SED: A New Method for Discrete Token-based ASR via Structural Entropy

Anonymous ACL submission

Abstract

Building speech processing models with Large Language Models (LLMs) has become a new effective paradigm. A key challenge in this approach is representing speech features that align well with LLMs. While continuous speech features from self-supervised learning (SSL) models capture rich information, they pose alignment challenges and lead to high computational costs. Discrete tokenization using K-means improves efficiency but suffers from fixed cluster constraints and limited adaptability to diverse speech signals. In this paper, we propose SED, a novel **S**tructural **E**ntropy-based Speech **D**iscretization method that models speech features as graph nodes and performs adaptive clustering by minimizing 2D Structural Entropy. SED automatically determines the optimal number of clusters and captures robust acoustic correlations to improve cluster quality. Experimental results demonstrate that SED achieves lower word error rates (WER) and higher clustering purity than K-means, highlighting its effectiveness for discrete token-based ASR.

1 Introduction

With the rapid development of Large Language Models (LLMs), significant revolution has been made in various natural language processing (NLP) (Peng et al., 2023; Pu et al., 2023; Ravaut et al., 2023; Lu et al., 2023) and computer vision (CV) (Driess et al., 2023; Liu et al., 2023; Ye et al., 2024) tasks. Simultaneously, the field of speech processing has seen remarkable developments, especially with the emergence of Speech Language Models (SpeechLMs) such as SpeechGPT (Zhang et al., 2023), Salmonn (Tang et al., 2024) and Qwen-Audio (Chu et al., 2023). These models have demonstrated impressive speech recognition, synthesis, translation and understanding capabilities, driving a shift toward more integrated, efficient and multi-modal AI systems.

Building on the foundational architecture and powerful capabilities of LLMs, adapting them for speech processing tasks is a natural progression. This adaption allows us to take advantage of both the rich contextual understanding of language and the nuanced features of speech, enabling more accurate and robust multimodal applications. Such advancements have led researchers to explore improved representations of speech as a sequence for LLMs. Broadly speaking, methods for representing speech inputs can be categorized into continuous features and discrete tokens. Continuous speech features are commonly extracted using self-supervised learning (SSL) models such as HuBERT (Hono et al., 2024), WavLM (Das et al., 2024), and the encoder of Whisper (Shu et al., 2023). Raw waveforms are converted into high-dimensional embeddings and fed into large language models (LLMs) through adapters. In this paradigm, the key challenge lies in effectively bridging the representation gap between continuous speech features and the embedding space of LLMs. To address this issue, (Yu et al., 2024a) and SALMONN (Tang et al., 2024) proposed using a query transformer (Q-Former) (Li et al., 2023) to convert whisper-extracted speech features into fixed-length representations suitable for models such as LLaMA (Touvron et al., 2023) and Vicuna (Chiang et al., 2023). Furthermore, (Dong et al., 2024) introduces a word boundary-sensitive compression method combined with the optimal transport algorithm to improve the alignment between speech characteristics and LLM text embeddings. Despite the effectiveness of these methods, the high dimensionality and length of the continuous speech features increase computational costs and memory demands.

Alternatively, recent studies (Yang et al., 2024a; Wang et al., 2024; Mousavi et al., 2024; Chang et al., 2024) have explored discrete speech units derived from SSL representations. These approaches typically employ K-means clustering to convert continuous speech features into discrete tokens. Models such as AudioPalm (Rubenstein et al., 2023) and SpeechGPT (Zhang et al., 2023) leverage these discretized speech tokens for SpeechLMs. Discrete speech tokens not only preserve the semantic content and temporal structure of speech but also align with the next-token prediction mechanisms of large language models (LLMs), thereby eliminating the need for additional adapters and facilitating unified speech-text modeling. However, K-means-based discretization relies heavily on predefined cluster centroids and a fixed number of clusters, which may limit its adaptability to diverse speech signals. This could result in suboptimal clustering performance and unstable outcomes.

In this paper, we address the aforementioned challenges from an information-theoretic perspective. Drawing inspiration from graph-based clustering methods widely used in NLP tasks such as social event and community detection (Ren et al., 2022; Yang et al., 2024b; Yu et al., 2024b), we explore the potential application to speech processing and propose **SED**, a novel **S**tructural **E**ntropy (SE)-based Speech **D**iscretization method for discrete token-based ASR. Specifically, we model speech features extracted from SSL models as nodes in a graph, where edges represent similarity between speech features. Clustering is performed by minimizing 2D SE, which iteratively and incrementally partitions the graph while preserving structural coherence and minimizing information loss. This process adaptively determines the optimal number of clusters. It captures robust correlations among speech units, ensuring that similar acoustic patterns are grouped more compactly, ultimately enhancing the performance of discrete token-based ASR. Our contributions are summarized as follows.

- We propose a new speech discretization method based on 2D Structural Entropy minimization. Unlike K-means, this approach automatically determines the number of clusters, offering a more adaptive and precise alignment with acoustic units by effectively capturing correlations among speech features.

- To mitigate the high computational cost of graph clustering for large-scale speech representations, we utilize an incremental structural entropy-based graph partitioning method, significantly improving clustering efficiency.

- By integrating adaptive similarity regularization, our SED method further improves clustering robustness and generalization, achieving superior performance over discrete token-based ASR baselines.

2 Preliminary

Structural Entropy (SE) (Li and Pan, 2016) is defined as the minimum number of bits required to encode a vertex that can be reached in a single step of a random walk on a graph G. Quantifies the complexity of the intrinsic structure of the graph and is closely associated with its encoding tree $\mathcal{T}$. In the following, we provide the definitions of the encoding tree and structural entropy as presented in (Li and Pan, 2016).

Given an undirected and weighted graph $G = (V, E)$ with n vertices and weights W, where V is the vertex set and E is the edge set, we have the following.

