Are discrete units necessary for Spoken Language Modeling?

Tu Anh Nguyen, Benoit Sagot, Emmanuel Dupoux

***Abstract*—Recent work in spoken language modeling shows the possibility of learning a language unsupervisedly from raw audio without any text labels. The approach relies first on transforming the audio into a sequence of discrete units (or pseudo-text) and then training a language model directly on such pseudo-text. Is such a discrete bottleneck necessary, potentially introducing irreversible errors in the encoding of the speech signal, or could we learn a language model without discrete units at all? In this work, we study the role of discrete versus continuous representations in spoken language modeling. We show that discretization is indeed essential for good results in spoken language modeling. We show that discretization removes linguistically irrelevant information from the continuous features, helping to improve language modeling performances. On the basis of this study, we train a language model on the discrete units of the HuBERT features, reaching new state-of-the-art results in the lexical, syntactic and semantic metrics of the Zero Resource Speech Challenge 2021 (Track 1 - Speech Only).**

***Index Terms*—Spoken Language Modeling, Discrete Units, HuBERT**

I. Introduction

Pre-training language models on large-scale text data have achieved tremendous success in natural language understanding and have become a standard in Natural Language Processing (NLP) [1]–[5]. Recently, [3] showed that very large language models are actually few-shot learners, and manage to perform well even in zero-shot settings.

Large-scale self-supervised pre-training for speech data has also become more and more popular as a method to boost the performance of Automatic Speech Recognition (ASR) [6]–[8]. However, these models mostly rely on fine-tuning, which requires more training and text labels, to either improve the model or evaluate the learned representations of the speech. Lately, [9] introduces a new unsupervised task: Spoken language modeling, the learning of a language unsupervisedly from raw audio without any text labels, along with a suite of 4 zero-shot metrics probing for the quality of the learned models at different linguistic levels: phonetic, lexical, syntactic, semantic. The metrics are evaluated using the representations extracted from the model (phonetic, semantic) or pseudo-probability scores given by the model (lexical, syntactic). Their proposed baseline approach relies on transforming the audio into a sequence of frame-by-frame discrete units (or pseudo-text) and training a language model on the pseudo-text. The trained models displayed better-than-chance performances on nearly all the evaluation metrics of the challenge [9], [10]. However, this paradigm creates a discrete bottleneck between a speech encoder and a language model which could be a potential source of error, and in addition requires multiple training phases (learning an acoustic representation, clustering it, and learning a language model). Is such a discrete bottleneck necessary?

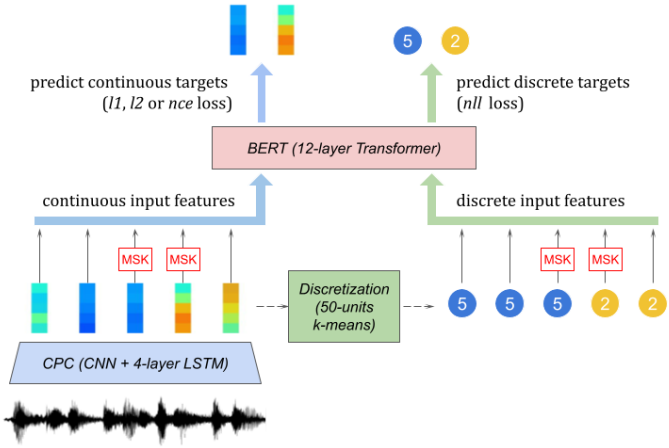

Fig. 1. Overview of the trained BERT models. The BERT model takes as input either the continuous features extracted from CPC or the sequences of frame-by-frame discretized units obtained from k-means, and tries to predict either continuous target features (with L1, L2 or NCE loss) or discrete target units (with NLL loss).

One way in which discrete units could help language modeling stems from the fact that in contrast to text, audio data contains a lot more details, some of which are linguistically relevant (intonation, rhythm, non verbal vocalization), others not so (background noise, reverberation, speaker identity, etc). To the extent that discretization effectively removes linguistically irrelevant information from the continuous features [11], it could indeed help language modeling. Of course, this potential gain could be counterbalanced by the fact that discretization could also make errors and remove useful information.

