HuBERT

...

https://arxiv.org/abs/2106.07447

Compiled by - Vaibhav

Abstract

Self-supervised approaches for speech representation learning are challenged by three unique problems:

- There are multiple sound units in each input utterance.
- There is no lexicon of input sound units during the pre-training phase.
- Sound units have variable lengths with no explicit segmentation.

HuBERT approach for self-supervised speech representation learning, which utilizes an offline clustering step to provide aligned target labels for a BERT-like prediction loss.

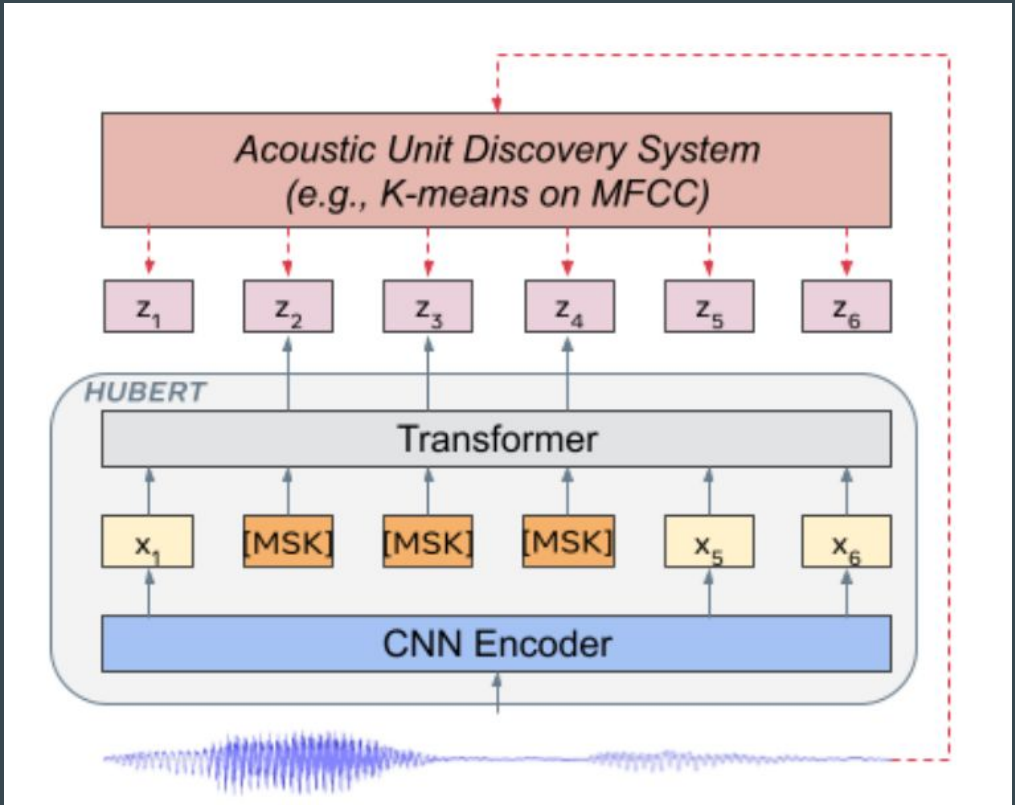
Abstract

A key ingredient of our approach is applying the prediction loss over the masked regions only, which forces the model to learn a combined acoustic and language model over the continuous inputs.

HuBERT relies primarily on the consistency of the unsupervised clustering step rather than the intrinsic quality of the assigned cluster labels.

Starting with a simple k-means teacher of 100 clusters, and using two iterations of clustering, the HuBERT model either matches or improves upon the state-of-the-art wav2vec 2.0 performance on the Librispeech.

