HuBERT: Self-Supervised Speech Representation

건국대학교
컴퓨터공학과

김동환

2024.07.15

Abstract

문제

음성을 Self-supervised 방식으로 학습할때 문제

1. 같은 입력 발화에 여러 음성 단위가 포함
2. pre-training 단계에서는 입력 소리 단위의 사전의 부재
3. 소리 단위는 명백한 부분 없이 가변 길이를 가짐

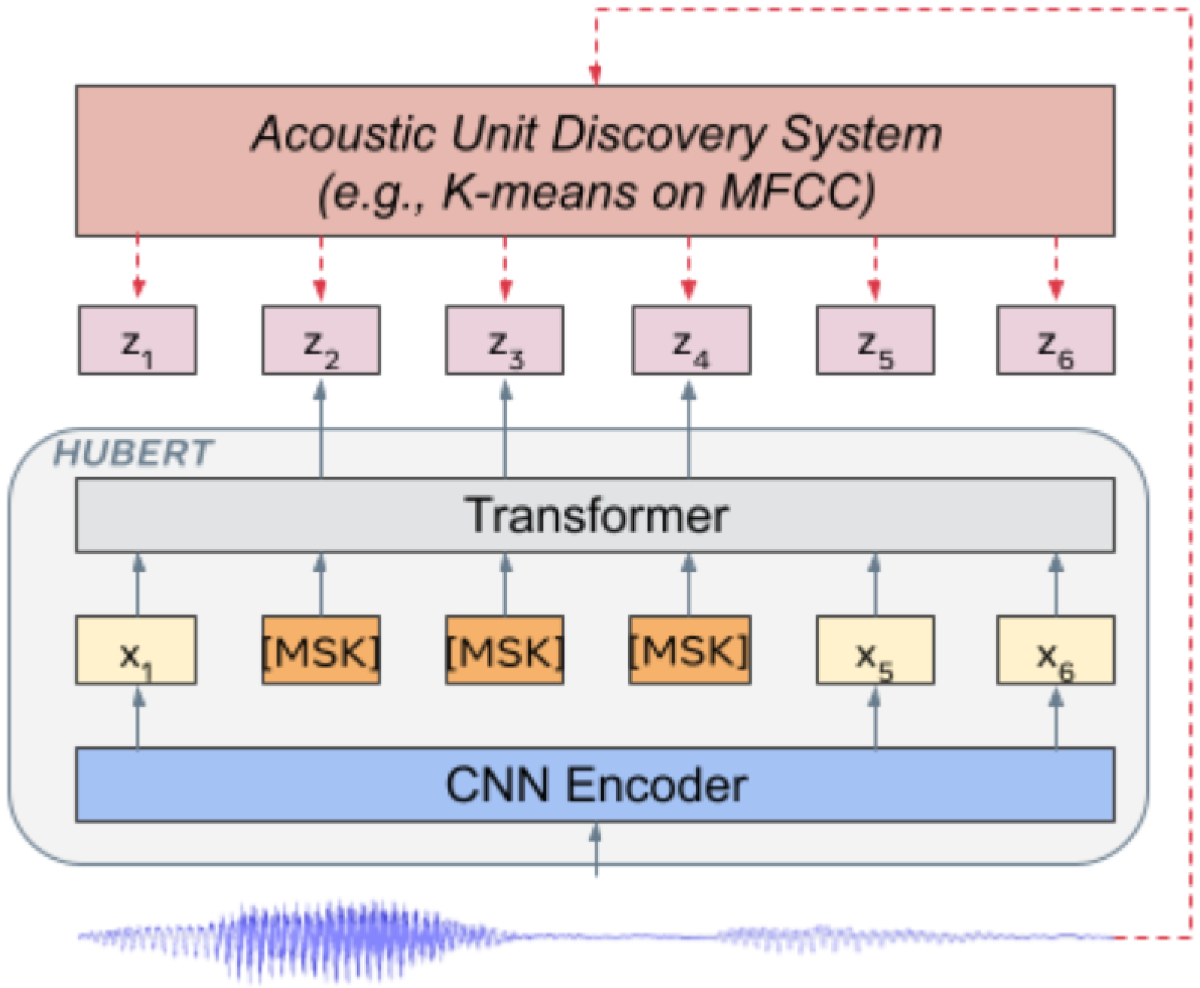
해결

Facebook AI의 Hidden-Unit BERT (HuBERT) 모델

offliine clustering를 이용하여 target labels 생성

Introduction

HuBERT



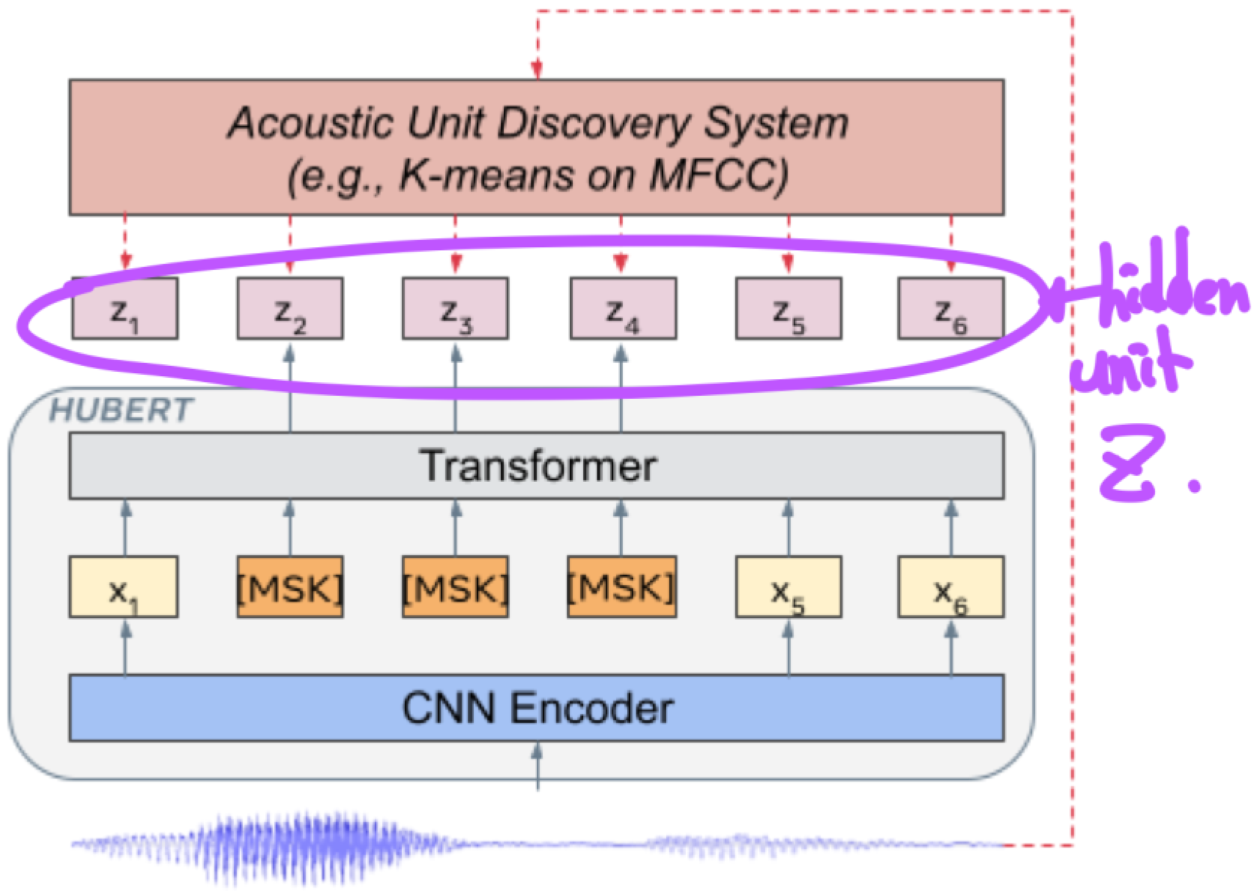
연구 개요

특정 시점의 X 입력이 들어왔을때,
Masked 영역을 알맞게 clustering 하는 것

HuBERT: Self-Supervised Speech Representation

Introduction

HuBERT



빨간 점선 : firstiteration
회색 실선 : afterfirstiteration

HuBERT의 아키텍쳐

T 시점의 발화를 X
Z=h(X), Z는 clustering 결과
h는 clustering model

$$h(X) = Z = [z_1, \cdots, z_T]$$

HuBERT: Self-Supervised Speech Representation

고려해야 할 점

어떻게 masking 할 것인가

각 시점 입력의
p%를 택해 l 만큼 마스킹

Loss를 어떻게 측정할 것인가

마스킹 처리 된 부분과
처리되지 않은 부분을 각각
CrossEntropy