In this work, we analyse the importance of discretization in spoken language modeling. We employ a pre-trained acoustic model to obtain either continuous or discretized features from audio data. We then train BERT language models with a Masked Language Modeling (MLM) objective on both discrete and continuous features used either as inputs or as targets and evaluate the resulting systems on zero-shot spoken language modeling metrics. We also evaluate HuBERT [7], a single

Tu Anh Nguyen is with Meta and Inria, France, e-mail: nguyentu-anh208@gmail.com
Benoit Sagot is with Inria, France, e-mail: benoit.sagot@inria.fr
Emmanuel Dupoux is with Meta and EHESS, ENS-PSL, CNRS, Inria, France, e-mail: emmanuel.dupoux@gmail.com

model trained from raw waveform with discrete targets, on these metrics and compare the results with our best models.

Our contributions can be listed as follows:

- We show experimentally that discretization is beneficial for spoken language modeling, but we can get rid of discrete bottlenecks by using low-level continuous inputs so long as we still use discrete targets.
- We show that discretization disentangles linguistic information from non-linguistic signals, forcing the transformer to focus on linguistic ones.
- We show that a self-supervised model trained with a MLM objective on discrete targets like HuBERT achieves very good results on spoken language modeling metrics, showing that it can learn not only acoustic but also high-level linguistic information.

II. Related Work

a) Discretization in Self-Supervised Approaches: Self-supervised models for learning speech representation have become more and more popular as an effective pre-training method for downstream Automatic Speech Recognition (ASR) task, notably wav2vec2.0 [6] and HuBERT [7]. Both models comprise a feature extractor (CNN Encoder) followed by a feature encoder (Transformer Encoder), and are trained with a MLM objective like BERT. However, wav2vec2.0 discretizes the latent features obtained by the CNN Encoder and uses them as the target for the Transformer Encoder using a contrastive loss against negative samples in the sentence. On the other hand, HuBERT discretizes fixed features obtained from a teacher model and uses these fixed discrete units as the target for the Transformer Encoder using a cross-entropy loss. Finally, our work is mostly similar to [12], where they compare BERT models training on discrete units obtained from vq-wav2vec [13] and continuous features obtained from wav2vec [14] on the ASR task. They found that training BERT model on discrete vq-wav2vec units is more effective for ASR.

b) Spoken Language Modeling: Following the huge success of language models on text data [2]–[4], the Zero Resource Speech Challenge 2021 [9], [10] opens up new possibilities for learning high-level language properties from raw audio without any text labels. They introduced 4 zero-shot evaluation metrics at different linguistic levels (phonetic, lexical, syntactic, semantic), along with composite baseline systems consisting of an acoustic discretization module (Contrastive Predictive Coding, or CPC+k-means) followed by a language model (BERT or LSTM) on the discretized units. The CPC model takes the raw audio as input and produces phonetic representations at a lower frame rate of 100Hz, helping the language model to learn high-level information from the raw audio. In the same spirit, [15] introduced Generative Spoken Language Modeling (GSLM), the task of learning and generating spoken language from raw audio only. They provided baseline systems consisting of a discrete speech encoder (CPC, wav2vec 2.0, HuBERT), a generative language model (GPT-like model), and a speech decoder (Tacotron-2, [16]). The models are evaluated on spoken language modeling metrics [9], ASR-based generation metrics [15] as well as human evaluation metrics.

III. Experimental Setup

In this section, we first present the evaluation metrics as well as the dataset used to train and evaluate the models. We then explain our models and the inference methods for model evaluation.

