and structurally redundant (Xu et al., 2023), resulting in significant storage and computational burdens in large-scale training.

To mitigate training costs and enhance data efficiency, data pruning (DP) techniques have been extensively studied in the computer vision (CV) domain, where large-scale datasets often exhibit substantial redundancy. These methods aim to identify and retain representative and informative subsets that enable effective model training while reducing computational overhead (Paul et al., 2022; Killamsetty et al., 2021; Margatina et al., 2021; Agarwal et al., 2020; Coleman et al., 2019; Mirzasoleiman et al., 2020). Despite their success in CV, such strategies do not transfer directly to speech data, where redundancy arises both across utterances and within utterances at the frame level. Systematic approaches that explicitly address these unique forms of redundancy remain largely underexplored.

To address this gap, we begin by empirically

* Equal contribution.
† Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 13751–13765

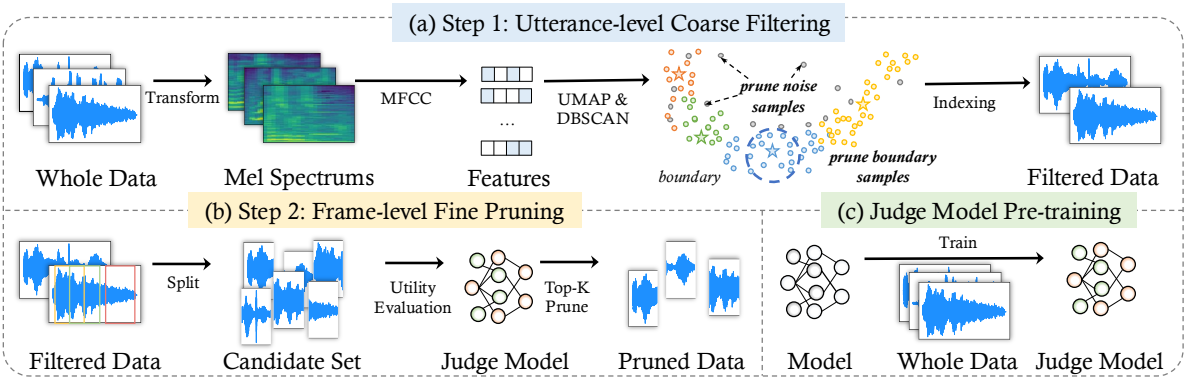

Figure 2: Pipeline of MelTrim. (a) Step 1: At the coarse stage, we operate on the utterance level by applying density-based clustering (DBSCAN) to dimensionally reduced MFCC representations. (b) Step 2: At the fine-grained stage, we prune at the frame level using utility-based selection to discard uninformative segments, retaining only the most discriminative frames for downstream modeling. (c) The utility is judged by a pre-trained model on whole data.

investigating redundancy in speech data using syllable-level features extracted from the M3ED dataset via Sylber (Cho et al., 2024). Specifically, we compute cosine similarities for intra- and inter-sample features. As shown in Figure 1, speech data exhibits two key forms of redundancy:

- **Inter-sample redundancy**, as shown in Figure 1(a), where numerous utterances share overlapping semantic patterns and thus contribute limited additional information.
- **Intra-sample redundancy**, as illustrated in Figure 1(b), where adjacent frames within an utterance carry redundant information content due to speech continuity.

These redundancies substantially reduce the utility density of speech corpora, leading to inefficient data usage, longer training times, and diminished scalability for large-model training.

To tackle these challenges, we propose a two-stage coarse-to-fine pruning framework, MelTrim, for speech data, as depicted in Figure 2. (i) At the coarse stage, we operate on the utterance level: applying density-based clustering (DBSCAN) to dimensionally reduced MFCC representations to eliminate low-utility utterances while preserving acoustic diversity. (ii) At the fine-grained stage, we prune at the frame level using utility-based selection to discard uninformative segments (*e.g.*, silence, filler, and redundant content), retaining only the most discriminative frames for downstream modeling. Our contributions are as follows:

- We explore the redundancy phenomenon in speech recognition. As illustrated in Figure 1, we observe that both utterance-level and frame-level redundancy associated with audio samples across different datasets.
- To the best of our knowledge, we are the first to design a generally applicable, coarse-to-fine pruning framework to eliminate redundancy at both utterance and frame levels, enhancing the efficiency and compactness of speech data.
- We validate our method on representative speech classification tasks, *i.e.*, Speech Emotion Recognition and Speaker Identification, demonstrating that it achieves substantial data reduction with minimal impact on model performance. **Particularly, in Speech Emotion Recognition task, our method achieves up to a 49.5% improvement in WA on the MEAD dataset.** In the Speaker Identification task, it leads to a 41.9% reduction in EER on the VoxCeleb1 dataset. Beyond speech classification, we also conduct a small-scale *ASR* evaluation on MELD with a Wav2Vec judge; MelTrim yields consistent WER reductions under low-budget coresets.
- We further show MelTrim's *weak-to-strong* ability for different Whisper architectures, demonstrating its ability for more efficient cross-architecture pruning for speech classification.

2 Method: MelTrim

2.1 Preliminaries: Data Pruning

Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, the objective of data pruning (DP) is to obtain a smaller dataset $\tilde{\mathcal{D}} = \{(\tilde{x}_i, y_i)\}_{i=1}^{m}$ with $m \ll n$ samples. Note that each $\tilde{x}_i$ can either correspond to an element in $\mathcal{D}$ or be a compressed representation of a sample $x_j \in \mathcal{D}$. The desired dataset $\tilde{\mathcal{D}}$ should enable a model to achieve comparable, or even lossless, performance when compared to a model trained on $\mathcal{D}$. In practice, the performance is evaluated on a held-out test dataset $\mathcal{D}_{\text{test}}$. Specifically, for a model f parameterized by θ trained with cross-

entropy loss $\mathcal{L}$, the following condition must hold:

$$\min_{\tilde{\mathcal{D}}} \sum_{(x,y)\in\mathcal{D}_{\text{test}}} \left\| \mathcal{L}(f_{\theta_{\mathcal{D}}}(x), y) - \mathcal{L}(f_{\theta_{\tilde{\mathcal{D}}}}(x), y) \right\|, \tag{1}$$

where $\theta_{\mathcal{D}}$ and $\theta_{\tilde{\mathcal{D}}}$ denote the parameters of models trained on the original dataset $\mathcal{D}$ and the pruned dataset $\tilde{\mathcal{D}}$, respectively.

2.2 Step 1: Utterance-level Coarse Filtering

To efficiently prune the speech dataset, we first perform coarse filtering based on Mel-Frequency Cepstral Coefficients (MFCC), a classic yet effective representation in speech signal processing (Xu et al., 2004). The objective at this stage is to preserve the full *acoustic diversity* of the data—capturing variations in pitch, timbre, prosody, and recording conditions—before any task-specific selection. Unlike deep features from modern pretrained models (e.g., Whisper) that are explicitly optimized for *semantic invariance* and thus conflate acoustically distinct utterances with the same content, MFCCs remain highly sensitive to acoustic variability. This property makes them a principled choice for clustering at the utterance level, ensuring that the resulting coreset faithfully covers the acoustic manifold rather than collapsing into semantically homogeneous representations. The full process of Step 1 is illustrated in Figure 2(a).

MFCC Feature Extraction. MFCC features effectively capture both speech and tonal information within a speech signal, making them essential for speech feature extraction (Pakyurek et al., 2020; Deng et al., 2020). The process mainly consists of five steps, *i.e.*, Pre-emphasis, Framing and Windowing, Fast Fourier Transform, Mel Filter Bank, and Logarithm and Discrete Cosine Transform.

Formally, given an input waveform x_i, we extract its MFCC representation via a transformation: Given the established rationale for prioritizing acoustic fidelity, we operationalize this by employing the MFCC feature extraction pipeline. This pipeline is a standard and effective method for transforming a raw audio waveform into a compact, informative representation that preserves the essential spectro-temporal characteristics of the speech signal (Pakyurek et al., 2020; Deng et al., 2020).

The process systematically distills the acoustic information through five key stages: Pre-emphasis, Framing and Windowing, Fast Fourier Transform, Mel Filter Bank, and Logarithm and Discrete Cosine Transform. Formally, given an input waveform x_i, we extract its MFCC representation via a transformation:

$$\mathbf{X}_i = \text{MFCC}(x_i) \in \mathbb{R}^{20\times L_i}, \tag{2}$$

where $\mathbf{X}_i$ is the MFCC feature matrix of the i-th sample, containing the first 20 cepstral coefficients per frame over L_i frames. This approach allows us to capture the dynamic spectral changes in the speech signal and provides valuable information for subsequent clustering and distillation processes.