$$L_m(f; X, M, Z) = \sum_{t \in M} \log p_f(z_t \mid \tilde{X}, t),$$

고려해야 할 점

$$L = \alpha L_m + (1-\alpha) L_u$$

teacher	C	PNMI	dev-other WER (%) $\alpha = 1.0$	$\alpha = 0.5$	$\alpha = 0.0$
Chenone (supervised top-line)	8976	0.809	10.38	9.16	9.79
K-means on MFCC	50	0.227	18.68	31.07	94.60
	100	0.243	17.86	29.57	96.37
	500	0.276	18.40	33.42	97.66
K-means on BASE-it1-layer6	500	0.637	11.91	13.47	23.29
K-means on BASE-it2-layer9	500	0.704	10.75	11.59	13.79

Loss를 어떻게 측정할 것인가

마스킹 처리 된 부분과
처리되지 않은 부분을 각각
CrossEntropy

$$L_m(f; X, M, Z) = \sum_{t \in M} \log p_f(z_t \mid \tilde{X}, t),$$

모델앙상블

$$L_m(f; X, \{Z^{(k)}\}_k, M) = \sum_{t \in M} \sum_k \log p_f^{(k)}(z_t^{(k)} \mid \tilde{X}, t)$$

teacher	WER
K-means {50,100}	17.81
K-means {50,100,500}	17.56
Product K-means-0-100	19.26
Product K-means-1-100	17.64
Product K-means-2-100	18.46
Product K-means-{0,1,2}-100	16.73