A. Evaluation Metrics

We evaluate our models with the ZeroSpeech 2021 Benchmark Metrics [9], consisting of 4 zero-shot tests probing for the quality of spoken language models at four linguistic levels: phonetic (Libri-light ABX metrics), lexical (sWUGGY spot-the-word metrics), syntactic (sBLIMP acceptability metrics) and semantic (sSIMI similarity metrics).

a) Libri-light ABX metrics: Given a pair of similar triphones (e.g., 'aba'-'apa') spoken by a same speaker and an intervening sound (either 'aba' or 'apa'), the model has to tell which sound has a closer representation to the intervening sound. The ABX metrics is reported as the error rate that the model fails to choose the correct triphone.

b) sWUGGY spot-the-word metrics: Given a pair of a word and a similar non-word (e.g., 'brick'-'blick'), the model has to tell which is the word based on their probability. The spot-the-word metrics is reported as the accuracy that the model assigns a higher probability to the word.

c) sBLIMP acceptability metrics: Given a linguistic minimal sentence pair of matched grammatical and ungrammatical sentences (e.g., 'he loves it'-'he love it'), the model has to tell which is the grammatical sentence. The acceptability metrics is reported as the accuracy that the model assigns a higher probability to the grammatical sentence.

d) sSIMI similarity metrics: Given a pair of words (e.g., 'happy'-'joyful'), the model has to compute a similarity score based on their representations. The similarity metrics is reported as the Pearson correlation coefficient (PCC) between model scores and human judgements. In this work, the sSIMI scores are weighted across different subsets according to their sizes and averaged across LibriSpeech and synthetic subsets to make it more accurate and consistent. We reported it as wSIMI.

B. Datasets

a) Training Dataset: We train our models on LibriSpeech [17], an English corpus containing 1000 hours of read speech based on public domain audio books. The models are validated on LibriSpeech dev-clean and dev-other subsets, comprising 10 hours of speech in total.

b) Metrics Datasets: The metrics datasets are either extracted sounds from LibriSpeech (ABX, sSIMI) or synthesised using Google API[1] (sWUGGY, sBLIMP, sSIMI). The datasets containing words or sentences were filtered to only contain the LibriSpeech vocabulary (except sWUGGY non-words), and are split into dev and test sets. The dev sets have been made publicly available at the ZeroSpeech 2021 Challenge website [2].

[1] https://cloud.google.com/text-to-speech

[2] https://zerospeech.com/2021/instructions.html#evaluation-dataset

C. Models

a) ZeroSpeech 2021 Baseline: The ZeroSpeech 2021 Baseline System [9] is a composite of three components: an acoustic model (CPC, [18], [19]), a clustering module (k-means) and a language model (BERT, [4]). The CPC model is first trained to obtain good phonetic representations of the speech, which are then discretized into sequences of units with the k-means model. The BERT model is finally trained on these discrete units to better learn linguistic information.

As we only focus on the language modeling system in this work, we shall use the best CPC model in the ZeroSpeech 2021 Baseline System, which comprises a 5-layer 1D-CNN Encoder followed by a 4-layer LSTM autoregressive model. The features are extracted from the 2nd layer (unless otherwise specified) of the LSTM model, with a rate of 100Hz, and are either discretized with a 50-unit k-means model (discrete) or left unchanged (continuous).

b) BERT with discrete and continuous features: We modify the BERT model so that it is able to take as input either discrete units obtained from k-means or continuous features extracted from CPC, in which case the masking is done by replacing the features with a masked embedding vector. We also allow the model to predict either discrete target units or continuous target features, with multiple choices of an appropriate objective for each case. When predicting discrete targets, we use a cross-entropy objective (Negative Log-Likelihood, or NLL loss) but with two slightly different implementations. We could simply employ a linear classification head at the output of the BERT model as usual (which we denote by *linear NLL*, or NLL-l) or force the BERT output features to be similar to the embedding vectors of the target units as for HuBERT (cf. equation (3) from [7], we denote this by *embedding NLL*, or NLL-e). In the case of continuous targets, it can be a reconstruction objective (L1 loss or L2 loss) or a contrastive objective (Noise Contrastive Estimation, or NCE loss). In the latter case, the predicted features are contrasted with 100 negative features sampled from the same phrase (similar to continuousBERT, [12]).