Flattening. To ensure consistency across the dataset and enable batch processing for downstream analysis, we first zero-pad or truncate each MFCC feature matrix $\mathbf{X}_i \in \mathbb{R}^{20\times L_i}$ a fixed sequence length L_{fix}. This results in a representation:

$$\tilde{\mathbf{X}}_i \in \mathbb{R}^{20\times L_{\text{fix}}}, \tag{3}$$

where $\tilde{\mathbf{X}}_i$ denotes the padded MFCC matrix of i-th sample. We then flatten each matrix into a vector:

$$\mathbf{f}_i = \text{Flatten}(\tilde{\mathbf{X}}_i) \in \mathbb{R}^d, \quad \text{with } d = 20 \cdot L_{\text{fix}}. \tag{4}$$

This yields a fixed-length feature matrix: $\mathcal{X}_{\text{flat}} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N]^\top \in \mathbb{R}^{n\times d}$. This uniform structure allows for efficient dimensionality reduction and clustering in the subsequent steps.

UMAP Dimensionality Reduction. To further reduce computational complexity and enable low-dimensional clustering, we apply Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018) to the flattened MFCC feature matrix. Given the unified feature matrix $\mathcal{X}_{\text{flat}} \in \mathbb{R}^{n\times d}$ obtained from the previous step, UMAP projects it to a 2D space:

$$\mathcal{Z} = \text{UMAP}(\mathcal{X}_{\text{flat}}) \in \mathbb{R}^{n\times 2}, \tag{5}$$

where $\mathcal{Z}$ denotes the 2D representation of all n samples. UMAP builds a neighborhood graph in the high-dimensional space and learns a low-dimensional embedding that preserves the local manifold structure. This projection facilitates subsequent clustering and visualization.

Clustering with DBSCAN in 2D Space. After dimensionality reduction, the speech features are mapped into a 2D plane using UMAP. To further analyze and select representative samples, we apply the DBSCAN (Wang et al., 2017) clustering algorithm, which is a density-based clustering method that can automatically discover clusters of arbitrary

shape and effectively identify outliers without requiring the number of clusters in advance.

We first apply DBSCAN to the 2D embeddings $\mathcal{Z} = \{\mathbf{z}_i\}_{i=1}^n$ to obtain cluster labels $c_i \in \{1, \ldots, K\} \cup \{-1\}$, where -1 denotes noise points. All samples labeled as noise are discarded. We then obtain a pre-selected subset $\mathcal{S}$ by sampling representative points from the remaining clusters.

Let $\mathcal{C}_k = \{\mathbf{z}_i \mid c_i = k\}$ denote the k-th cluster, and let $\tilde{m}$ be the total number of samples to be selected. To allocate samples proportionally according to cluster size, we compute:

$$M_k = \left\lfloor \frac{|\mathcal{C}_k|}{\sum_{j=1}^{K} |\mathcal{C}_j|} \cdot \tilde{m} \right\rfloor, \quad \text{for } k = 1, \ldots, K, \tag{6}$$

where $\lfloor \cdot \rfloor$ denotes the floor function. The centroid of cluster $\mathcal{C}_k$ is defined as:

$$\boldsymbol{\mu}_k = \frac{1}{|\mathcal{C}_k|} \sum_{\mathbf{z}_i \in \mathcal{C}_k} \mathbf{z}_i. \tag{7}$$

For each cluster, we retain M_k points that are closest to the centroid in terms of Euclidean distance:

$$\mathcal{S} = \bigcup_{k=1}^{K} \underset{\mathbf{z}_i \in \mathcal{C}_k}{\text{Top-}M_k} \left(-\|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2 \right). \tag{8}$$

The resulting subset $\mathcal{S}$ is used for subsequent selection, where $|\mathcal{S}| = \tilde{m}$.

2.3 Step 2: Frame-level Fine-grained Pruning

To enhance the representativeness of speech data and improve model generalization, we further propose frame-level fine pruning based on utility. The overall process consists of three steps: *utterance split*, *utility-based segment scoring*, and *key frames selection* with the highest utility scores. The entire process of Step 2 is shown in Figure 2(b).

Utterance Split. Given a pre-selected subset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{\tilde{m}}$, where each $x_i \in \mathbb{R}^L$ is a full-length speech sample, we further split multiple fixed-length segments from each x_i to construct a frame-level candidate set. Each segment has length $w = \frac{L}{4}$, and its starting point t_k is sampled uniformly from $[0, L - w]$, resulting in:

$$\xi_{i,k} = x_i[t_k : t_k + w], \quad t_k \sim \mathcal{U}(0, L - w) \tag{9}$$

The final candidate set for each x_i is:

$$\mathcal{P}_i = \{\xi_{i,1}, \xi_{i,2}, \ldots, \xi_{i,K}\}, \quad \xi_{i,k} \in \mathbb{R}^w \tag{10}$$

Utility Scoring. For each segment, we evaluate its importance with its *Utility* towards a pre-trained *Judge Model*.

The Judge Model as a Task-Aligned Proxy. To provide an objective evaluation standard for scoring and pruning, we pre-train a **Judge Model**, as illustrated in Figure 2(c). The conceptual role of this model is to act as a **task-aligned, lightweight proxy** for the final downstream model. Its design follows two key principles:

1. **Task-Alignment:** The Judge Model must be trained on the same task (e.g., Speech Emotion Recognition) as the final evaluation model. This ensures that the gradients used for utility scoring are semantically meaningful and relevant to the downstream objective.

2. **Efficiency:** The Judge Model should be computationally efficient. We use a lightweight architecture, as the utility scoring step requires performing forward and backward passes on a large number of candidate segments.

Following these principles, the model is trained to convergence on the full dataset $\mathcal{D}$ using a standard cross-entropy loss. Once trained, its parameters are **frozen**, allowing it to serve as a fixed and consistent "teacher" for assessing the semantic utility of different segments. The specific architecture for each task is detailed in our experimental setup 3.1.

After obtaining the frozen judge model, we formally introduce the definition of Utility as follows.

Definition 1 (Gradient Flow)**.** Let $\mathcal{L}_t$ denote the cross-entropy loss of the model parameterized by $\theta^{(t)}$ at iteration t. The gradient flow computed on a mini-batch $\mathcal{B}$ is defined as:

$$\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x), y; \mathcal{B}) := \frac{\partial \mathcal{L}_t(f_{\theta^{(t)}}(x), y)}{\partial t}. \tag{11}$$

The gradient flow $\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x), y; \mathcal{B})$ characterizes the instantaneous rate of change of the loss for a given example (x, y) during training, providing a continuous-time approximation of the training dynamics. Unlike discrete stochastic gradient descent updates, which may introduce noise, the gradient flow offers a smooth analytical perspective to quantify the importance of data points. Leveraging this, we evaluate the effect of removing a single data point (x_i, y_i) and define a utility function below to serve as a pruning criterion.

Definition 2 (Utility)**.** Based on the gradient flow defined in Definition 1, consider a data point (x_i, y_i) in a dataset $\mathcal{D}$, and let $\mathcal{B} \subseteq \mathcal{D}$ be the mini-batch at iteration t. Define $\mathcal{B}_{\neg i} := \mathcal{B} \setminus \{(x_i, y_i)\}$.

The importance of (x_i, y_i) is measured by the maximal change in gradient flow over all relevant pairs:

$$\mathcal{U}(x_i, y_i; f_{\theta^{(t)}}) := \max_{(x_j, y_j) \in \mathcal{D}} \left| \dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}) - \dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}_{\neg i}) \right|.$$

This utility captures the *worst-case* impact of removing a single data point on the gradient flow, ensuring it accurately reflects the point's influence on training. By maximizing the difference in $\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B})$ across all $(x_j, y_j) \in \mathcal{D}$, the utility identifies samples with the greatest effect on training dynamics. This aligns with the objective of dataset pruning: retaining critical samples while discarding less impactful ones.

Definition 3 (Gradient Norm)**.** The gradient norm of a training example (x, y) for a model f parameterized by $\theta^{(t)}$ at iteration t is given by:

$$\|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x), y)\|. \tag{12}$$

Leveraging Definition 3, we show that the utility can be upper bounded by the gradient norm.

Theorem 4 (Utility is bounded by Gradient Norm, Proof in Appendix B)**.** *Let the utility function $\mathcal{U}$ be defined as in Definition 2. There exists a constant $c > 0$ such that:*

$$\mathcal{U}(x_i, y_i; f_{\theta^{(t)}}) \leq c \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_i), y_i)\|. \tag{13}$$

Based on Theorem 4, we efficiently approximate the utility of each sample via the upper bound provided by the gradient norm. This allows direct selection of the most influential samples by choosing those with the highest gradient norms.

After obtaining the judger f_{θ_T}, we assess the representativeness of each segment. Specifically, for each segment $\xi_{i,k}$, we compute the gradient norm to approximate its utility as follows:

$$s_{i,k} = \|\nabla_{\theta_T} \mathcal{L}\left(f_{\theta_T}(\xi_{i,k}), y_i\right)\|_2, \tag{14}$$

where $\mathcal{L}$ is the cross-entropy loss, and y_i is the corresponding ground truth label. In the implementation, we forward propagate and backpropagate for each segment individually, iterating over all model parameters, summing the squared gradients, and then taking the square root to obtain the final score. This score reflects the frame sensitivity to model parameter updates, *i.e.*, its semantic complexity and representativeness.