Definition 1) An encoding tree $\mathcal{T}$ of graph G is a hierarchical clustering partition of G, which includes all nodes of G as leaf nodes. This encoding tree represents a graph partition, making it applicable to partition-based clustering. The root node λ of $\mathcal{T}$ corresponds to the whole sets of the graph, i.e. $\mathcal{T}_\lambda = V$. Each tree node $\alpha \subseteq \mathcal{T}$ corresponds to a partitioning of the graph, i.e. $\mathcal{T}_\alpha \subseteq V$. For any tree node α, its leaf nodes $\{\gamma_1, ..., \gamma_n\}$ form a partition of $\mathcal{T}_\alpha$.

Definition 2) The height of each node α in $\mathcal{T}$ is denoted as $h(\alpha)$. By definition, the leaf node γ has a height of zero, i.e., $h(\gamma) = 0$. For any other node α, its height is given by $h(\alpha) = h(\alpha^-) + 1$, where α^- represents its parent node. The height of the encoding tree $\mathcal{T}$ is defined as $h(\mathcal{T}) = \max_{\alpha \in \mathcal{T}}\{h(\alpha)\}$.

Definition 3) The structural entropy of a graph G with encoding tree $\mathcal{T}$ is defined as:

$$
\begin{aligned}
H^{\mathcal{T}}(G) &= \sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} H^{\mathcal{T}}(G; \alpha) \\
&= \sum_{\alpha \in \mathcal{T}, \alpha \neq \lambda} -\frac{g_\alpha}{vol(\lambda)} \log_2 \frac{vol(\alpha)}{vol(\alpha^-)}
\end{aligned} \tag{1}
$$

where g_α represents the sum of the degrees of cut edges in $\mathcal{T}_\alpha$, where cut edges are those in E that have exactly one endpoint within $\mathcal{T}_\alpha$. The terms $vol(G)$, $vol(\alpha)$, and $vol(\alpha^-)$ denote the total sum of vertex degrees in G, $\mathcal{T}_\alpha$, and its parent node $\mathcal{T}_{\alpha^-}$, respectively.

Definition 4) The 2-Dimension (2D) SE is defined using an encoding tree with a height of 2. A

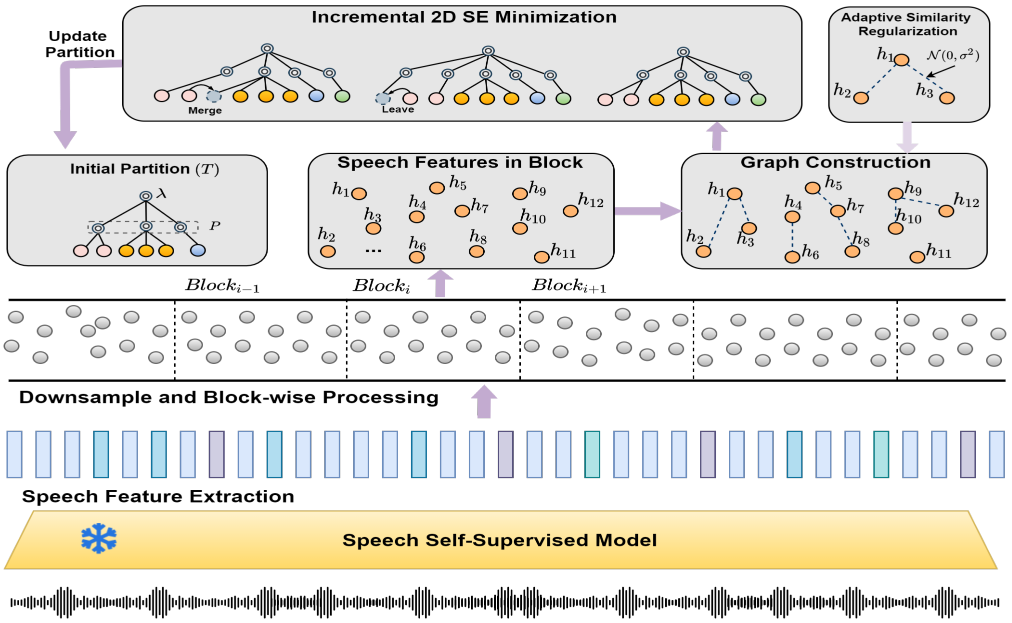

Figure 1: The framework and workflow of the proposed SED method.

2D encoding tree $\mathcal{T}$ corresponds to a graph partitioning $P = \{p_1, p_2, \ldots, p_L\}$ over V, where each p_j denotes a partition of the graph. The 2D SE is formally defined as:

$$H^{(2)}(G) = -\sum_{p_i \in P} \sum_{v_j \in p_i} \frac{g_j}{vol(G)} \log_2 \frac{d_j}{vol(p_i)} - \sum_{p_i \in P} \frac{g_{p_i}}{vol(G)} \log_2 \frac{vol(pi)}{vol(G)}, \tag{2}$$

where d_j represents the degree of vertex v_j, while g_j denotes the total weight of edges linking v_j to other vertices. The terms $vol(p_i)$ and $vol(G)$ correspond to the volumes, which are defined as the sum of degrees of the vertex within the partition p_i and throughout the graph G, respectively. Furthermore, g_{p_i} quantifies the total weight of the edges connecting vertices inside p_i to those outside it.

3 Methodology

The entire framework of the proposed SED method is illustrated in Figure 1.

3.1 Problem Formalization

Given a series of waveform data, the high-dimensional feature matrix $H = \{h_1, h_2, \ldots, h_T\} \in R^{T \times D}$ is extracted using a speech SSL model (e.g., HuBERT (Hsu et al., 2021) or WavLM (Chen et al., 2022)), where T represents the length of the feature sequence, and D denotes the dimensionality of the speech features.

By treating each h_i as a node, we construct a speech feature graph $G = (V, E, W)$, where V is the set of vertices corresponding to speech features H, E represents the edges connecting the vertices, and W denotes the edge weights, which measure the similarities of cosine between the vertices. Minimizing the structural entropy of the graph G results in partitioning the nodes into unsupervised clusters. Each speech feature is assigned to a cluster, which discretizes the speech data into a token sequence $Z = \{z_1, z_2, \ldots, z_T\}$. These tokens can be processed as text symbols, allowing their direct input into LLMs.