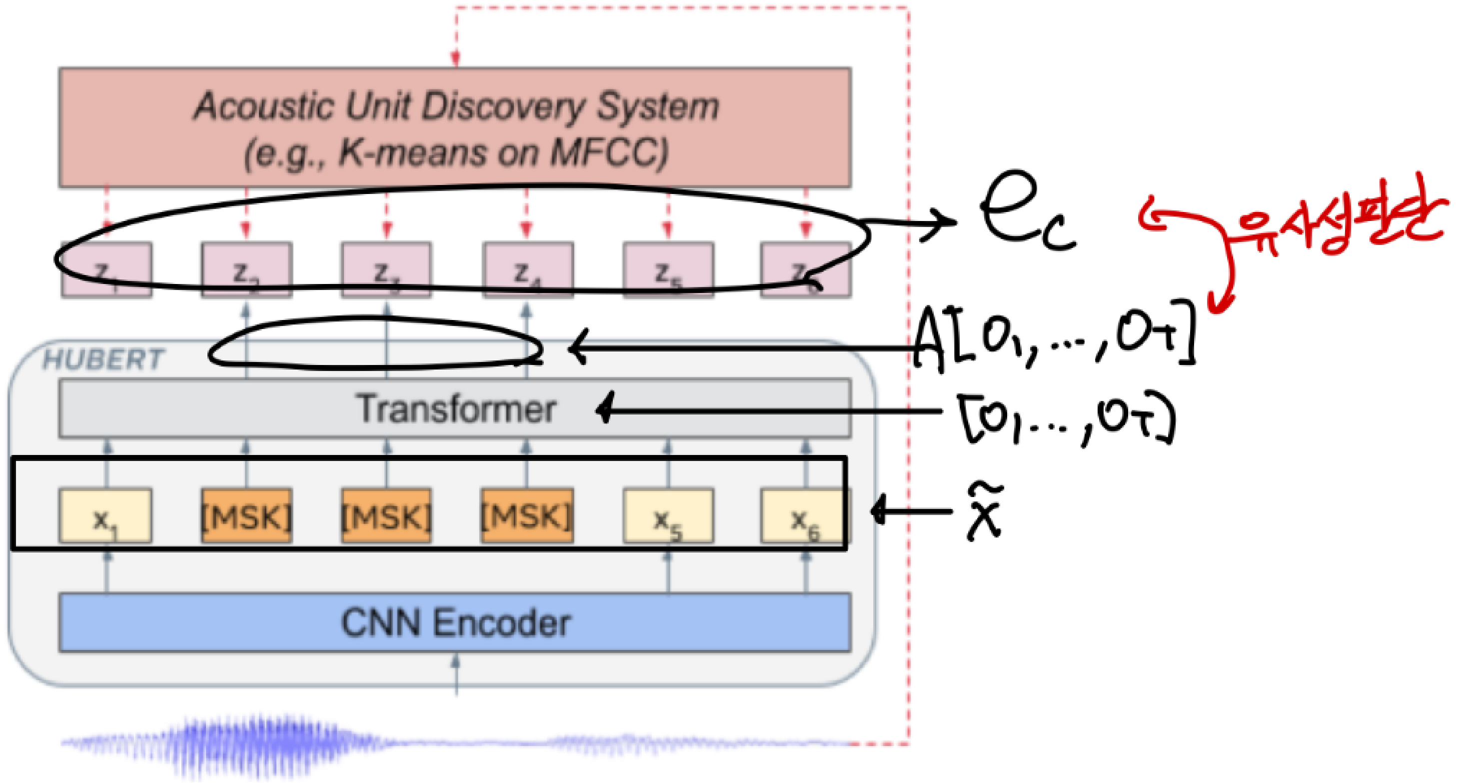
Loss를 어떻게 측정할 것인가

마스킹 처리 된 부분과
처리되지 않은 부분을 각각
CrossEntropy

$$L_m(f; X, M, Z) = \sum_{t \in M} \log p_f(z_t \mid \tilde{X}, t),$$

Codeword의 확률분포

HuBERT



$$p_f^{(k)}(c \mid \tilde{X}, t) = \frac{\exp(\text{sim}(A^{(k)} o_t, e_c)/\tau)}{\sum_{c'=1}^{C} \exp(\text{sim}(A^{(k)} o_t, e_{c'})/\tau)}$$

Codeword의 확률분포

입력이 주어졌을때
어느 codeword로매핑된 확률

비교 : Softmax $y_k = \frac{exp(a_k)}{\sum_{i=1}^{n} exp(a_i)}$

HuBERT: Self-Supervised Speech Representation

Result

Model	Unlabeled Data	LM	dev-clean	dev-other	test-clean	test-other
		10-min labeled				
DiscreteBERT [51]	LS-960	4-gram	15.7	24.1	16.3	25.2
wav2vec 2.0 BASE [6]	LS-960	4-gram	8.9	15.7	9.1	15.6
wav2vec 2.0 LARGE [6]	LL-60k	4-gram	6.3	9.8	6.6	10.3
wav2vec 2.0 LARGE [6]	LL-60k	Transformer	4.6	7.9	4.8	8.2
HUBERT BASE	LS-960	4-gram	9.1	15.0	9.7	15.3
HUBERT LARGE	LL-60k	4-gram	6.1	9.4	6.6	10.1
HUBERT LARGE	LL-60k	Transformer	4.3	7.0	4.7	7.6
HUBERT X-LARGE	LL-60k	Transformer	4.4	6.1	4.6	6.8
		1-hour labeled				
DeCoAR 2.0 [50]	LS-960	4-gram	-	-	13.8	29.1
DiscreteBERT [51]	LS-960	4-gram	8.5	16.4	9.0	17.6
wav2vec 2.0 BASE [6]	LS-960	4-gram	5.0	10.8	5.5	11.3