Next, we retain the most informative segment for each utterance based on the highest score:

$$\xi_i^\star = \arg\max_k s_{i,k}, \quad \phi_i^\star = \max_k s_{i,k}, \tag{15}$$

where $s_{i,k}$ is the score assigned to the k-th segment of the i-th utterance. This results in a candidate pool $\{(\xi_i^\star, y_i, \phi_i^\star)\}_{i=1}^{\tilde{m}}$. We then select the final subset $\tilde{\mathcal{D}}$ by ranking all $\xi_i^\star$ according to their scores and retaining the top-m segments:

$$\tilde{\mathcal{D}} = \text{Top-}m\left(\{(\xi_i^\star, y_i)\}_{i=1}^{\tilde{m}}, \text{ by } \phi_i^\star\right). \tag{16}$$

Based on Step 1 and Step 2, we obtain a compressed dataset that incorporates both utterance-level and frame-level pruning.

3 Experiments

3.1 Experimental Setup

Datasets. We conducted experiments on four datasets, including three corpora for emotion recognition and one speaker identification corpus, each corresponding to a distinct classification task. Specifically, we used the following datasets: (i) the M3ED dataset, (ii) the MEAD dataset, and (iii) the MELD dataset, all from the EmoBox (Ma et al., 2024a), where we selected four representative emotion categories—*Neutral*, *Angry*, *Happy*, and *Sad*—for classification; and (iv) the VoxCeleb1 dataset (Nagrani et al., 2017), from which we selected 10 speakers with the most data as a subset to construct a speaker identification task.

Coreset Size. To ensure a fair comparison, we evaluated performance under different ratios. For all methods, we chose 1, 5, 10, and 50 samples per class for each dataset, corresponding to coreset ratios. For example, choosing 1, 5, 10, and 50 samples per class corresponds to 0.036%, 0.18%, 0.363%, and 1.813% of MEAD dataset.

Baselines. We compared our approach against a range of widely-used data selection baselines, covering diversity-based, uncertainty-based, and influence-based strategies: (i) *Random*, which uniformly samples data from the full training set; (ii) *Herding*, a representativeness-based method that selects samples closest to the class mean; (iii) *K-Center* (Sener and Savarese, 2018), which maximizes the coverage radius by iteratively choosing the farthest point from current selections; (iv) *Least Confidence* (LC) (Coleman et al., 2019), which selects samples with the lowest predicted confidence; (v) *Entropy*, which ranks samples based on

Table 1: Results of WA (%) on MEAD, MELD and M3ED datasets across different coreset ratios.

Dataset	MEAD				MELD				M3ED			
Ratio (%)	0.036	0.18	0.363	1.813	0.049	0.243	0.485	2.426	0.026	0.131	0.261	1.307
Random	22.96±2.8	28.16±2.6	35.19±2.1	40.71±4.9	37.52±4.7	34.60±3.2	34.68±3.0	38.26±4.9	28.40±5.8	29.99±4.5	32.89±2.1	43.04±5.0
Herding	23.01±3.1	23.49±3.4	26.24±4.3	33.33±1.7	31.46±8.2	36.60±2.3	33.27±4.0	24.83±2.8	16.04±0.0	13.77±4.4	13.33±0.9	18.77±0.9
K-Center	30.29±2.6	32.17±2.2	38.06±4.4	50.18±3.9	35.76±9.2	31.99±5.0	31.22±2.6	38.11±1.1	34.03±6.54	29.73±0.9	33.76±6.1	36.77±2.0
Entropy	31.55±2.2	34.31±3.2	37.45±1.5	44.49±2.1	38.53±4.7	39.07±6.5	36.68±2.9	41.04±4.1	32.58±1.7	30.62±6.4	31.61±1.4	44.38±3.0
Craig	28.26±3.2	34.68±2.8	35.53±1.6	40.72±1.4	39.06±2.3	37.42±6.0	24.23±6.2	24.62±2.0	24.87±2.2	33.72±5.3	36.65±6.7	40.14±3.7
CD	28.30±4.4	32.85±3.5	35.24±1.5	42.39±0.5	30.73±3.2	35.37±2.7	37.10±7.0	30.71±3.1	38.07±3.8	40.67±0.8	38.57±5.9	38.88±4.0
Cal	32.01±3.7	31.97±4.0	38.32±2.4	44.60±1.4	29.52±8.0	50.85±5.1	55.22±1.0	57.41±0.1	44.90±5.4	44.45±5.3	46.57±3.0	53.17±1.2
GraND	22.52±5.3	27.42±6.4	24.85±4.1	31.10±3.5	22.80±10.2	41.53±6.3	27.47±4.6	22.06±1.0	35.50±5.9	28.65±6.4	38.65±2.4	39.72±6.2
Forgetting	28.06±1.4	34.51±0.0	36.86±2.9	31.25±0.1	38.22±5.9	33.02±4.4	34.07±5.8	41.64±6.3	31.54±5.9	33.21±3.1	31.50±0.4	33.35±1.5
LC	27.90±2.1	32.71±1.4	35.90±2.8	38.09±1.2	47.72±6.0	35.01±8.8	17.86±4.2	21.08±4.7	26.71±3.9	48.60±1.2	42.00±1.7	48.95±4.2
Margin	31.54±2.3	34.37±3.5	37.65±1.5	46.21±2.5	39.09±4.3	39.07±6.6	35.61±3.3	44.13±5.5	33.57±2.4	30.66±6.3	31.71±1.6	43.85±3.1
MelTrim	**44.28±0.6**	**51.84±1.7**	**53.41±0.6**	**64.67±0.8**	**57.55±0.4**	**58.10±0.3**	**58.89±0.3**	**60.14±0.2**	**52.01±1.1**	**52.78±0.0**	**53.84±0.7**	**55.76±0.8**
Whole	84.68±1.1				58.34±0.3				56.17±0.3			

Table 2: Results of UA (%) on MEAD, MELD and M3ED datasets across different coreset ratios.

Dataset	MEAD				MELD				M3ED			
Ratio (%)	0.036	0.18	0.363	1.813	0.049	0.243	0.485	2.426	0.026	0.131	0.261	1.307
Random	23.59±1.6	29.62±1.2	36.43±2.3	41.05±3.7	26.96±0.6	28.90±0.9	32.26±1.8	30.16±3.6	28.45±3.3	29.80±0.7	30.24±2.5	32.30±4.9
Herding	21.62±2.1	22.49±2.1	24.04±3.1	30.24±1.3	24.08±0.7	27.13±0.4	26.45±0.5	28.97±0.7	25.40±1.8	24.99±0.1	25.47±1.0	29.77±1.0
K-Center	29.99±2.2	32.28±1.8	38.71±3.6	49.45±3.0	27.83±1.9	29.86±0.7	29.79±1.3	30.73±1.7	27.75±3.6	29.50±2.1	31.25±2.3	33.22±1.7
Entropy	29.29±2.0	32.80±2.3	37.07±1.9	44.61±1.3	26.32±2.4	28.99±0.4	32.71±1.0	34.99±1.5	29.29±2.0	30.78±0.9	32.29±2.4	35.80±1.7
Craig	29.41±1.1	33.50±1.5	35.78±2.4	41.83±3.0	28.60±1.0	29.89±0.6	29.32±2.6	28.56±2.9	30.18±0.6	28.52±2.0	29.01±0.8	33.19±2.0
CD	28.69±3.2	30.54±1.3	34.75±2.1	43.11±1.3	27.82±1.9	30.28±1.6	28.64±3.6	33.99±2.0	27.60±2.1	32.81±1.1	33.36±0.9	33.96±0.2
Cal	30.31±2.3	29.55±3.4	35.27±2.7	40.01±1.2	26.20±0.3	27.96±0.9	28.39±0.9	28.63±2.2	26.69±0.9	28.75±1.7	29.09±1.0	35.21±0.2
GraND	27.41±0.4	30.63±2.6	32.44±2.7	36.40±1.5	27.13±1.9	30.38±0.6	30.07±1.1	30.75±2.1	28.79±1.3	28.31±3.0	30.15±1.1	32.53±2.1
Forgetting	26.42±1.6	32.91±1.5	33.51±1.9	35.51±0.9	26.86±0.4	30.71±0.6	31.40±0.2	34.13±1.3	24.60±1.1	28.35±0.7	27.18±1.3	28.70±0.7
LC	27.37±1.6	29.43±1.0	31.48±2.0	37.73±0.9	26.44±1.1	29.35±1.0	26.68±2.3	32.91±1.4	25.10±2.2	24.98±0.1	24.52±0.3	27.77±0.2
Margin	29.24±1.8	32.75±2.5	37.16±1.9	45.15±2.1	26.30±2.4	28.95±0.4	33.56±1.1	33.23±1.4	29.10±1.1	30.42±1.3	32.30±2.4	36.47±1.7
MelTrim	**39.79±1.9**	**50.17±0.8**	**51.72±2.1**	**62.64±0.4**	**34.73±0.5**	**37.81±0.6**	**38.44±1.5**	**42.04±0.4**	**38.16±0.3**	**40.62±0.3**	**41.92±0.4**	**44.80±0.6**
Whole	82.86±0.9				40.20±0.9				41.89±0.3			

Table 3: Results of F1 (%) on MEAD, MELD and M3ED datasets across different coreset ratios.