3.2 Graph Construction

A well-structured graph serves as the foundation for effective graph partitioning. Unlike traditional clustering methods that rely on predefined assumptions about the number of clusters (e.g., K-means), a graph-based approach enables us to model the intrinsic relationships between speech features more flexibly and adaptively. Using graph partitioning, we aim to uncover the inherent structure of speech and effectively capture dependencies within the feature space.

Given an SSL-extracted speech feature sequence

$H = \{h_1, h_2, \ldots, h_T\}$, we construct a speech feature graph $G = (V, E, W)$, where V represents the set of feature nodes and E denotes the edges that capture the relationships between these nodes. This graph-based formulation explicitly models dependencies among frames, providing a more structured representation of speech dynamics. We establish edges based on the similarity between speech features to define the graph topology. The weight of each edge, represented by the weighted adjacency matrix W, is calculated using cosine similarity: $w(i,j) = CosSim(h_i, h_j)$, where h_i and h_j are vectors corresponding to nodes i and j. This weighted graph ensures that strongly correlated speech features remain closely connected.

3.3 Speech Discretization via Incremental 2D-SE Minimization

Minimizing structural entropy (SE) effectively reveals reliable node correlations in noisy raw graphs and has been applied in various fields. Although 2D SE minimization is unsupervised and effective, it becomes computationally prohibitive for large-scale and complex graphs. Traditional bottom-up greedy merging method (Li and Pan, 2016) is costly, making them impractical for large and densely connected graphs. Hierarchical 2D SE minimization (Cao et al., 2024) improves efficiency to some extent, but the dense interconnections between nodes make graph partitioning challenging, potentially leading to information loss.

Optimization efficiency becomes critical in speech processing, where many speech frames must be clustered. To address this, we build upon the incremental 2D-SE minimization approach proposed by (Xian et al., 2025) and treat the clustering process as an incremental and dynamic procedure. Specifically, we introduce two key strategies to enhance efficiency while preserving essential structural information: 1) **Downsampling**: a sampling factor s is defined to downsample the feature sequence, reducing computational complexity while retaining critical structural correlations. 2) **Block-wise Processing**: the downsampled speech feature sequence is then divided into N equal-length blocks: $\{B_1, \ldots, B_N\}$, where $N = \lfloor T/L \rfloor$, and L is the block length. Graph construction and 2D-SE minimization are performed block by block, ensuring incremental optimization while maintaining computational feasibility.

Initially, 2D-SE minimization is applied to the first block B_1 and the resulting clusters are retained. As new blocks arrive, the graph and cluster assignments are dynamically updated. This update process leads to one of three possible outcomes for each node: 1) **remaining in its current cluster**, 2) **leaving to form a new cluster**, or 3) **merging into an existing cluster**. Given a graph and its corresponding partition is $P = \{p_1, p_2, \ldots, p_L\}$, first, if a node x remains in its current cluster, the set of partition P remains unchanged. As a result, there is no variation in the graph's 2D structural entropy, which can be expressed as: $\Delta_{Keep} = 0$. Second, if node x leaves its current cluster p_i and forms a new cluster, the partition set is updated to: $P' = \{p_1, \ldots, p'_i, \ldots, p_L, x\}$, where $p'_i = p_i \setminus \{x\}$. Consequently, the change in 2D structural entropy is given by:

$$\begin{aligned}
\Delta_{Leave} &= H^{\mathcal{T}'}(G) - H^{\mathcal{T}}(G) \\
&= \sum_{p \in P'} H^{(2)}(p'_n) - \sum_{p \in P} H^{(2)}(p_n) \\
&= H^{(2)}(p'_i) + H^{(2)}(x) - H^{(2)}(p_i) \\
&= \frac{g_{p_i}}{vol(G)} \log \frac{vol(p_i)}{vol(G)} - \frac{g_{p'_i}}{vol(G)} \log \frac{vol(p'_i)}{vol(G)} \\
&+ \frac{vol(p'_i)}{vol(G)} \log \frac{vol(p'_i)}{vol(p_i)} + \frac{d_x}{vol(G)} \log \frac{vol(p_i)}{vol(G)},
\end{aligned} \tag{3}$$

where Δ_{Leave} denotes the variation in 2D SE when a node x exits cluster p_i to establish a new cluster. The encoding tree associated with the updated partition set P' is represented as $\mathcal{T}'$. The 2D SE values of the graph under the partition sets P and P' are given by $H^{\mathcal{T}}(G)$ and $H^{\mathcal{T}'}(G)$, respectively. The term $H^{(2)}(p_i)$ indicates the SE of cluster p_i. The total volume of the graph, as well as the volumes of cluster p_i and its newly formed counterpart p'_i, are denoted as vol(G), vol(p_i), and vol(p'_i), respectively. Additionally, g_{p_i} and $g_{p'_i}$ represent the cuts associated with p_i and p'_i, respectively. Third, the process of merging node x from cluster p_i into cluster p_j can be decomposed into two sequential steps: a) node x departs from p_i and forms a new cluster. b) Then, it transitions from this newly formed cluster to p_j. Notably, the change in the graph's 2D SE resulting from leaving the new cluster and joining p_j is exactly the inverse of the change caused by leaving p'_j (where $p'_j = p_j \cup x$) and forming a new cluster. Thus, the node merging strategy can be formalized as follows:

$$\Delta_{Merge} = \Delta_{Leave}(x, p_i) - \Delta_{Leave}(x, p'_j), \tag{4}$$

where Δ_{Merge} represents the variation in the graph's 2D SE when node x moves from cluster p_i to p_j. The terms $\Delta_{Leave}(x, p_i)$ and $\Delta_{Leave}(x, p'_j)$ correspond to the changes in 2D SE when node x exits clusters p_i and p'_j, respectively, to form a new cluster.

Based on the above analysis, the optimal strategy for updating the partition tree to minimize the 2D SE for a newly arrived node x is determined as:

$$\min\{\Delta_{Keep}, \Delta_{Leave}, \Delta_{Merge}\}. \tag{5}$$

Finally, after determining the partition P, each speech feature in H is assigned a cluster label based on the maximum cosine similarity. The workflow of the proposed method is described in Algorithm 1. The time complexity is analyzed in Appendix A.