wav2vec 2.0 LARGE [6]	LL-60k	Transformer	2.9	5.4	2.9	5.8
HUBERT BASE	LS-960	4-gram	5.6	10.9	6.1	11.3
HUBERT LARGE	LL-60k	Transformer	2.6	4.9	2.9	5.4
HUBERT X-LARGE	LL-60k	Transformer	2.6	4.2	2.8	4.8
		10-hour labeled				
SlimIPL [54]	LS-960	4-gram + Transformer	5.3	7.9	5.5	9.0
DeCoAR 2.0 [50]	LS-960	4-gram	-	-	5.4	13.3
DiscreteBERT [51]	LS-960	4-gram	5.3	13.2	5.9	14.1
wav2vec 2.0 BASE [6]	LS-960	4-gram	3.8	9.1	4.3	9.5
wav2vec 2.0 LARGE [6]	LL-60k	Transformer	2.4	4.8	2.6	4.9
HUBERT BASE	LS-960	4-gram	3.9	9.0	4.3	9.4
HUBERT LARGE	LL-60k	Transformer	2.2	4.3	2.4	4.6
HUBERT X-LARGE	LL-60k	Transformer	2.1	3.6	2.3	4.0
		100-hour labeled				
IPL [12]	LL-60k	4-gram + Transformer	3.19	6.14	3.72	7.11
SlimIPL [54]	LS-860	4-gram + Transformer	2.2	4.6	2.7	5.2
Noisy Student [61]	LS-860	LSTM	3.9	8.8	4.2	8.6
DeCoAR 2.0 [50]	LS-960	4-gram	-	-	5.0	12.1
DiscreteBERT [51]	LS-960	4-gram	4.0	10.9	4.5	12.1
wav2vec 2.0 BASE [6]	LS-960	4-gram	2.7	7.9	3.4	8.0
wav2vec 2.0 LARGE [6]	LL-60k	Transformer	1.9	4.0	2.0	4.0
HUBERT BASE	LS-960	4-gram	2.7	7.8	3.4	8.1
HUBERT LARGE	LL-60k	Transformer	1.8	3.7	2.1	3.9
HUBERT X-LARGE	LL-60k	Transformer	1.7	3.0	1.9	3.5