Hidden Unit Selection Process



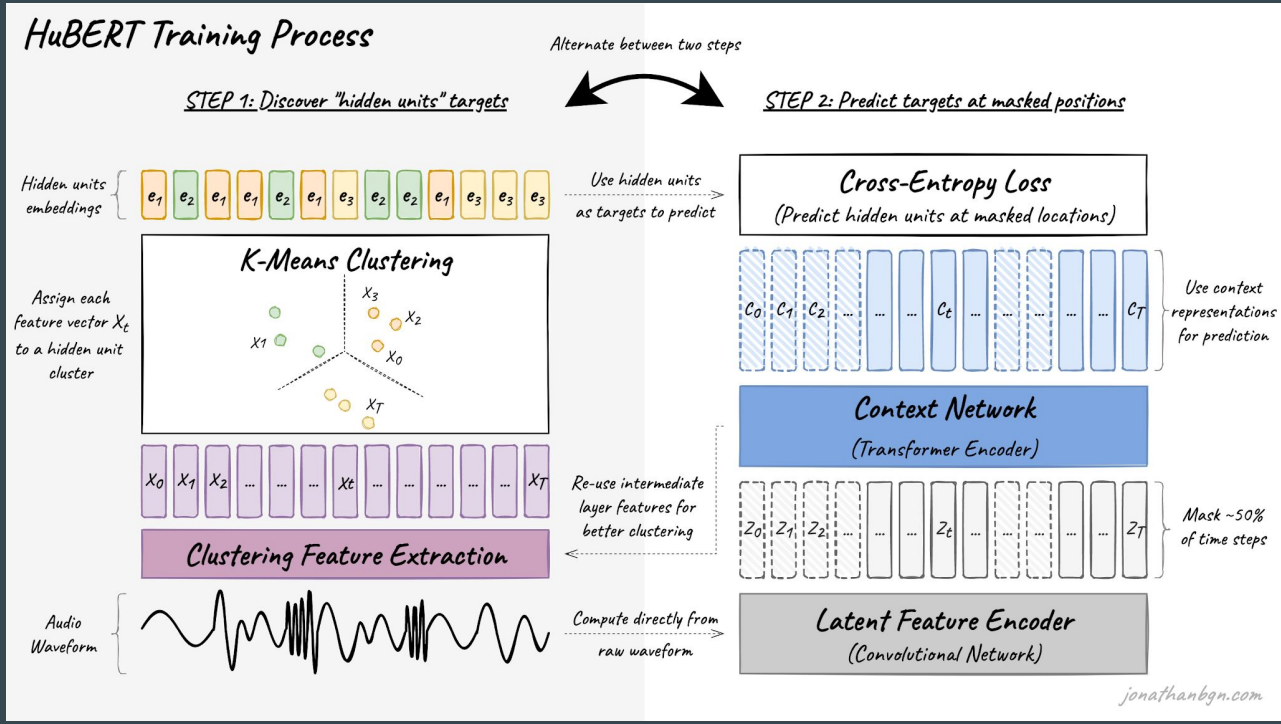
Original Paper

HuBERT Framework

The training process alternates between two steps:

a clustering step to create pseudo-targets,

and, a prediction step where the model tries to guess these targets at masked positions.