Algorithm 1 Incremental 2D-SE Minimization for Speech Feature Clustering.

Input: Speech features: $H = \{h_1, h_2, \dots, h_T\}$
Output: Cluster labels: $Z = \{z_1, z_2, \dots, z_T\}$
Initialization: Define block size L,
sampling factor s, similarity threshold θ,
Initialize partition $P = \emptyset$
for $n = 1$ ***to*** $N = \lfloor T/L \rfloor$ **do**
  Extract block $B_n = \{h_{(n-1)L+1}, \dots, h_{nL}\}$
  Down-sample B_n with factor s to obtain B'_n;
  Construct graph $G_n = (V_n, E_n, W_n)$ from B'_n, keep edges that weight greater than θ;
  if $n == 1$ **then**
    Compute $H^{(2)}(G_1)$,
    obtain initial partition P
  else
    repeat
      foreach *node* $x \in B'_n$ **do**
        Assign x to
        $\arg\min\{\Delta_{Keep}, \Delta_{Leave}, \Delta_{Merge}\}$
      end
    until $|\Delta_{SE}| < \epsilon$;
    Update partition P accordingly
  end
end
Finally: Dump cluster labels
foreach *speech feature* $h \in H$ **do**
  $z = \arg\max_{p \in P} \text{CosSim}(h, p)$
end
return final cluster labels Z

3.4 Adaptive Similarity Regularizatio

We introduce an adaptive similarity regularization strategy that injects Gaussian noise into the similarity calculation to improve clustering robustness and reduce sensitivity to spurious correlations. This strategy aims to simulate real-world speech variability caused by the environment or speakers. We modify the cosine similarity by adding Gaussian noise:

$$\hat{w}(i, j) = CosSim\{h_i, h_j\} + \mathcal{N}(0, \sigma^2) \tag{6}$$

where $\mathcal{N}(0, \sigma^2)$ represents zero-mean Gaussian noise with variance σ. This perturbation encourages the clustering to be more robust.

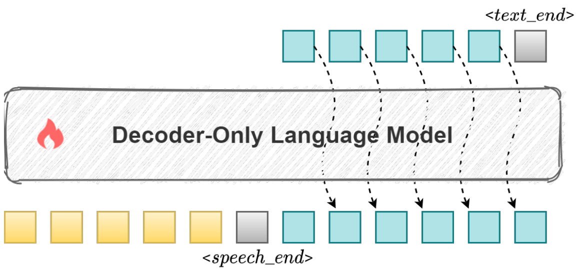

Figure 2: The framework of the discrete token-based, decoder-only language model for ASR. Model is trained to predict the next token using cross-entropy loss.

3.5 Speech Discrete Token based ASR Model

In line with discrete token-based ASR models, we build an ASR model with a decoder-only language model, incorporating discrete speech tokens obtained through 2D SE minimization, as illustrated in Figure 2. Given the speech token sequence $Z = \{z_1, z_2, \dots, z_T\}$ and its corresponding transcription $Y = \{y_1, y_2, \dots, y_m\}$, the language model is trained to generate the text sequence Y based on the discrete speech tokens Z. To integrate both speech and text within the model, the original embedding matrix E of the language model, which has dimensions $t \times d$ (where t represents the number of text tokens and d is the embedding size), is expanded to $(t + s) \times d$ to accommodate an additional set of s speech tokens. The cross-entropy (CE) is utilized for training:

$$\mathcal{L}_{CE} = -\sum \log p(y_t | Z, y_{<t}; \phi), \tag{7}$$

where y_t is the text token at time step t and $y < t$ is the text tokens earlier than time step t, Z is the speech discrete token, and ϕ are trainable parameters.

4 Experiments Setup

Dataset: Consistent with widely used benchmarks for discrete token-based ASR models (Chen et al.,

Architecture	Models	dev-clean	dev-other	test-clean	test-other
Encoder-Decoder	Conformer	3.10	8.91	3.29	8.81
	Whisper Large-v2	2.22	6.07	2.37	6.08
Decoder-Only	HuBERT-Large + GPT2	3.05	6.63	3.11	7.12
Discretized via K-means	WavLM-Large + GPT2	3.41	7.26	3.59	7.21
	HuBERT-Large + QWen2-0.5B	5.02	9.1	5.56	9.39
	WavLM-Large + QWen2-0.5B	4.65	8.51	5.01	8.58
Decoder-Only,	HuBERT-Large + GPT2	**2.83**	**5.71**	**2.94**	**6.02**
Discretized via SE (ours)	WavLM-Large + GPT2	3.10	6.52	3.21	6.58
	HuBERT-Large + QWen2-0.5B	3.77	6.79	3.70	7.33
	WavLM-Large + QWen2-0.5B	3.71	7.36	4.09	7.26
Discretized via SE (ours),	HuBERT-Large + GPT2	**2.68**	**5.45**	**2.71**	**5.89**
+ Adaptive Regularization	HuBERT-Large + QWen2-0.5B	3.60	6.32	3.61	7.06

Table 1: WER on the LibriSpeech dev and test sets for ASR models with different architectures. Results are reported on dev-clean, dev-other, test-clean, and test-other sets. Lower WER indicates better performance.

2024; Wang et al., 2024), we evaluate the effectiveness of the proposed SED method on the LibriSpeech corpus (Panayotov et al., 2015), which consists of a 960-hour training set. Performance is evaluated regarding word error rates (WER) across the dev-clean, dev-other, test-clean, and test-other sets. Evaluation is also conducted on the GigaSpeech (Guoguo Chen, 2021) M-size datasets.
Speech Token Discretization: For speech feature extraction, we use HuBERT-large[1] and WavLM-Large[2] pre-trained models, both composed of convolutional layers and transformer encoder layers with a hidden size of 1024. To reduce the computational cost, the downsampling factor s is set to 0.001, resulting in approximately 177K randomly sampled speech frames for clustering. These frames are grouped into blocks of length 1000. For graph construction, we evaluate performance across different cosine similarity thresholds θ. Following the baseline configuration, we employ SentencePiece[3] to tokenize speech tokens, yielding 6000 subword units.
Decoder-only LM: Due to limited GPU resources, we employ GPT2-medium[4] (350M parameters) and Qwen2-0.5B[5] as language models for decoder only for discrete token-based ASR. GPT2-medium consists of a 24 layers transformer, a hidden size of 1024, and a vocabulary of 50,257 text tokens, while Qwen2-0.5B has a 24 layers transformer, a hidden size of 896, and a vocabulary of 151,643 text tokens. We expand the vocabulary with 6000 speech subword units to accommodate speech tokens. Additionally, we introduce two special end tokens, <speech_end> and <text_end> for GPT2-medium, while reuses <|endoftext|> and <|im_end|> for Qwen2-0.5B as delimiters.