Dataset	MEAD				MELD				M3ED			
Ratio (%)	0.036	0.18	0.363	1.813	0.049	0.243	0.485	2.426	0.026	0.131	0.261	1.307
Random	21.40±1.2	28.59±1.2	35.60±1.5	39.81±4.3	23.41±3.5	26.31±0.9	28.41±1.4	28.67±4.9	22.96±4.6	26.33±1.8	28.62±1.4	28.78±6.9
Herding	18.63±1.8	21.43±1.5	21.92±3.1	30.59±1.0	17.51±1.7	27.89±0.4	24.56±1.5	23.51±1.3	16.73±1.4	7.59±1.5	13.78±3.2	18.15±2.1
K-Center	27.77±3.2	31.27±1.9	36.44±3.9	48.50±3.3	25.45±2.9	25.50±2.2	26.09±1.7	27.20±3.3	23.86±0.7	26.52±1.1	28.17±3.1	32.34±1.5
Entropy	26.06±2.5	32.24±2.8	35.61±1.5	43.44±1.7	24.18±1.2	27.20±1.5	30.19±1.7	33.01±1.7	26.06±2.5	27.29±2.1	29.05±0.7	35.04±1.4
Craig	26.30±1.1	32.53±1.9	33.84±2.5	38.80±2.1	26.87±1.2	26.42±0.7	22.03±5.8	23.98±1.6	22.77±0.8	24.52±2.0	27.54±2.9	32.56±1.5
CD	26.07±4.3	30.53±1.3	32.84±1.7	41.09±0.8	23.73±1.7	24.95±1.7	25.09±3.9	28.58±1.5	23.63±3.5	31.95±0.6	32.10±1.3	32.35±0.1
Cal	27.25±3.5	27.41±4.1	32.83±2.2	38.49±1.2	20.10±3.8	25.99±0.4	26.16±1.6	25.55±3.6	22.95±1.6	27.10±1.7	28.14±0.7	31.58±1.5
GraND	18.47±4.2	21.22±5.3	21.36±5.5	30.18±3.2	15.75±4.5	26.06±0.2	21.89±3.3	20.99±0.7	24.38±3.3	24.74±1.7	29.14±0.7	28.69±3.4
Forgetting	22.49±1.0	31.58±1.3	32.35±2.0	30.04±0.9	24.71±3.1	27.01±1.2	28.53±2.0	31.63±0.3	23.00±1.5	26.79±1.2	26.26±0.9	27.91±1.0
LC	24.04±3.0	27.88±0.7	29.43±2.4	36.44±1.6	21.31±1.9	22.57±6.4	13.88±2.1	20.03±2.1	20.10±2.3	11.00±3.3	15.17±4.8	16.37±1.7
Margin	26.22±2.6	32.35±3.0	35.75±1.5	44.61±2.1	24.39±1.4	27.15±1.4	30.16±1.6	32.46±1.6	26.01±2.3	27.05±2.2	29.09±0.7	35.08±1.7
MelTrim	**38.60±2.4**	**49.58±0.8**	**51.09±1.4**	**61.90±0.1**	**32.65±2.5**	**35.79±0.5**	**34.94±1.3**	**39.45±0.5**	**36.25±0.5**	**37.97±0.6**	**39.21±0.6**	**45.13±0.5**
Whole	82.57±1.2				39.41±0.9				38.90±0.6			

output entropy; (vi) *Margin*, which selects based on the margin between top two predicted probabilities; (vii) *Craig* (Mirzasoleiman et al., 2020), which solves a submodular optimization problem to approximate the optimal core-set; (viii) *Contextual Diversity* (CD) (Agarwal et al., 2020), which selects samples that maximize diversity in representation space; (ix) *Cal* (Margatina et al., 2021), a calibration-based strategy that selects samples with high calibration error; (x) *GraND* (Paul et al., 2022), which uses gradient norm magnitude to prioritize impactful examples; (xi) *Forgetting* (Toneva et al., 2019), which selects samples that are frequently forgotten during training.

Judge Model. To score frame-level utility, we adopt a lightweight Judge Model that is trained only for a few initial epochs and then frozen, instead of being optimized to full convergence. For speech emotion recognition we use a Whisper-tiny backbone with a randomly-initialized classification head trained for a small number of epochs (7 on M3ED/MEAD/MELD) and subsequently frozen. For speaker identification we use a RawNet3 (Jung et al., 2022b) model consistent with the final backbone. This strategy yields task-aligned gradients at negligible cost and enables efficient pruning without requiring a fully converged state-of-the-art model.

Evaluation. For each setting, we repeated the training and evaluation 10 times to ensure statistical reliability. For the emotion classification task, we reported Weighted Accuracy (WA), Unweighted Accuracy (UA), and F1-score as evaluation metrics. For the speaker identification task, we evaluated

Table 4: Equal Error Rate (EER) results (%) on the VoxCeleb1 dataset. Note that lower EER values indicate better performance.

Method	Coreset Ratio			
	0.261%	1.303%	2.606%	13.031%
Herding	37.04±0.5	30.63±2.5	24.21±3.3	11.83±0.6
Forgetting	36.54±0.6	31.25±2.3	23.46±1.2	8.25±0.1
Cal	35.71±0.4	26.31±0.9	19.69±0.1	10.81±0.2
MelTrim	**31.38±0.4**	**19.79±0.8**	**12.67±1.6**	**4.79±0.3**
Whole	2.38±0.3			

the models using Equal Error Rate (EER).

3.2 Main Results and Key Observations

Emotion Classification. We first present the experimental results across the MEAD, MELD, and M3ED datasets under different coreset ratios, evaluated by Weighted Accuracy (WA), Unweighted Accuracy (UA), and F1-score metrics. As shown in Table 1 2 3, MelTrim consistently outperforms all baseline methods across all datasets and evaluation metrics. For instance, on the MEAD dataset at the coreset ratio of 1.813%, MelTrim achieves 64.67% in WA, significantly surpassing the best-performing baseline method K-Center, which achieves 50.18%. Notably, on the MELD dataset we observe an unexpected drop in WA for most baselines when the selection ratio increases from 0.243% to 0.485%. This decline suggests that, beyond a certain point, additional samples selected by methods like GraND, LC or Craig may introduce noisy or poorly representative examples—particularly in MELD's challenging dialogue context—thereby outweighing the gains from added coverage. In contrast, MelTrim maintains stronger performance across all ratios.

Speaker Identification. Table 4 reports the speaker identification results evaluated by Equal Error Rate (EER) metric, based on different selection ratios applied to the predefined subset of the VoxCeleb1 dataset. We adopt RawNet3 (Jung et al., 2022b) as the backbone model for this task.

Across all ratios, MelTrim consistently achieves the lowest EER among all subset selection strategies. Notably, MelTrim reduces the EER to 31.38% at 0.261% ratio, significantly outperforming Herding (37.04%), Forgetting (36.54%), and Cal (35.71%). As the ratio increases, the advantage of MelTrim remains consistent. At 13.031% data usage, MelTrim achieves 4.79% EER, closing the gap with the full-data baseline (2.38%), while other methods still exhibit higher error rates (*e.g.*, Herding: 11.83%). These results demonstrate that MelTrim is more effective at selecting informative speech segments for speaker identification.

Automatic speech recognition. We additionally evaluate MelTrim on the MELD dataset for ASR, using Word Error Rate (WER) as the metric. With a Wav2Vec judge and CTC-based scoring, MelTrim achieves consistent improvements over baselines across different coreset ratios. Detailed results are reported in Appendix C.

3.3 Ablation Study

Ablations on Feature Space and Judge Objective. We validate two design hypotheses underlying MelTrim's coarse-to-fine scheme. *(i) Coarse filtering needs acoustic fidelity.* Replacing MFCCs in Step 1 with penultimate-layer Whisper-tiny embeddings—while keeping the pipeline unchanged—consistently degrades performance on MEAD and M3ED, as shown in Table 5, indicating that semantically invariant deep features induce representation collapse for diversity-based sampling. *(ii) Fine pruning requires a semantically grounded Judge.* Substituting the ASR-pretrained Whisper-tiny Judge with a self-supervised Wav2Vec 2.0 Judge yields less stable and often inferior results. Table 6 shows that gradients tied to content semantics are critical for utility scoring.

Ablation on Scoring Metrics. To assess the impact of scoring strategies on segment selection, we compared two approaches: scoring based on (i) utility and scoring based on (ii) cross-entropy loss. As shown in Table 7, the utility-based method outperforms cross-entropy across multiple evaluation metrics, indicating its superior effectiveness in identifying informative segments.