We use a BERT base model, which comprises a 12-layer Transformer Encoder. Our implementation is based on the wav2vec2.0 [6] Transformer Encoder [3] using fairseq [20]. Each input sequence contains the features of a full audio file, and we consider at most 15.6 seconds of audio per file. We trained all models for 250k update steps on 32 GPUs, with a batch size of 175s per GPU. The learning rate was warmed up to a peak value of 1×10^{-5} after 32k steps. For the masking, we masked M consecutive tokens for each span, where $M \sim \mathcal{N}(10, 10)$, with a total masking coverage of roughly half of the input tokens (spans may overlap).

D. Model Inference for Evaluation

a) ABX Distance: For the ABX metrics, we extract frame-by-frame representation features for each audio file. Then, the ABX distance between two files is computed as the average angular distance of the representations along the realigned Dynamic Time Wrapping path. Given two audio files x and y with two sequences of representation $\mathbf{r}^x = r_1^x, \ldots, r_T^x$ and $\mathbf{r}^y = r_1^y, \ldots, r_S^y$ respectively, the ABX distance between x and y is computed as follows:

$$d_{ABX}(x, y) = \frac{1}{|\text{path}_{\text{DTW}}(\mathbf{r}^x, \mathbf{r}^y)|} \sum_{(i,j) \in \text{path}_{\text{DTW}}(\mathbf{r}^x, \mathbf{r}^y)} sim(r_i^x, r_j^y), \tag{1}$$

where $sim(r_i^x, r_j^y)$ is the angular distance (in radian) between the embeddings r_i^x and r_j^y.

We note that in this paper the ABX metrics are mainly used to evaluate the input and target features of the BERT model, and therefore the ABX distances are mostly performed on the CPC features without using the BERT model.

b) Probability Estimation: For sWUGGY and sBLIMP metrics, we compute for each audio file a model-based pseudo log-probability (m-PLP) of the trained BERT model. Given an audio file x with the input and target features for the BERT model $x_1 ... x_T$ and $\hat{x}_1 ... \hat{x}_T$ respectively, the m-PLP is computed as follows:

$$\text{m-PLP}(x) = \sum_{\substack{j=0 \\ i=j\Delta t}}^{\lfloor (T-M)/\Delta t \rfloor} \sum_{m=1}^{M} PLP(\hat{x}_{i+m} | \overline{x_{i+1} .. x_{i+M}}), \tag{2}$$

where M is a chosen size of a sliding window, Δt is a chosen step of the sliding window and $PLP(\hat{x}_{i+m} | \overline{x_{i+1} .. x_{i+M}})$ is a pseudo log-probability of the target $\hat{x}_{i+m}$ given by the BERT model with M-span masked inputs $x_1 .. x_i m .. m x_{i+M+1} .. x_T$ (m represents a masked feature).

For models with NLL or NCE loss, $PLP(\hat{x}_i | \overline{x_{i+1} .. x_{i+M}})$ is computed as the log value of the probability given by the softmax layer of the BERT model (in the NLL case, the probability is computed over all tokens, while in the NCE case it is computed over all sampled negative examples). For models with L1 or L2 loss, we compute $PLP(\hat{x}_i | \overline{x_{i+1} .. x_{i+M}})$ as the negative reconstruction loss of the predicted feature and the target feature $\hat{x}_i$. The negativity ensures that a correct target has a higher m-PLP.

The m-PLP extends the span-masked pseudo probability (span-PP) [9] to BERT models with continuous targets. It is derived from the pseudo-loglikelihood score (PLL) for MLMs [21], which was shown to be an effective sentence scoring method for BERT models in many scenarios [22].

The choice of M and Δt is determined for each model using the dev sets, and is given in Table VI. In our experiments, we always consider $\Delta t = 5$ and vary M in $\{15, 25, 35, 45, 55\}$. For models trained on HuBERT features (section IV-C), we vary M in $\{5, 10, 15, 20, 25\}$ as the frame rate is 50Hz instead of 100Hz as for CPC.

c) Similarity Score: For the sSIMI metrics, we extract a fixed-length representation for each audio file by applying a pooling function (mean, max, min) over hidden features from one layer of the Transformer Encoder. The similarity score of two audio files is computed as the cosine similarity between the two corresponding representations. The choice of the hidden layer and the pooling function is determined for each model using the dev sets and is given in Table VI.