The models are trained using the Adam optimizer, which has a learning rate 3e-4 for 10 epochs on 8 A40 GPUs. Additionally, time masking is applied to all input tokens, including speech and text tokens, by replacing each token with a special padding token with a probability of 0.3.

5 Results

5.1 Main Results

Table 1 presents the WER results in the LibriSpeech dataset. The Whisper Large-v2 model performs best, with WERs of 2.22% on dev-clean and 6.07% on dev-other. However, this can be attributed to its large model size (1.55B parameters) and extensive weakly labeled training data (680,000 hours). The Conformer model (consists of 12-layers Conformer encoder and 6-layers Transformer decoder), achieving a WER of 3.10% on dev-clean.

For discrete token-based ASR models, HuBERT-Large + GPT2 trained on K-means clustered tokens achieves a WER of 3.05% on dev-clean and 6.63% on dev-other. WavLM-Large + GPT2 shows slightly higher WERs while using Qwen2-0.5B, as the language model results in a performance drop, likely due to architectural and linguistic differences. The proposed SE method significantly improves WER compared to K-means. Specifi-

[1] https://dl.fbaipublicfiles.com/hubert/hubert_large_ll60k.pt
[2] https://github.com/microsoft/unilm/tree/master/wavlm
[3] https://github.com/google/sentencepiece
[4] https://huggingface.co/gpt2-medium
[5] https://huggingface.co/Qwen/Qwen2-0.5B

cally, HuBERT-Large + GPT2 with SE reduces WER from 3.05% to 2.83% on dev-clean and from 6.63% to 5.71% on dev-other. Similar trends are observed across the test sets, confirming that SE enhances speech token clustering quality, thereby improving ASR performance. Notably, SE substantially improves dev-other and test-other, which contain more acoustically challenging and diverse data. This shows its robustness in handling noisy and complex speech scenarios. Furthermore, incorporating adaptive regularization further refines clustering, leading to improved generalization. This enhancement achieves the best performance among all discrete token-based models, demonstrating the effectiveness of SE and adaptive regularization in handling speech variations.

Table 2 presents the WER results for the GigaSpeech M-size test set. The performance trend is consistent with the results on LibriSpeech. The SE method significantly outperforms the K-means in all evaluated models. For instance, HuBERT-Large + GPT2 reduces WER from 17.74% (K-means) to 13.35% (SE), while WavLM-Large + GPT2 improves from 15.48% to 13.89%. Similarly, the use of SE leads to notable improvements for models that incorporate Qwen2-0.5B. These results further confirm that SE provides more phonemically coherent discrete representations, which benefit downstream ASR performance.

Method	Models	WER
Discretized via K-means	HuBERT-L + GPT2	17.74
	WavLM-L + GPT2	15.48
	HuBERT-L + QWen2-0.5B	19.56
	WavLM-L + QWen2-0.5B	16.85
Discretized via SE	HuBERT-L + GPT2	**13.35**
	WavLM-L + GPT2	13.89
	HuBERT-L + QWen2-0.5B	16.27
	WavLM-L + QWen2-0.5B	14.71

Table 2: WER on the GigaSpeech M-size test set.

5.2 Discrete Token Quality

We further assess the clustering performance of the proposed SED method compared to the traditional K-means. The quality of the resulting discrete speech tokens is measured based on their correlation with phoneme boundaries and labels on the Librispeech set *dev-clean* and *dev-other*. Specifically, we employ three widely used metrics: Cluster Purity (ClsPur), Phoneme Purity (PhnPur), and Phone-Normalized Mutual Information (PNMI). ClsPur quantifies the homogeneity of phoneme classes within each cluster. A higher ClsPur indicates that clusters are more consistent in representing specific phonemes. PhnPur measures the consistency of cluster assignments for each phoneme. A higher PhnPur suggests that phonemes are predominantly aligned with specific clusters, indicating a stronger phoneme-to-cluster correspondence. Phone-Normalized Mutual Information (PNMI) evaluates the mutual dependency between discrete speech tokens and phoneme labels, normalized to account for phoneme frequency distribution. Higher PNMI values reflect a stronger alignment between the discrete token and the underlying phoneme.

As shown in Table 3, from the perspective of WER, the clustering of K-means is highly sensitive to the choice of K. In contrast, SE demonstrates greater robustness to parameter variations, with WER consistently decreasing as θ increases and maintains a relatively stable range between 4.36% and 5.04%. This indicates that SE is less sensitive to hyperparameter choices and provides more reliable performance across different settings. Regarding cluster quality, the ClsPur score for SE is 21.68%, more than three times higher than the best K-means result (7.00%). This shows that SE forms more compact and well-structured clusters. Furthermore, SE consistently achieves higher PhnPur and PNMI scores, indicating that the discrete tokens generated by SE exhibit better phonemic coherence, contributing to improved ASR performance. Furthermore, we observed that SE yields a more compact and balanced token distribution than K-means while reducing the token sequence length. See Appendix B for details.