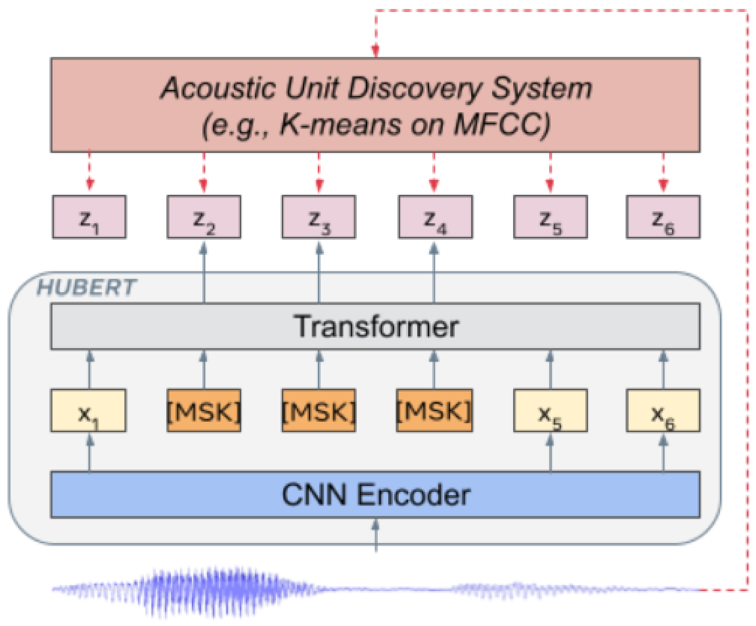
적은 시간으로 작은 WER 보임
SOTA인 wav2vec 2.0, Discrete BERT를 능가

Q) DiscreteBERT와HuBERT는모두
Masking방법으로Pre-training하는데,
성능 차이가 심한 이유?

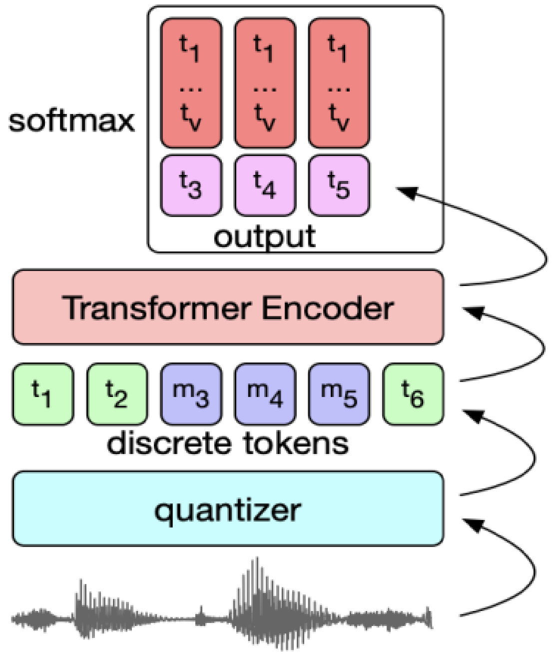
정보 손실을 피하기 위해 모델의 입력으로
파형(waveform)사용, 반복적 개선(iterativerefinement)

Result

HuBert



DiscreteBERT



(a) Quantized Inputs

적은 시간으로 작은 WER 보임
SOTA인 wav2vec 2.0, Discrete BERT를 능가

Q) DiscreteBERT와HuBERT는모두
Masking방법으로Pre-training하는데,
성능 차이가 심한 이유?