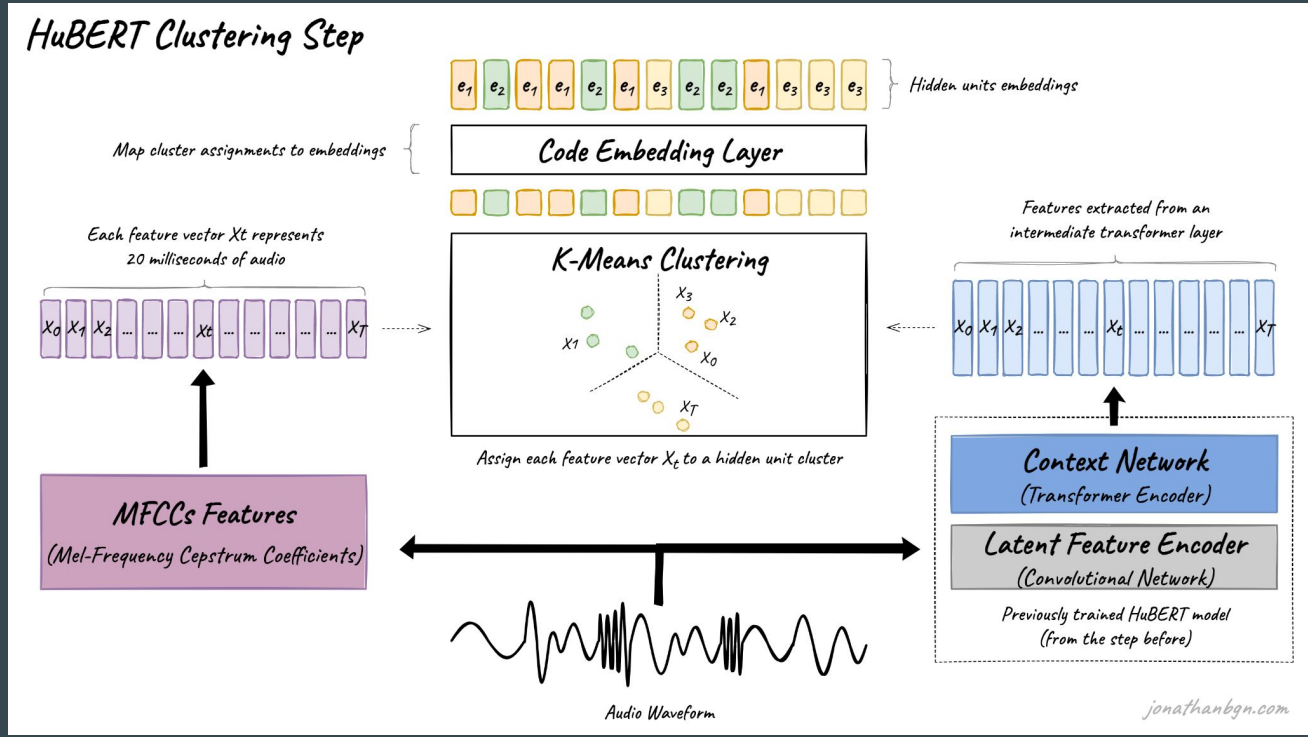
Jonathanbgn - HuBERT explained

Original Paper

Discover “hidden units” targets through clustering

Mel-Frequency Cepstral Coefficients (MFCCs) are used for the first clustering step.

However, for subsequent clustering steps, representations from an intermediate layer of the HuBERT transformer encoder (from the previous iteration) are re-used.



Discover “hidden units” targets through clustering

The first step is to extract the hidden units (pseudo-targets) from the raw waveform of the audio. The K-means algorithm is used to assign each segment of audio (25 milliseconds) into one of K clusters.