5.3 Clustering Visualization

We conduct clustering visualization using Ground Truth labels, K-means (K=2000) clustering, and SE (θ=0.7) clustering results on the LibriSpeech dev-clean subset. High-dimensional speech features were projected onto a 2D plane through PCA for dimensionality reduction. For Ground Truth, we directly utilize the provided phoneme labels, while for K-means and SE Clustering, cluster assignments were derived from their respective algorithms. It is important to note that the number of clusters in K-means and SE clustering exceeds that of the Ground Truth, meaning that multiple clusters may correspond to a single phoneme in the Ground

Method	#Clusters	ClsPur(%) ↑	PhnPur(%) ↑	PNMI(%) ↑	AvgWER(%) ↓
K-means	$K = 1000$	7.00 / 6.46	70.95 / 67.17	73.00 / 67.76	10.89
	$K = 2000$	4.23 / 3.84	74.03 / 69.77	76.50 / 71.14	4.98
	$K = 3000$	3.20 / 2.92	75.55 / 71.25	78.25 / 72.96	9.07
SE	$\theta = 0.65, P = 1323$	21.68 / 20.63	71.18 / 73.51	67.84 / 69.79	5.04
	$\theta = 0.68, P = 2263$	18.89 / 17.53	73.58 / 75.19	71.92 / 75.86	4.85
	$\theta = 0.70, P = 3178$	16.45 / 15.72	77.32 / 74.57	75.64 / 77.60	4.36

Table 3: Clustering performance of K-means and SE in terms of clustering purity (ClsPur), phoneme purity (PhnPur), and PNMI, as well as average WER (AvgWER) on the Librispeech dev and test sets.

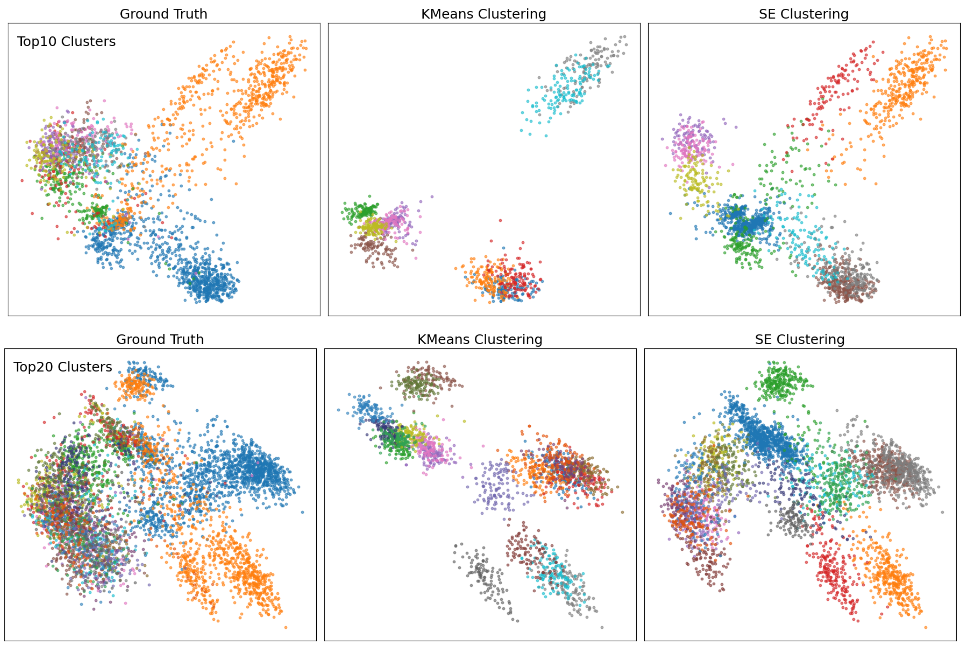

Figure 3: PCA-based 2D visualization of top-10 (upper) and top-20 (lower) clusters from Ground Truth, K-means, and SE Clustering on the LibriSpeech dev-clean subset. Each point represents a sampled speech feature, with colors indicating different clusters.

Truth. Each data point retained its original index, ensuring precise alignment with its corresponding label across different clustering methods.

The upper panel of Figure 3 illustrates the top 10 clusters for the ground truth, K-means, and SE clustering results, whereas the lower panel presents the top 20 clusters. We randomly sample 100 speech features from each cluster to ensure representative visualizations. The visualizations reveal that K-means, due to its centroid-based approach, form compact, well-defined clusters, whereas SE clustering captures more organic, flexible structures. Notably, SE Clustering outperforms K-means in preserving the intrinsic data distribution, particularly within complex clusters. As clusters increase, SE Clustering demonstrates superior adaptability, maintaining meaningful separations and reflecting the underlying data structure more effectively.

6 Conclusion

In this paper, we proposes the SED, a new discretization method for speech token-based ASR via 2D structure entropy minimization. Unlike traditional K-means clustering, this approach automatically determines the number of clusters, offering a more adaptive and precise alignment with acoustic units by effectively capturing correlations among speech features. Experimental results demonstrate that the SED consistently outperforms K-means across various ASR models, achieving notable reductions in WER. Furthermore, clustering performance metrics indicate that SED generates more phonetically consistent speech tokens while reducing the average token length, leading significant reduction in computational cost. These results validate the effectiveness of SED in improving token discretization and downstream ASR performance.

7 Limitations

Despite promising results, the proposed SED method has limitations. First, its performance depends on the quality of speech representations extracted from SSL models. Variations in pre-training data and model architectures may lead to inconsistent clustering quality, potentially affecting downstream ASR performance. Second, SED employs a random sampling strategy for feature clustering, which may limit the representativeness of the clustered speech tokens and overlook rare but important acoustic patterns in the entire dataset. Lastly, K-means and SED focus on clustering high-dimensional speech features into discrete tokens, which may inadvertently neglect the fine-grained temporal dependencies inherent in continuous speech. Future work will explore more efficient clustering algorithms and robust adaptation techniques to address these challenges and further enhance the effectiveness of SED.

References

Yuwei Cao, Hao Peng, Zhengtao Yu, and S Yu Philip. 2024. Hierarchical and incremental structural entropy minimization for unsupervised social event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI 2024)*, volume 38, pages 8255–8264.

Xuankai Chang, Brian Yan, Kwanghee Choi, Jee-Weon Jung, Yichen Lu, Soumi Maiti, Roshan Sharma, Jiatong Shi, Jinchuan Tian, Shinji Watanabe, Yuya Fujita, Takashi Maekaku, Pengcheng Guo, Yao-Fei Cheng, Pavel Denisov, Kohei Saijo, and Hsiu-Hsuan Wang. 2024. Exploring speech recognition, translation, and understanding with discrete speech units: A comparative study. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024)*, pages 11481–11485.