Table 7: Ablation study of different scoring methods. We demonstrate results of WA (%) on MEAD dataset.

Scoring Method	Coreset Ratio			
	0.036%	0.18%	0.363%	1.813%
Loss	39.23±0.00	49.16±1.16	52.83±1.62	61.27±0.50
Utility	**44.28±0.59**	**51.84±1.67**	**53.41±0.59**	**64.67±0.75**

Time and Efficiency Analysis. We quantify MelTrim's computational benefit by reporting wall-clock time for each pruning stage and the end-to-end pipeline (Table 8). Although pruning introduces a one-time selection cost (MFCC extraction, clustering, Judge pre-training, utility scoring), the much smaller coreset yields substantially shorter training, making the *total* time lower than full-data training. On MELD, end-to-end time drops from 3360.0s to 1313.9s (-60.9%). Thus, MelTrim delivers tangible efficiency gains in addition to accu-

Table 5: Ablation study on the choice of features for utterance-level coarse filtering. We compare using MFCCs against deep features from Whisper. The results demonstrate the superiority of MFCCs in preserving acoustic diversity, leading to better downstream performance across all metrics.

Dataset	**MEAD**				**M3ED**			
Ratio (%)	**0.036**	**0.18**	**0.363**	**1.813**	**0.026**	**0.131**	**0.261**	**1.307**
Metric: Weighted Accuracy (WA)								
MFCC	**44.28±0.59**	**51.84±1.67**	**53.41±0.59**	**64.67±0.75**	**52.01±1.05**	**52.78±0.03**	53.84±0.65	**55.76±0.80**
Whisper	35.47±0.96	51.75±3.76	50.99±3.32	61.43±1.32	48.83±2.95	51.81±0.97	**54.00±0.53**	54.71±0.67
SOTA Baseline	31.55±2.2	34.68±2.8	38.32±2.4	50.18±3.9	44.90±5.4	48.60±1.2	46.57±3.0	53.17±1.2
Metric: Unweighted Accuracy (UA)								
MFCC	**39.79±1.93**	50.17±0.78	51.72±2.14	**62.64±0.37**	**38.16±0.32**	**40.62±0.27**	**41.92±0.39**	**44.80±0.64**
Whisper	37.69±0.68	**50.68±1.88**	**51.98±1.52**	60.37±1.53	35.39±2.12	38.56±0.36	39.54±0.42	41.69±1.32
SOTA Baseline	30.31±2.3	33.50±1.5	38.71±3.6	49.45±3.0	30.18±0.6	30.78±0.9	33.36±0.9	36.47±1.7
Metric: F1-score (F1)								
MFCC	**38.60±2.40**	**49.58±0.82**	**51.09±1.39**	**61.90±0.09**	**36.25±0.51**	**37.97±0.56**	**39.21±0.63**	**45.13±0.49**
Whisper	31.65±1.42	49.35±2.66	49.47±3.03	59.45±1.44	32.89±0.59	36.29±0.69	38.09±0.99	42.10±1.12
SOTA Baseline	27.77±3.2	32.53±1.9	36.44±3.9	48.50±3.3	26.06±2.5	27.29±2.1	32.10±1.3	35.08±1.7

Table 6: We assess the Judge Model's pre-training objective by replacing our default ASR-pretrained Whisper-tiny with a self-supervised Wav2Vec 2.0 model. The resulting performance degradation highlights the importance of strong semantic grounding for effective utility-based pruning. Note that the "SOTA Baseline" is a composite baseline, representing the best result achieved by any of the evaluated baseline methods from previous experiments for each specific setting.

Method	**MEAD**				**MELD**				**M3ED**			
	0.036%	**0.18%**	**0.363%**	**1.813%**	**0.049%**	**0.243%**	**0.485%**	**2.426%**	**0.026%**	**0.131%**	**0.261%**	**1.307%**
Metric: Weighted Accuracy (WA)												
SOTA Baseline	31.55	34.68	38.32	50.18	47.72	50.85	55.22	57.41	44.90	48.60	46.57	53.17
MelTrim (Judge: Wav2Vec)	32.43	40.67	34.09	57.73	56.81	57.21	57.67	59.79	52.01	52.78	53.84	55.76
MelTrim (Judge: Whisper-tiny)	**44.28**	**51.84**	**51.84**	**64.67**	**57.55**	**58.10**	**58.89**	**60.14**	**52.27**	**55.77**	**55.09**	**55.31**
Metric: Unweighted Accuracy (UA)												
SOTA Baseline	30.31	33.50	38.71	49.45	28.60	30.71	33.56	34.13	30.18	30.78	33.36	36.47
MelTrim (Judge: Wav2Vec)	32.11	36.62	32.67	56.12	25.00	26.62	27.09	32.92	30.36	37.29	37.54	41.02
MelTrim (Judge: Whisper-tiny)	**39.79**	**50.17**	**51.72**	**62.64**	**34.73**	**37.81**	**38.44**	**42.04**	**38.16**	**40.62**	**41.92**	**44.80**
Metric: F1-score												
SOTA Baseline	27.77	32.53	36.44	48.50	26.87	27.89	30.19	33.01	26.06	27.29	32.10	35.08
MelTrim (Judge: Wav2Vec)	29.33	33.96	26.29	54.60	18.11	21.84	22.68	31.34	28.96	34.76	33.74	39.06
MelTrim (Judge: Whisper-tiny)	**38.60**	**49.58**	**51.09**	**61.90**	**32.65**	**35.79**	**34.94**	**39.45**	**36.25**	**37.97**	**39.21**	**45.13**

Table 8: Wall-clock time (seconds) for the MelTrim pruning process and end-to-end training. Although selection incurs a one-time cost, the total time is substantially reduced vs. full-data training (↓). All timings were measured on a single NVIDIA H20 (NVLink) GPU.

Stage / Task	**MELD**	**M3ED**	**MEAD**
Pruning Stage Breakdown (One-time Cost)			
MFCC Extraction (Step 1)	130.86	84.59	220.38
Clustering (UMAP & DBSCAN) (Step 1)	126.69	166.13	91.28
Judge Model Pre-training (Step 2)	587.99	228.00	2100.02
GradNorm Scoring (Step 2)	293.34	257.62	278.35
Total Selection Cost	**1138.88**	**736.34**	**2690.03**
End-to-End Time (Selection Cost + Training Cost)			
+ Training on 0.049% / 0.026% / 0.036% Coreset	1241.96 ↓63.0%	881.78 ↓61.3%	2842.69 ↓32.3%
+ Training on 0.243% / 0.131% / 0.180% Coreset	1246.40 ↓62.9%	887.02 ↓61.1%	2847.52 ↓32.2%
+ Training on 0.485% / 0.261% / 0.363% Coreset	1256.76 ↓62.6%	895.27 ↓60.7%	2855.83 ↓32.0%
+ Training on 2.426% / 1.307% / 1.813% Coreset	1313.87 ↓60.9%	949.46 ↓58.3%	2910.98 ↓30.7%
Baseline			
Whole Data Training (40 epochs)	3359.96	2280.00	4200.03

racy improvements.

4 Conclusion

We present MelTrim, a novel dataset pruning framework tailored specifically for speech data, employing a two-stage coarse-to-fine approach that effectively addresses both inter-sample and intra-sample redundancies. By integrating utterance-level clustering with frame-level utility-based selection, MelTrim significantly reduces dataset size while preserving the critical information necessary for robust model training. Extensive experiments demonstrate that MelTrim surpasses general-purpose pruning baselines, achieving a 49.5% improvement in weighted accuracy on the M3ED dataset for Speech Emotion Recognition (SER) and a 41.9% reduction in EER on the VoxCeleb1 dataset for Speaker Identification (SID). Beyond accuracy, MelTrim delivers significant end-to-end efficiency improvements and exhibits a promising weak-to-strong transfer effect across model scales. These results highlight the effectiveness and broad applicability of our method in improving data efficiency for large-scale speech classification.

5 Limitations

While MelTrim is designed to be task-agnostic and operates effectively across various speech classification tasks, the current study focuses solely on unimodal acoustic features, without incorporating auxiliary modalities like transcriptions. Our future work will explore multimodal pruning strategies that leverage richer contextual information to improve redundancy estimation.

6 Broader Impact and Ethical Considerations

Our proposed framework, MelTrim, aims to improve the efficiency and scalability of training speech models through intelligent data reduction. By reducing dataset size without compromising downstream performance, it can substantially lower the computational cost and carbon footprint associated with training large-scale models, aligning with growing concerns over the environmental sustainability of deep learning. The framework's modular design also supports plug-and-play integration with various Judge Models, enabling broad applicability across speech domains, including emotion recognition and speaker identification.

Despite these positive impacts, the use of a utility-driven pruning mechanism introduces critical ethical considerations. The primary concern is that the selection process relies on a Judge Model trained on downstream tasks (e.g., ASR or SER), which may inadvertently encode or reinforce demographic, acoustic, or affective biases present in the training data. This could lead to the disproportionate pruning of data from underrepresented speakers, marginalized communities, or rare emotional states, particularly if the utility model exhibits uneven generalization. Furthermore, pruning speech data at a fine granularity (e.g., frame-level) raises potential challenges in ensuring speaker privacy and contextual coherence.