[3] https://github.com/pytorch/fairseq/tree/main/examples/wav2vec

IV. Results

A. Discrete bottleneck seems to be essential for spoken language modeling

Table I reports the performances of our BERT models, trained with either continuous or discrete CPC features of the LibriSpeech 960h dataset, on lexical (sWUGGY), syntactic (sBLIMP) and semantic (wSIMI) metrics.

id	input	target	loss	sWUGGY↑	sBLIMP↑	wSIMI↑
	discrete input, discrete target					
1	disc.	disc.	NLL-l	79.28	59.71	6.32
2	disc.	disc.	NLL-e	**80.02**	**59.86**	**7.87**
	continuous input, discrete target					
3	*cont.*	disc.	NLL-l	**60.36**	**53.23**	8.39
4	*cont.*	disc.	NLL-e	60.20	52.78	**9.49**
	continuous input, continuous target					
5	*cont.*	*cont.*	NCE	56.84	52.62	**9.16**
6	*cont.*	*cont.*	L1	59.23	53.12	7.85
7	*cont.*	*cont.*	L2	**60.56**	**53.33**	6.55
	discrete input, continuous target					
8	disc.	*cont.*	NCE	65.69	**57.24**	9.33
9	disc.	*cont.*	L1	73.93	56.02	**10.69**
10	disc.	*cont.*	L2	**74.22**	55.75	5.97

TABLE I
PERFORMANCES ON THE DEV SETS OF SWUGGY, SBLIMP, WSIMI METRICS OF BERT MODELS USING EITHER CONTINUOUS ZEROSPEECH CPC FEATURES (LAYER 2 OF THE LSTM MODULE OF CPC-BIG) OR DISCRETIZED FEATURES (WITH A 50-UNIT K-MEANS MODEL) AS INPUTS AND TARGETS. BEST SCORES IN EACH CATEGORY ARE IN BOLD, BEST SCORES OVERALL ARE UNDERLINED.

We first examine how the continuity of the input and target features affects the quality of the BERT model on the evaluation metrics. By comparing the best scores in each case, we see that having discrete inputs helps the model learn better lexical and syntactic information, whereas models with continuous inputs do have better than chance performance on the lexical task. We observe that the best models on the language model tasks are obtained with discrete inputs and discrete targets, which is the classic configuration of BERT. Predicting continuous targets from discrete inputs, where the model acts as an autoencoder decoder, is also beneficial and nearly catches up with the best models. It is interesting, still, to note that it is possible to acquire some language information without any discretization. The wSIMI scores are still quite low, but we see in general that having continuous information does help.

B. Is continuous input always bad?

We observe during our training experiments that the masked prediction objective is too easy for some models with continuous inputs and could quickly lead to overfitting. This could be explained by the fact that the input and target features are extracted from the same layer of the LSTM autoregressive module of CPC. As a consequence, we try using the input features from different layers of the LSTM module, while maintaining the same target layer. We keep using the NLL-e loss for discrete targets while using NCE and L1 loss for continuous targets. The results are reported in Table III.

		ABX within↓ clean	ABX within↓ other	ABX across↓ clean	ABX across↓ other
layer 0	*cont.*	11.50	14.09	18.53	24.70
	disc.	21.46	24.21	30.77	34.91
layer 2	*cont.*	3.41	4.84	4.20	7.65
	disc.	6.38	10.22	8.22	14.86
layer 4	*cont.*	9.49	11.95	10.01	15.70
	disc.	19.81	21.64	24.39	28.04

TABLE II
WITHIN AND ACROSS SPEAKER ABX ERROR (LOWER IS BETTER) ON LIBRI-LIGHT DEV-CLEAN AND -OTHER FOR CONTINUOUS AND DISCRETIZED FEATURES OF DIFFERENT LAYERS OF THE LSTM AUTOREGRESSIVE MODULE OF CPC-BIG MODEL. LAYER 0 MEANS THE OUTPUT OF THE CNN ENCODER MODULE.