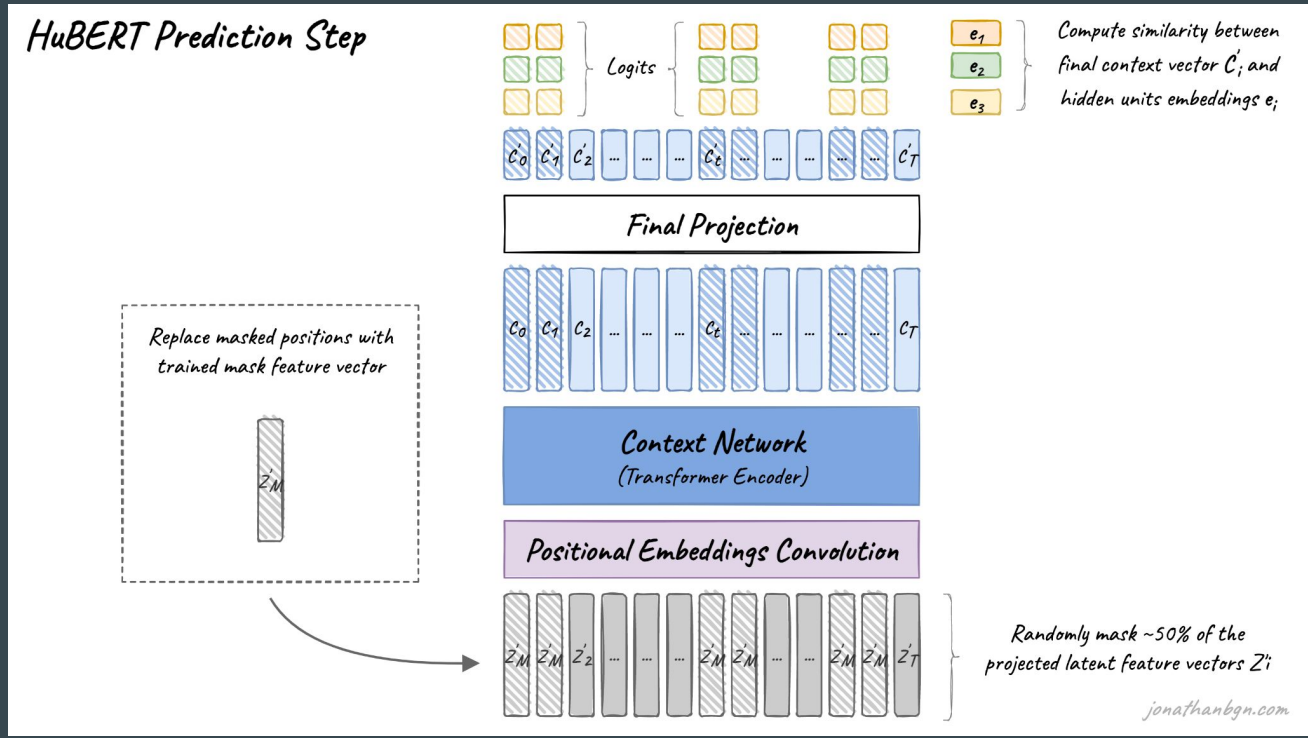
Each identified cluster will then become a hidden unit, and all audio frames assigned to this cluster will be assigned with this unit label.

Each hidden unit is then mapped to its corresponding embedding vector that can be used during the second step to make predictions.

Predict “noisy” targets from context

50% of transformer encoder input features are ***masked***, and the model is asked to predict the targets for these positions.

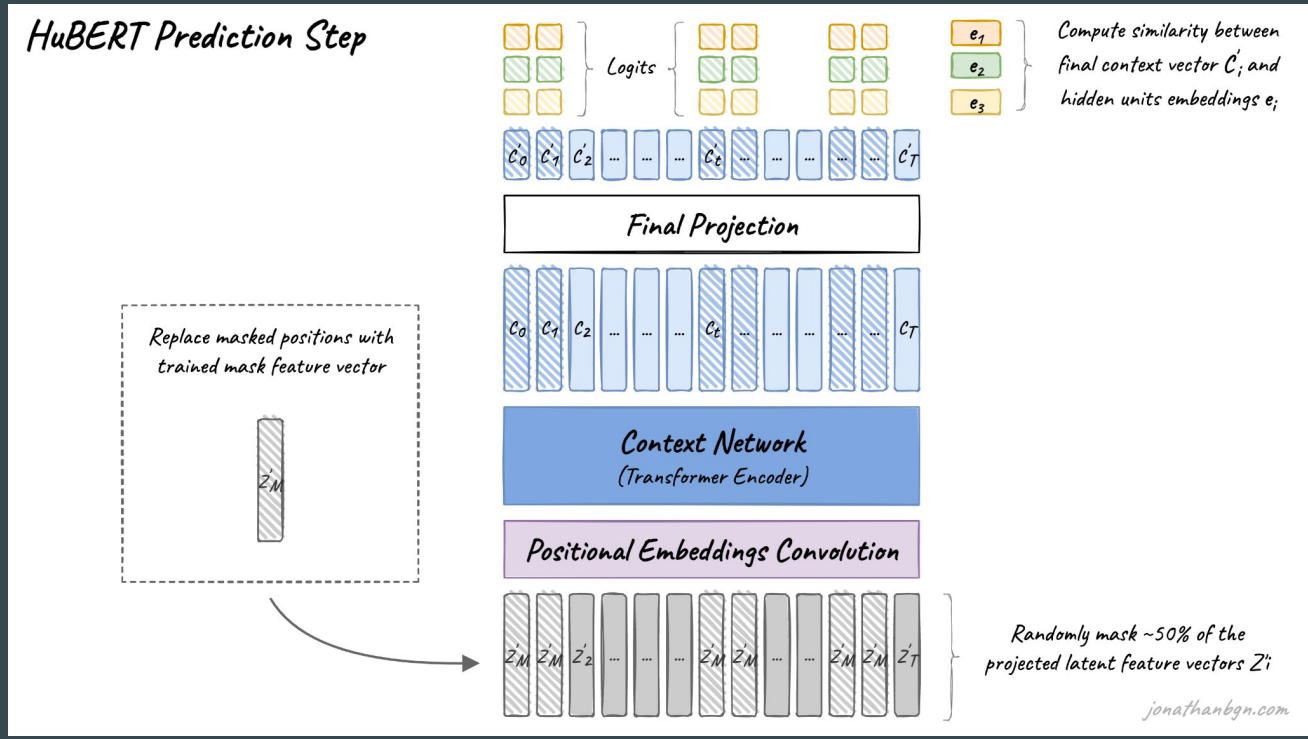
The *cosine similarity is computed between the transformer outputs and each hidden unit embedding* from all possible hidden units to give prediction logits.