To mitigate these risks in our work, all experiments were conducted on publicly available, anonymized benchmark datasets with demographic documentation (e.g., M3ED, VoxCeleb1). For real-world deployment, we believe that combining utility-based selection with transparent documentation and rigorous, fairness-oriented audits is crucial. Future work should investigate fairness-aware pruning strategies to ensure the equitable retention of speech content across diverse speaker attributes (e.g., age, gender, language variety), thereby fostering responsible, inclusive, and sustainable speech model development.

7 Acknowledge

This work was supported by Alibaba Group through Alibaba Innovative Research Program.

References

Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. 2020. Contextual diversity for active learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 137–153. Springer.

Keyu An, Qian Chen, Chong Deng, Zhihao Du, Changfeng Gao, Zhifu Gao, Yue Gu, Ting He, Hangrui Hu, Kai Hu, Shengpeng Ji, Yabin Li, Zerui Li, Heng Lu, Haoneng Luo, Xiang Lv, Bin Ma, Ziyang Ma, Chongjia Ni, and 14 others. 2024. Funaudiollm: Voice understanding and generation foundation models for natural interaction between humans and llms. *Preprint*, arXiv:2407.04051.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Seung bin Kim, Chan yeong Lim, Jungwoo Heo, Ju ho Kim, Hyun seo Shin, Kyo-Won Koo, and Ha-Jin Yu. 2024. Mr-rawnet: Speaker verification system with multiple temporal resolutions for variable duration utterances using raw waveforms. In *Interspeech 2024*, pages 2125–2129.

Cheol Jun Cho, Nicholas Lee, Akshat Gupta, Dhruv Agarwal, Ethan Chen, Alan W Black, and Gopala K Anumanchipalli. 2024. Sylber: Syllabic embedding representation of speech from raw audio. *arXiv preprint arXiv:2410.07168*.

Yunfei Chu, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, Yuanjun Lv, Jinzheng He, Junyang Lin, Chang Zhou, and Jingren Zhou. 2024. Qwen2-audio technical report. *CoRR*, abs/2407.10759.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. 2019. Selection via proxy: Efficient data selection for deep learning. In *Int. Conf. Learn. Represent.*

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. Arcface: Additive angular margin loss for deep face recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694.

Muqing Deng, Tingting Meng, Jiuwen Cao, Shimin Wang, Jing Zhang, and Huijie Fan. 2020. Heart sound classification based on improved mfcc features and convolutional recurrent neural networks. *Neural Networks*, 130:22–32.

Brecht Desplanques, Jenthe Thienpondt, and Kris Demuynck. 2020. Ecapa-tdnn: Emphasized channel attention, propagation and aggregation in tdnn based speaker verification. In *Interspeech 2020*, pages 3830–3834.

Moataz El Ayadi, Mohamed S Kamel, and Fakhri Karray. 2011. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern recognition*, 44(3):572–587.

Moataz MH El Ayadi, Mohamed S Kamel, and Fakhri Karray. 2007. Speech emotion recognition using gaussian mixture vector autoregressive models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–957. IEEE.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460.

Shujie Hu, Long Zhou, Shujie Liu, Sanyuan Chen, Lingwei Meng, Hongkun Hao, Jing Pan, Xunying Liu, Jinyu Li, Sunit Sivasankaran, Linquan Liu, and Furu Wei. 2024. Wavllm: Towards robust and adaptive speech large language model. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 4552–4572. Association for Computational Linguistics.

Dias Issa, M Fatih Demirci, and Adnan Yazici. 2020. Speech emotion recognition with deep convolutional neural networks. *Biomedical Signal Processing and Control*, 59:101894.

Jee-weon Jung, You Jin Kim, Hee-Soo Heo, Bong-Jin Lee, Youngki Kwon, and Joon Son Chung. 2022a. Pushing the limits of raw waveform speaker recognition. *Preprint*, arXiv:2203.08488.

Jee-weon Jung, You Jin Kim, Hee-Soo Heo, Bong-Jin Lee, Youngki Kwon, and Joon Son Chung. 2022b. Pushing the limits of raw waveform speaker recognition. *Proc. Interspeech*.

Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. 2021. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *Proc. Int. Conf. Mach. Learn.*, pages 5464–5474.

Siddique Latif, Rajib Rana, Shahzad Younis, Junaid Qadir, and Julien Epps. 2018. Transfer learning for improving speech emotion classification accuracy. *arXiv preprint arXiv:1801.06353*.

Aristidis Likas, Nikos Vlassis, and Jakob J Verbeek. 2003. The global k-means clustering algorithm. *Pattern recognition*, 36(2):451–461.

Xiaoqian Liu, Guoqiang Hu, Yangfan Du, Erfeng He, Yingfeng Luo, Chen Xu, Tong Xiao, and Jingbo Zhu. 2024. Recent advances in end-to-end simultaneous speech translation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 8142–8150. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Ziyang Ma, Mingjie Chen, Hezhao Zhang, Zhisheng Zheng, Wenxi Chen, Xiquan Li, Jiaxin Ye, Xie Chen, and Thomas Hain. 2024a. Emobox: Multilingual multi-corpus speech emotion recognition toolkit and benchmark. *Preprint*, arXiv:2406.07162.

Ziyang Ma, Zhisheng Zheng, Jiaxin Ye, Jinchao Li, Zhifu Gao, ShiLiang Zhang, and Xie Chen. 2024b. emotion2vec: Self-supervised pre-training for speech emotion representation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15747–15760, Bangkok, Thailand. Association for Computational Linguistics.

Katerina Margatina, Giorgos Vernikos, Loïc Barrault, and Nikolaos Aletras. 2021. Active learning by acquiring contrastive examples. *arXiv preprint arXiv:2109.03764*.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *Proc. Int. Conf. Mach. Learn.*, pages 6950–6960.

Edmilson Morais, Ron Hoory, Weizhong Zhu, Itai Gat, Matheus Damasceno, and Hagai Aronowitz. 2022. Speech emotion recognition using self-supervised features. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6922–6926. IEEE.

A. Nagrani, J. S. Chung, and A. Zisserman. 2017. Voxceleb: a large-scale speaker identification dataset. In *INTERSPEECH*.

Muhammet Pakyurek, Mahir Atmis, Selman Kulac, and Umut Uludag. 2020. Extraction of novel features based on histograms of mfccs used in emotion classification from generated original speech dataset. *Elektronika ir Elektrotechnika*, 26(1):46–51.

Mansheej Paul, Brett Larsen, Surya Ganguli, Jonathan Frankle, and Gintare Karolina Dziugaite. 2022. Lottery tickets on a data diet: Finding initializations with sparse trainable networks. In *Adv. Neural Inform. Process. Syst.*, volume 35, pages 18916–18928.

Karl Pearson. 1901. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

Alexander Platzer. 2013. Visualization of snps with t-sne. *PloS one*, 8(2):e56883.

Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. 2020. Identifying mislabeled data using the area under the margin ranking. In *Adv. Neural Inform. Process. Syst.*, volume 33, pages 17044–17056.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2019. An empirical study of example forgetting during deep neural network learning. In *Int. Conf. Learn. Represent.*

Daren Wang, Xinyang Lu, and Alessandro Rinaldo. 2017. Dbscan: Optimal rates for density based clustering. *arXiv preprint arXiv:1706.03113*.

Max Welling. 2009. Herding dynamical weights to learn. In *Proc. Int. Conf. Mach. Learn.*, pages 1121–1128.

Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. 2022. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *Int. Conf. Learn. Represent.*

Chen Xu, Yuhao Zhang, Chengbo Jiao, Xiaoqian Liu, Chi Hu, Xin Zeng, Tong Xiao, Anxiang Ma, Huizhen Wang, and Jingbo Zhu. 2023. Bridging the granularity gap for acoustic modeling. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 10816–10833. Association for Computational Linguistics.

Min Xu, Ling-Yu Duan, Jianfei Cai, Liang-Tien Chia, Changsheng Xu, and Qi Tian. 2004. Hmm-based audio keyword generation. In *Pacific-Rim Conference on Multimedia*, pages 566–574. Springer.

A Related Works

Data Pruning. Dataset pruning, or coreset selection, aims to reduce the size of a dataset by identifying and selecting the most informative samples based on predefined criteria. Several approaches, such as Herding (Welling, 2009) and Moderate (Xia et al., 2022), measure the distance between samples in feature space to identify relevant data points. Entropy-based methods (Coleman et al., 2019) focus on the uncertainty and decision boundaries of predicted outputs to assess sample importance. GradND (Paul et al., 2022) evaluates sample importance by analyzing its gradient norm and loss difference norm, respectively. In addition, gradient-matching methods such as GradMatch (Killamsetty et al., 2021) and Craig (Mirzasoleiman et al., 2020) minimize the distance between gradients computed for the full dataset and the coreset. Moreover, methods like Forgetting (Toneva et al., 2019) and AUM (Pleiss et al., 2020) have shown the potential of leveraging training dynamics to assess sample importance. Despite their effectiveness in various domains, these methods overlook the unique characteristics of speech data, rendering them ill-suited for addressing the specific challenges posed by speech recognition tasks.