We observe that using continuous input features from a different layer does reduce overfitting during training, which significantly improves the performances of the models on LM metrics, especially for sWUGGY scores. Interestingly, we note that using continuous input features from a lower LSTM layer (layer 0, where the ABX errors are high, cf. Table II) to predict target features from a higher LSTM layer (layer 2) is more beneficial to the model than using high quality continuous input features from the same or higher layer as the target features (layer 2, layer 4). This is not the case, however, for discrete input models, where the model benefits from good quality input units.

id	input	target	sWUGGY↑	sBLIMP↑	wSIMI↑
	discrete input, discrete target, NLL-e loss				
2	layer 2	layer 2	**80.02**	**59.86**	7.87
11	layer 0	layer 2	64.91	52.45	7.48
12	layer 4	layer 2	70.68	55.06	**8.61**
	continuous input, discrete target, NLL-e loss				
4	*layer 2*	layer 2	60.20	52.78	**9.49**
13	*layer 0*	layer 2	**77.19**	**55.30**	7.25
14	*layer 4*	layer 2	67.41	54.13	8.06
	continuous input, continuous target, NCE loss				
5	*layer 2*	*layer 2*	56.84	52.62	**9.16**
15	*layer 0*	*layer 2*	**65.53**	**55.20**	6.17
16	*layer 4*	*layer 2*	59.81	52.93	8.32
	continuous input, continuous target, L1 loss				
6	*layer 2*	*layer 2*	59.23	53.12	**7.85**
17	*layer 0*	*layer 2*	**67.83**	**53.59**	7.25
18	*layer 4*	*layer 2*	63.68	53.26	6.90
	discrete input, continuous target, NCE loss				
8	layer 2	*layer 2*	**65.69**	**57.24**	**9.33**
19	layer 0	*layer 2*	58.55	52.31	8.07
20	layer 4	*layer 2*	58.61	54.48	7.72
	discrete input, continuous target, L1 loss				
9	layer 2	*layer 2*	**73.93**	**56.02**	**10.69**
21	layer 0	*layer 2*	62.98	53.47	5.20
22	layer 4	*layer 2*	65.92	53.94	7.01

TABLE III
PERFORMANCES ON THE DEV SETS OF SWUGGY, SBLIMP, WSIMI METRICS OF BERT MODELS USING THE INPUT FEATURES FROM DIFFERENT LAYERS OF THE LSTM MODULE OF CPC-BIG MODEL. LAYER 0 MEANS THE OUTPUT OF THE CNN ENCODER MODULE. BEST SCORES IN EACH CATEGORY ARE IN BOLD, BEST SCORES OVERALL ARE UNDERLINED.

Overall, we observe that discrete-discrete model (with the

same input and target units) yields the best performance when the quality of discrete units is good. Continuous-discrete is also a great choice when using low-level input features. When there are no discrete units at all, the LM performances are still limited, even if using a NCE loss could help a bit with syntactic and semantic metrics.

C. Varying the number of discrete units

Here, we address the question as to why discrete units are better than continuous ones. One hypothesis is that discrete units manage to remove linguistically irrelevant information and force the transformer to focus on linguistic ones. To test this, we run a speaker discrimination probe on the discrete units and continuous features. In addition, we run a new experiment varying the number of discrete units from 20 to 2000. Hypothetically, when the number of units is too small (eg, smaller than the number of phonemes), the resulting phonetic confusions should degrade the learning of higher linguistic representations. Conversely, when the number of units is too large, the quantization step would start to leak other-than-phonetic information into the representation, hence making it closer to the continuous representations.