정보 손실을 피하기 위해 모델의 입력으로
파형(waveform)사용, 반복적 개선(iterativerefinement)

HuBERT: Self-Supervised Speech Representation

Result

feature	C	PNMI (mean ± std) with K-means Training Size = 1h	10h	100h
MFCC	100	0.251 ± 0.001	0.253 ± 0.001	0.253 ± 0.001
	500	0.283 ± 0.001	0.285 ± 0.000	0.287 ± 0.001
BASE-it1-L6	100	0.563 ± 0.012	0.561 ± 0.012	0.575 ± 0.008
	500	0.680 ± 0.005	0.684 ± 0.003	0.686 ± 0.004

TABLE IV: Stability of K-means as an unsupervised unit discovery algorithm with respect to different features, numbers of clusters, and training data sizes. PNMI stands for phone-normalized mutual information.

다양한 parameter에서
K-mean clustering의
표준편차가 안정적임

MFCC보다 HuBERT가
압도적인 PNMI Score

PNMI : Phone Normalized Mutual Information

Result

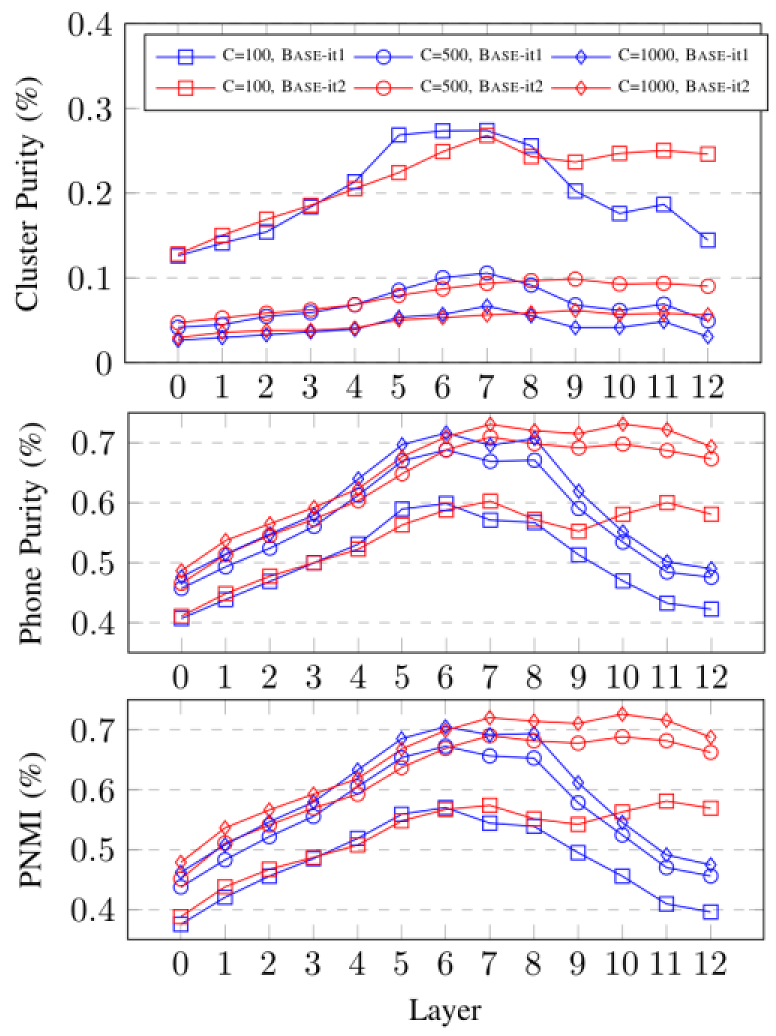


Fig. 2: Quality of the cluster assignments obtained by running k-means clustering on features extracted from each transformer layer of the first and the second iteration BASE HuBERT models.

iteration을 반복할 수 수록
순도가 높아지는 것을 확인할 수 있음

Conclusion

방법

HuBERT는
Masking 처리된 음성 데이터를 받아 K-means clustering 하여 모델 학습

성능

다양한 Finetuning dataset에 대해 SOTA 능가

Clustering 할당 개선

이전 반복의 학습된 표현을 사용하여 K-means clustering 개선

추후 연구과제

HuBERT의 훈련 절차를 단일 단계로 개선