Speech Emotion Recognition. Speech Emotion Recognition (SER) aims to identify and classify emotions expressed in speech, using acoustic, prosodic, and sometimes linguistic features. Early methods (El Ayadi et al., 2011) primarily relied on handcrafted features like Mel-frequency cepstral coefficients (MFCC), pitch, and intensity, which were then classified using techniques such as Gaussian mixture models (GMM) (El Ayadi et al., 2007). However, with the rise of deep learning, more sophisticated techniques such as Convolutional Neural Networks (CNN) based methods (Issa et al., 2020) have emerged, offering superior performance by automatically extracting features and capturing temporal dependencies in speech. Transfer learning (Latif et al., 2018) and self-supervised learning (Morais et al., 2022) have also shown promise in addressing data limitations and improving generalization across different domains.

Speaker Identification. Speaker Identification (SID) aims to recognize an individual speaker based on unique characteristics embedded in their speech. In recent years, deep learning-based methods have become dominant, moving beyond traditional statistical approaches. End-to-end neural architectures have shown great promise by automatically learning speaker-discriminative features from audio. ECAPA-TDNN (Desplanques et al., 2020) integrates attention-based channel modeling and multi-scale representations to improve robustness under challenging acoustic conditions. RawNet3 (Jung et al., 2022b) achieves competitive results by integrating learnable filter banks (ParamFBank), Res2Net-based residual blocks, and hierarchical feature aggregation. Trained with Additive Angular Margin Loss (Deng et al., 2019). RawNet3 outperforms many spectrogram-based methods. Extensions such as MR-RawNet (bin Kim et al., 2024) introduce multi-resolution encoding to enhance robustness to short utterances. With the development of large-scale speaker datasets and more sophisticated model architectures, speaker identification systems have become increasingly accurate and robust, showing strong generalization across diverse acoustic conditions.

B Proof of Theorem 4

Proof of Theorem 4. Using the chain rule for gradient flow, we have

$$\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}) = \nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j) \cdot \frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}},$$

and similarly for $\mathcal{B}_{\neg i}$. Thus, the change in gradient flow is

$$\begin{aligned}
&\left|\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}) - \dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}_{\neg i})\right| \\
&= \left|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j) \cdot \left(\frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}} - \frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}_{\neg i}}\right)\right|.
\end{aligned}$$

Under SGD with learning rate η, the update step is

$$\frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}} = -\eta \sum_{(x,y)\in\mathcal{B}} \nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x), y).$$

Removing (x_i, y_i) gives

$$\frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}_{\neg i}} = -\eta \sum_{(x,y)\in\mathcal{B}_{\neg i}} \nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x), y).$$

Taking the difference,

$$\frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}} - \frac{\partial \theta^{(t)}}{\partial t}\Big|_{\mathcal{B}_{\neg i}} = -\eta \nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_i), y_i).$$

Substituting this into the gradient flow change gives

$$\begin{aligned}
&\left|\dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}) - \dot{\mathcal{L}}_t(f_{\theta^{(t)}}(x_j), y_j; \mathcal{B}_{\neg i})\right| \\
&= \eta \left|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j) \cdot \nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_i), y_i)\right| \\
&\leq \eta \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j)\| \cdot \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_i), y_i)\|
\end{aligned}$$

where the last step follows from the Cauchy–Schwarz inequality. Let

$$c = \eta \max_{(x_j, y_j) \in \mathcal{D}} \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j)\|$$

be a constant independent of (x_i, y_i). Taking the maximum over $(x_j, y_j) \in \mathcal{D}$, we obtain

$$\mathcal{U}(x_i, y_i; f_{\theta^{(t)}}) \leq c \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_i), y_i)\|.$$

Note that $c = \eta \max_{(x_j, y_j) \in \mathcal{D}} \|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j)\|$ satisfies the following properties:

1. It is independent of the current measured data (x_i, y_i), ensuring that the bound in Theorem 4 holds uniformly for all training examples.
2. It only assumes that the gradient norm $\|\nabla_{\theta^{(t)}} \mathcal{L}_t(f_{\theta^{(t)}}(x_j), y_j)\|$ has an upper bound, which is a reasonable assumption for any successfully converged model.
3. Since the learning rate η can be chosen to be small in practice, the value of c remains controlled and does not become excessively large.

Parameter Settings 4.

C ASR Results in MELD dataset

Table 9: Word Error Rate (WER) results (%) on the MELD dataset. Note that lower WER values indicate better performance.

Method	Coreset Ratio		
	1%	5%	10%
Random	59.60±0.34	60.05±0.98	60.08±1.39
kCenterGreedy	58.04±0.47	58.70±0.77	56.44±0.64
Herding	59.78±0.05	58.84±0.61	57.29±0.20
ContextualDiversity	60.32±0.98	59.33±0.82	56.55±1.03
MelTrim	**56.24±0.27**	**55.53±0.36**	**53.72±0.33**
Whole	46.00±0.24		

D Ablation Studies on Method Variants

Cross-Architecture Performance. Table 10 presents an evaluation of our method's robustness and generalizability across different model scales. We report the Weighted Accuracy (WA) on the M3ED dataset for four Whisper encoder variants: tiny (T), base (B), small (S), and medium (M). All pruned datasets in this evaluation were generated using the same fixed Judge Model.

Notably, our main results by default utilize the `Whisper-T` model. However, we observe that pruning with the `Whisper-T` model also benefits larger-scale networks, suggesting a promising weak-to-strong transfer potential. Across all settings, MelTrim consistently achieves the best WA among all compared methods. Under the 1.3% selection ratio, MelTrim reaches 56.35% WA on the `Whisper-M` model, significantly outperforming Herding (50.65%), Forgetting (46.28%), and Cal (49.20%). Similar improvements are observed across the `Whisper-B` and `Whisper-S` variants at every ratio. These results demonstrate that MelTrim maintains high effectiveness across Whisper architectures of varying scales, validating its robustness and generalization ability.

Table 10: WA (%) results on M3ED dataset across Whisper architectures and coreset ratios.

Ratio (%)	Method	Whisper-T	Whisper-B	Whisper-S	Whisper-M
0.026	Random	28.40±5.76	37.32±0.48	37.28±4.48	45.94±1.15
	Herding	16.04±0.00	39.87±0.13	41.68±2.47	49.57±1.05
	Forgetting	31.54±5.93	36.32±6.70	37.93±2.42	44.91±0.72
	Cal	38.07±3.77	38.67±3.13	46.71±2.24	50.01±0.41
	MelTrim	**52.01±1.05**	**48.55±3.07**	**50.89±1.54**	**50.52±1.53**
0.131	Random	29.99±4.48	37.72±4.82	36.77±3.98	37.63±3.03
	Herding	13.77±4.41	39.73±2.05	44.21±1.89	50.65±0.55
	Forgetting	33.21±3.06	42.33±4.35	45.56±2.11	46.28±0.52
	Cal	40.67±0.75	42.55±6.35	46.29±2.39	49.20±0.69
	MelTrim	**52.78±0.03**	**49.63±1.11**	**50.70±0.70**	**56.35±0.51**
0.261	Random	32.89±2.15	34.79±5.01	37.86±4.93	38.30±5.92
	Herding	13.33±0.90	41.28±2.34	46.80±1.65	51.42±2.72
	Forgetting	31.50±0.40	43.58±6.35	45.61±1.91	47.39±2.24
	Cal	38.57±5.86	45.61±5.13	46.67±3.18	50.24±0.41
	MelTrim	**53.84±0.65**	**50.61±1.59**	**52.16±1.27**	**55.28±0.99**
1.307	Random	43.04±5.06	47.19±1.00	49.02±2.10	47.39±1.03
	Herding	18.77±0.85	41.88±0.97	49.90±0.60	50.68±0.41
	Forgetting	33.35±1.54	46.98±2.47	47.33±1.18	47.38±0.45
	Cal	38.88±4.00	52.12±0.70	50.42±4.43	55.00±0.35
	MelTrim	**55.76±0.80**	**53.47±0.33**	**51.94±0.88**	**55.53±1.51**
Whole Dataset		54.11±1.82	57.17±0.20	58.95±0.38	60.24±0.31

Comparison of Dimensionality Reduction Strategies. To evaluate the effectiveness of dimensionality reduction in coarse filtering, we replaced the default UMAP module with two commonly used alternatives: PCA (Pearson, 1901) and T-SNE (Platzer, 2013). All other components are kept unchanged during the evaluation. As shown in Table 11, neither PCA nor T-SNE outperforms UMAP in any evaluation metric. This indicates that UMAP is more effective at preserving the structural information of MFCC in the low-dimensional embedding space, which in turn enhances data quality.