Predict “noisy” targets from context

The ***cross-entropy loss*** is then used to penalize wrong predictions.

The loss is **only applied to the masked positions** as it has been shown to perform better when using noisy labels.



Fine Tuning

During fine-tuning, the convolutional waveform audio encoder parameters are fixed. Like wav2vec 2.0, the authors introduce a *freeze-step* hyperparameter to control how many fine-tuning steps the transformer parameters are fixed, and only the new softmax matrix is trained.

Results - Low Resource

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

Results - Low Resource

		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

Results - 960 hours

Model	Unlabeled Data	LM	dev-clean	dev-other	test-clean	test-other
		Superivsed				
Conformer L [62]	-	LSTM	-	-	1.9	3.9
		Self-Training				
IPL [12]	LL-60k	4-gram + Transformer	1.85	3.26	2.10	4.01
Noisy Student [61]	LV-60k	LSTM	1.6	3.4	1.7	3.4
		Pre-Training				
wav2vec 2.0 LARGE [6]	LL-60k	Transformer	1.6	3.0	1.8	3.3
pre-trained Conformer XXL [40]	LL-60k	LSTM	1.5	3.0	1.5	3.1
		Pre-Training + Self-Training				
wav2vec 2.0 + self-training [63]	LL-60k	Transformer	1.1	2.7	1.5	3.1
pre-trained Conformer XXL + Noisy Student [40]	LL-60k	LSTM	1.3	2.6	1.4	2.6
		This work (Pre-Training)				
HUBERT LARGE	LL-60k	Transformer	1.5	3.0	1.9	3.3
HUBERT X-LARGE	LL-60k	Transformer	1.5	2.5	1.8	2.9

Conclusion

This paper presents HuBERT, a speech representation learning approach that relies on predicting K-means cluster assignments of masked segments of continuous input.

The learned representation quality improves dramatically with iteratively refining K Means cluster assignments using learned latent representations for a previous iteration. Finally, HuBERT scales well to a 1B transformer model showing a relative reduction in WER of up to 13% on the test-other subset.

Given the high quality of its representations, we will consider using HuBERT pretrained representations for multiple downstream recognition and generation tasks beyond ASR.

Try HuBERT models on Hugging Face

-> All the models - Hugging Face HuBERT models

-> Test out the ASR performance on the base model via hosted inference or code - https://huggingface.co/facebook/hubert-large-ls960-ft

Discussion

- How do you think this architecture can be improved?
- Thoughts on Textless NLP via the Generative HuBERT?
- Multi-lingual HuBERT?