Qian Chen, Wen Wang, Qinglin Zhang, Siqi Zheng, Shiliang Zhang, Chong Deng, Yukun Ma, Hai Yu, Jiaqing Liu, and Chong Zhang. 2024. Loss masking is not needed in decoder-only transformer for discrete-token-based asr. In *Proceeding of International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024)*, pages 11056–11060.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. 2022. WavLM: Large-scale self-supervised pre-training for full stack speech processing. 16:1505–1518.

Wei Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. 2023. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *Audio and Speech Processing Repository*, arXiv:2311.07919.

Nilaksh Das, Saket Dingliwal, Srikanth Ronanki, Rohit Paturi, Zhaocheng Huang, Prashant Mathur, Jie Yuan, Dhanush Bekal, Xing Niu, Sai Muralidhar Jayanthi, Xilai Li, Karel Mundnich, Monica Sunkara, Sundararajan Srinivasan, Kyu J Han, and Katrin Kirchhoff. 2024. SpeechVerse: A large-scale generalizable audio language model. *Computation and Language Repository*, arXiv:2405.08295.

Ling Dong, Zhengtao Yu, Wenjun Wang, Yuxin Huang, Shengxiang Gao, and Guojiang Zhou. 2024. Integrating speech self-supervised learning models and large language models for asr. In *Proceedings of Interspeech 2024*, pages 3954–3958.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 2023. PaLM-E: an embodied multimodal language model. In *Proceedings of the International Conference on Machine Learning (ICML 2023)*, pages 8469 – 8488.

Guanbo Wang Jiayu Du Wei-Qiang Zhang Chao Weng Dan Su Daniel Povey Jan Trmal Junbo Zhang Mingjie Jin Sanjeev Khudanpur Shinji Watanabe Shuaijiang Zhao Wei Zou Xiangang Li Xuchen Yao Yongqing Wang Yujun Wang Zhao You Zhiyong Yan Guoguo Chen, Shuzhou Chai. 2021. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. In *Proceedings of Interspeech 2021*.

Yukiya Hono, Koh Mitsuda, Tianyu Zhao, Kentaro Mitsui, Toshiaki Wakatsuki, and Kei Sawada. 2024. Integrating pre-trained speech and language models for end-to-end speech recognition. In *Findings of the Association for Computational Linguistics (ACL 2024)*, pages 13289–13305.

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 transactions on audio, speech, and language processing*, 29:3451–3460.

Angsheng Li and Yicheng Pan. 2016. Structural information and dynamical complexity of networks. *IEEE*

Transactions on Information Theory, 62(6):3290–3339.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *Proceedinds of International conference on machine learning (ICML 2023)*, pages 19730–19742.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. In *Proceeds of Advances in Neural Information Processing Systems (NIPS2024)*, volume 36, pages 34892–34916.

Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2023. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. In *Proceedings of The International Conference on Learning Representations (ICLR 2023)*.

Pooneh Mousavi, Jarod Duret, Salah Zaiem, Luca Della Libera, Artem Ploujnikov, Cem Subakan, and Mirco Ravanelli. 2024. How should we extract discrete audio tokens from self-supervised models? *Sound Repositories*, arXiv:2406.10735.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, pages 5206–5210.

Keqin Peng, Liang Ding, Qihuang Zhong, Li Shen, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. 2023. Towards making the most of ChatGPT for machine translation. In *Proceedings of Findings of the Association for Computational Linguistics (EMNLP 2023)*, pages 5622–5633.

Xiao Pu, Mingqi Gao, and Xiaojun Wan. 2023. Summarization is (almost) dead. *Computation and Language Repositories*, arXiv:2309.09558.

Mathieu Ravaut, Hailin Chen, Ruochen Zhao, Chengwei Qin, Shafiq Joty, and Nancy Chen. 2023. PromptSum: Parameter-efficient controllable abstractive summarization. *Computation and Language Repositories*, arXiv:2308.03117.

Jiaqian Ren, Lei Jiang, Hao Peng, Yuwei Cao, Jia Wu, Philip S. Yu, and Lifang He. 2022. From known to unknown: Quality-aware self-improving graph neural network for open set social event detection. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM 2022)*, page 1696–1705.

Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. 2023. AudioPaLM: A large language model that can speak and listen. *Computation and Language Repositories*, arXiv:2306.12925.

Yu Shu, Siwei Dong, Guangyao Chen, Wenhao Huang, Ruihua Zhang, Daochen Shi, Qiqi Xiang, and Yemin Shi. 2023. LLaSM: Large language and speech model. *Computation and Language Repositories*, arXiv:2308.15930.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun MA, and Chao Zhang. 2024. SALMONN: Towards generic hearing abilities for large language models. In *Proceedings of the International Conference on Learning Representations (ICML2024)*.

Hugo Touvron, Louis Martin, Kevin Stone, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *Computation and Language Repositories*, arXiv:2307.09288.

Dingdong Wang, Mingyu Cui, Dongchao Yang, Xueyuan Chen, and Helen Meng. 2024. A comparative study of discrete speech tokens for semantic-related tasks with large language models. *Computation and Language Repositories*, arXiv:2411.08742.

Yantuan Xian, Pu Li, Hao Peng, Zhengtao Yu, Yan Xiang, and Philip S. Yu. 2025. Community detection in large-scale complex networks via structural entropy game. In *Proceedings of the WEB CONFERENCE 2025*.

Yifan Yang, Feiyu Shen, Chenpeng Du, Ziyang Ma, Kai Yu, Daniel Povey, and Xie Chen. 2024a. Towards universal speech discrete tokens: A case study for ASR and TTS. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2024)*, pages 10401–10405.

Zhiwei Yang, Yuecen Wei, Haoran Li, Qian Li, Lei Jiang, Li Sun, Xiaoyan Yu, Chunming Hu, and Hao Peng. 2024b. Adaptive differentially private structural entropy minimization for unsupervised social event detection. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM 2024)*, pages 2950–2960.