Table 11: Ablation study of dimensionality reduction methods. We show WA (%) results on MELD dataset.

Method	Coreset Ratio		
	0.049%	0.243%	0.485%
PCA	37.90±6.52	45.11±8.26	47.05±7.30
T-SNE	40.77±14.94	46.27±15.78	48.11±10.67
UMAP	**57.55±0.42**	**58.10±0.25**	**58.89±0.25**

Comparison of Clustering Strategies. To further evaluate the impact of clustering strategies in Step

1, we replaced the default DBSCAN algorithm with two alternative methods: KNN-based selection and K-Means clustering (Likas et al., 2003), while keeping UMAP as the dimensionality reduction method. As shown in Table 12, DBSCAN consistently outperforms the other methods across all evaluation metrics. Results indicate that DBSCAN is more effective at identifying high-density regions and filtering out noise during clustering, thereby improving the selection quality of representative speech samples and enhancing the effectiveness.

Table 12: Ablation study of clustering methods. We show WA (%) results on MELD dataset.

Clustering Method	Coreset Ratio		
	0.049%	0.243%	0.485%
KNN	29.50±4.16	49.39±9.75	44.34±4.63
K-Means	45.74±13.45	40.93±1.72	53.20±1.74
DBSCAN	**57.55±0.42**	**58.10±0.25**	**58.89±0.25**

Sensitivity to Number of Segments and Length. To evaluate the robustness of our pruning method, we perform ablation studies on two critical factors in the splitting process: the number of segments and the segment length ratio. Specifically, we vary the number of segments $n_s \in \{4, 5, 6\}$ and the segment length ratio from $\frac{1}{3}$, $\frac{1}{4}$, to $\frac{1}{5}$. Table 13 shows the results under different segment length ratios, while Table 14 presents the results across varying segment numbers. These variations have minimal impact on the quality of the selected samples, demonstrating the strong robustness of our method to splitting configurations. Unless otherwise specified, we use $n_s = 5$ and a segment length ratio of $\frac{1}{4}$ as default settings throughout all experiments.

Table 13: Ablation study of segment lengths. We show WA (%) results on MELD dataset.

Segment Length	Coreset Ratio			
	0.049%	0.243%	0.485%	2.426%
1/3	57.09±0.21	**58.49±0.48**	58.81±0.38	60.15±0.25
1/4	**57.55±0.42**	58.10±0.25	58.89±0.25	60.14±0.19
1/5	57.50±0.19	58.43±0.33	**59.03±0.57**	**60.30±0.25**

Table 14: Ablation study of different utterance split settings. We show results of WA (%) on MELD dataset.

Segment Numbers	Coreset Ratio			
	0.049%	0.243%	0.485%	2.426%
4	57.06±0.21	58.42±0.80	**59.73±0.34**	**60.56±0.27**
5	**57.55±0.42**	58.10±0.25	58.89±0.25	60.14±0.19
6	58.04±0.26	**59.35±0.18**	59.69±0.47	60.37±0.52

E Formal Justification for Feature Selection in Coarse Filtering

In this section, we provide a more formal argument to justify our choice of MFCCs over deep semantic features for the utterance-level diversity sampling in Step 1.

E.1 Objective of Diversity Sampling

The goal of our coarse filtering is to select a subset (coreset) $\mathcal{D}_c \subset \mathcal{D}$ that is most representative of the full dataset's *acoustic diversity*. Let $\mathcal{A}$ be the underlying manifold of acoustic properties. An ideal coreset would effectively cover this manifold. Many diversity sampling methods, including our approach which samples from clusters, implicitly aim to solve an objective like the *k-Center* problem:

$$\min_{\mathcal{D}_c \subset \mathcal{D}, |\mathcal{D}_c| = m} \max_{x \in \mathcal{D}} \min_{x_c \in \mathcal{D}_c} d(x, x_c) \tag{17}$$

where $d(x, x_c)$ is a distance metric. Critically, for our goal, the ideal distance metric $d(\cdot, \cdot)$ should reflect the **acoustic distance** between two samples, we denote it as $d_{\text{acoustic}}(a, a_c)$.

E.2 The Role of Feature Space as a Proxy

In practice, we cannot compute d_{acoustic} directly. Instead, we compute Euclidean distances in a feature space defined by an extractor $f(\cdot)$. The objective thus becomes minimizing this distance in the proxy space:

$$d(x, x_c) = \|f(x) - f(x_c)\|_2 \tag{18}$$

The quality of the resulting coreset is therefore entirely dependent on how well the feature space $f(\mathcal{D})$ preserves the geometric structure of the true acoustic manifold $\mathcal{A}$.

E.3 Contrasting MFCCs and Deep Features

We analyze the two feature extractors in this context.

MFCCs as a High-Fidelity Proxy (f_{MFCC}). MFCCs are designed to capture the spectral envelope of a signal. Therefore, we can posit that the distance in the MFCC space is a reasonably faithful proxy for acoustic distance. For two samples x_i, x_j with acoustic properties a_i, a_j:

$$\|f_{\text{MFCC}}(x_i) - f_{\text{MFCC}}(x_j)\|_2 \propto d_{\text{acoustic}}(a_i, a_j) \tag{19}$$

This means that samples which are acoustically far apart will also be far apart in the MFCC feature space.

Deep Features and Representation Collapse (f_{Whisper}). Deep features are trained for semantic invariance. This means for two samples x_i, x_j with different acoustic properties ($a_i \neq a_j$) but similar semantic content ($s_i \approx s_j$), the model is explicitly trained to minimize the feature distance. Specifically, if $s_i \approx s_j$, then the feature distance approaches zero:

$$\|f_{\text{Whisper}}(x_i) - f_{\text{Whisper}}(x_j)\|_2 \to 0, \qquad (20)$$

and this holds **regardless of** the acoustic distance $d_{\text{acoustic}}(a_i, a_j)$. This is the formal definition of **representation collapse** in the context of diversity sampling. The feature space systematically fails to distinguish between acoustically different samples.

E.4 Conclusion

When applying a diversity sampling objective (Eq. 1) in the Whisper feature space, the algorithm is misled. It cannot distinguish between an acoustically rich set of samples (e.g., "hello" spoken by a man, a woman, and a child) and a redundant set, because their feature representations are collapsed into a small region. The resulting coreset will therefore poorly represent the true acoustic diversity.

Conversely, because the MFCC space preserves acoustic distances, solving the k-Center objective in this space serves as a much better approximation of the ideal goal. This formally justifies why MFCCs, despite being a "classic" feature, are theoretically superior for the specific task of coarse-grained, diversity-focused data pruning.

F Hyperparameter Sensitivity Analysis

To assess the robustness of our framework to the choice of hyperparameters in the coarse filtering stage, we conducted a sensitivity analysis for the key parameters of UMAP (`n_neighbors`, `min_dist`) and DBSCAN (`eps`, `min_samples`). Figure 3 illustrates the performance (as measured by Weighted Accuracy on the MELD dataset) as each parameter is varied while the others are held at their optimal values.

The results indicate that our method's performance remains stable across a sensible range for all tested hyperparameters. For example, in Figure 3(a), while `n_neighbors=15` yields the best results for the largest coreset ratio, varying this parameter between 10 and 20 does not lead to a drastic performance collapse. This demonstrates that our framework is not overly sensitive to the precise hyperparameter settings, which enhances its practical applicability in real-world scenarios where extensive tuning may not be feasible. Our final chosen values are highlighted with a dashed line in each sub-figure.

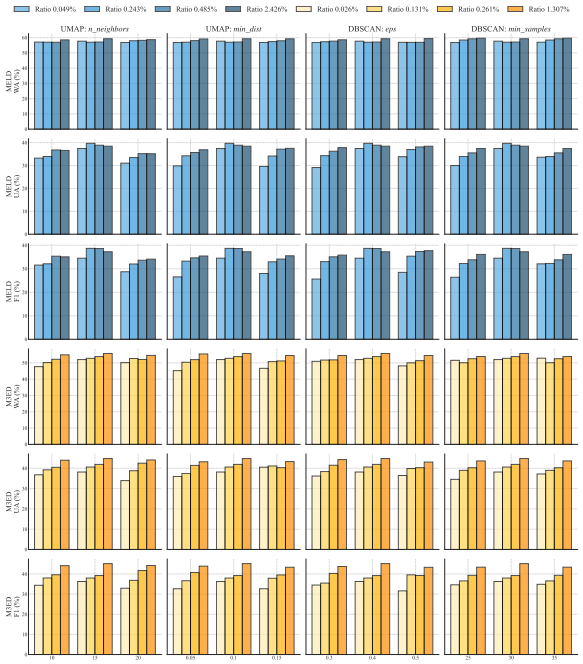

Figure 3: Hyperparameter sensitivity analysis on the MELD dataset (WA %). For each sub-figure, one hyperparameter is varied while the others are fixed to their optimal values. The red dashed line indicates the default value used in our main experiments. The results show that our method is generally robust to the choice of these hyperparameters.