Qinghao Ye, Haiyang Xu, Jiabo Ye, Ming Yan, Anwen Hu, Haowei Liu, Qi Qian, Ji Zhang, and Fei Huang. 2024. mPLUG-Owl2: Revolutionizing multi-modal large language model with modality collaboration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR 2024)*, pages 13040–13051.

Wenyi Yu, Changli Tang, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. 2024a. Connecting speech encoder and large language model for ASR. In *Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2024)*, pages 12637–12641.

Xiaoyan Yu, Yifan Wei, Shuaishuai Zhou, Zhiwei Yang, Li Sun, Hao Peng, Liehuang Zhu, and Philip S. Yu. 2024b. Towards effective, efficient and unsupervised social event detection in the hyperbolic space. *CoRR*, abs/2412.10712.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023. SpeechGPT: Empowering large language models with intrinsic cross-modal conversational abilities. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*.

A Time Complexity

The main computational steps of the proposed SED method involve 1) Graph Construction: Constructing a similarity graph from speech features has a complexity of $O(V^2)$, where V is the number of nodes (speech features); 2) Incremental 2D Structural Entropy minimization: The initialization step requires $O(L)$, where L is the block size of a speech feature segment. During the incremental minimization process, for each node, determining the optimal action (staying in its current cluster, forming a new cluster, or merging into an existing one) requires $O(k)$ operations, where k is the number of neighboring nodes considered. Given I iterations, the overall complexity of this step is $O(IkV)$. Thus, the total computational complexity is $O(V^2 + IkV)$. The graph construction being the most computationally intensive step.

B Discrete Token Distribution

We analyzed the frequency distribution of discrete speech tokens obtained using two clustering methods: K-means and SE clustering. Additionally, we compared the distribution of BPE-applied discrete tokens, as shown in Figure 4. The upper subfigure illustrates the clustering and BPE results using K-means, while the lower subfigure presents the results using SE clustering. The red dashed line represents the 95% cumulative frequency threshold.

From the figures, we observed that K-means clustering results in a more imbalanced token distribution, which can lead to inefficient representation and potential noise during downstream LLM training. In contrast, SE clustering generates a more compact token distribution, utilizing the codebook space more effectively and reducing the impact of underutilized tokens. Moreover, applying BPE enhances token granularity and significantly reduces the sequence length (as shown in 4), which can improve representation efficiency and downstream performance. The average token length using SE is about 60% of that of K-means, indicating a significant reduction in computational cost.

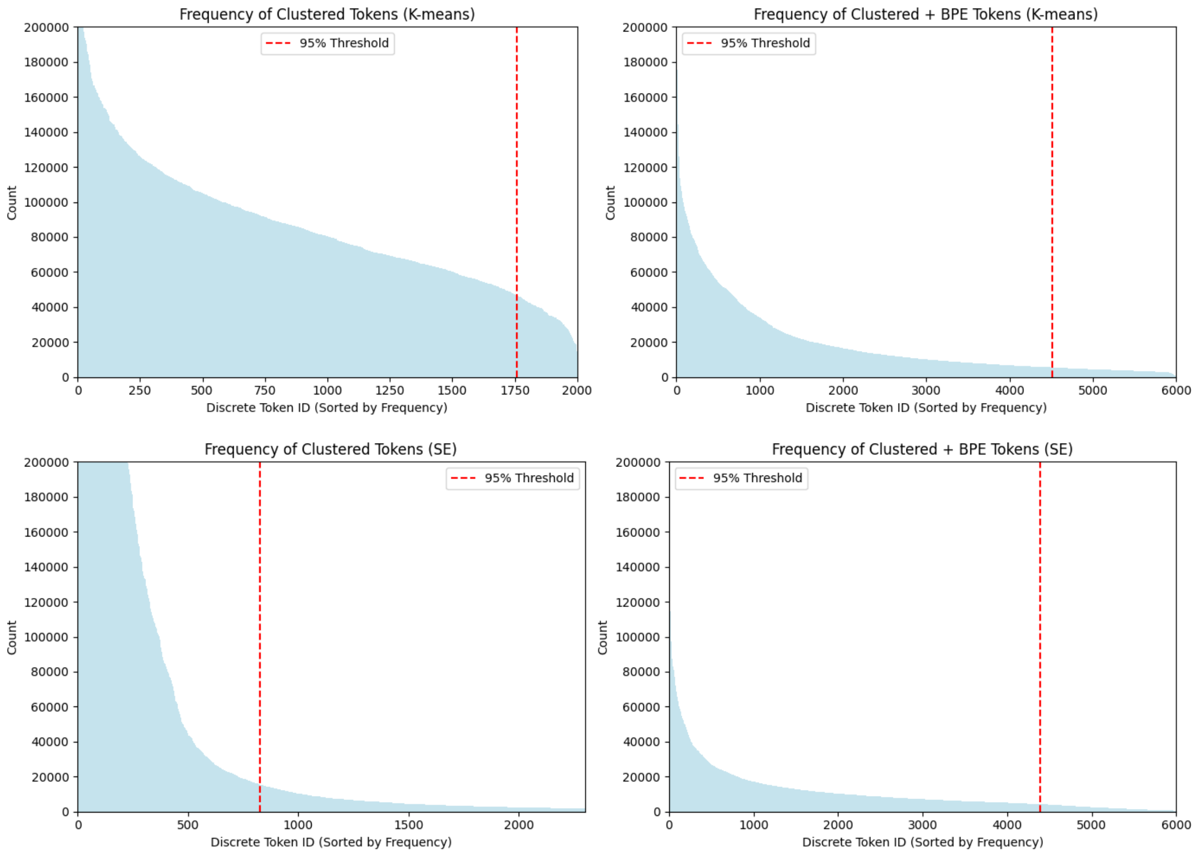

Figure 4: Frequency distribution of discrete tokens obtained via K-means $(K = 2000)$ and SE $(\theta = 0.7)$ clustering on Librispeech train set, as well as the BPE token distribution.

Method	**speech samples**	**speech frames**	**avgTokenLen (BEP applied)**
K-means	281,241	172,812,419	414
SE	281,241	172,812,419	253

Table 4: Statistics of Librispeech train-set token length obtained via K-means and SE clustering.