model	n units	unit quality		language modeling on units		
		spk prb↑	ABX↓	sWUGGY↑	sBLIMP↑	sSIMI↑
CPC	20	30.40	12.66	71.71	58.97	4.72
	50	34.00	9.89	80.02	**59.86**	**7.87**
	100	49.20	9.56	**80.47**	59.47	6.09
	200	56.00	9.72	79.90	58.90	4.45
	500	64.00	10.72	79.66	59.72	6.37
	1000	61.60	11.99	79.86	58.46	6.78
	2000	67.60	14.24	78.46	58.25	5.94
	cont.	*98.00*	*5.02*		-	
HuBERT	20	24.40	14.04	62.89	57.06	7.80
	50	38.00	9.19	76.79	61.12	8.61
	100	48.00	8.34	81.09	62.47	5.19
	200	61.60	7.57	81.54	62.78	7.03
	500	68.40	7.73	**83.06**	**62.89**	9.73
	1000	74.40	9.04	82.58	61.55	8.64
	2000	73.20	11.00	81.61	62.85	**10.66**
	cont.	*99.60*	*4.23*		-	
Forced Phones	*40*	*10.00*	*0.00*	*92.19*	*63.72*	*6.23*

TABLE IV
DISCRETE UNIT QUALITY (SPEAKER PROBING AND ABX) AND PERFORMANCE OF THE BERT MODELS TRAINED ON DISCRETE UNITS ON THE DEV SETS OF LM SCORES (SWUGGY, SBLIMP, WSIMI) FOR DIFFERENT NUMBERS OF CLUSTERS ON CPC AND HUBERT FEATURES. THE ABX IS AVERAGED ON DEV-CLEAN AND DEV-OTHER WITHIN AND ACROSS SUBSETS.

To support our hypothesis, we run kmeans on the continuous features of both CPC and HuBERT models, and vary k to be 20, 50, 100, 200, 500, 1000, and 2000, after which we train a discrete-discrete BERT model. For the CPC features, we take the layer 2 features of the CPC-big model as usual. For the HuBERT features, we train our own HuBERT base model as described in Section IV-D, we then take the features from layer 12 of the Transformer Encoder after the 2nd iteration, which have the best ABX (cf. Table VII). Following [23], we train a speaker classifier in the following way: We randomly split LibriSpeech dev-clean utterances into train/valid/test (80%/10%/10%) sets and train a two-layer

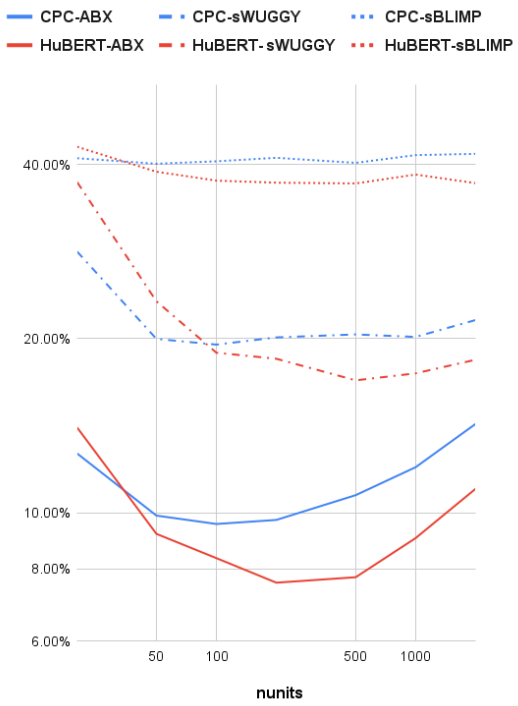

Fig. 2. ABX of Discrete units and Error rate on the dev sets of LM scores (sWUGGY, sBLIMP) for different numbers of clusters for CPC and HuBERT features. The ABX is averaged on dev-clean and dev-other within and across subsets.

Transformer classifier on the sequences of discrete units or continuous features of the utterances. The classification head is performed on the first token (`bos`, or begin-of-sentence) of the transformer outputs. For this speaker probing task, there are 40 classes (speakers). The models are trained for 20 epochs and are validated on the valid set. We finally report the test accuracy. For reference, we also include the forced phonemes units (frame-by-frame phonemes). As the forced phonemes contain a silence, there are 40 units in total.

The results are reported in Table IV and illustrated in Figure 2. As expected speaker classification accuracy increases with the number of clusters, and the continuous features yield the best classification. We can observe a U-shaped curve in performance across the different language metrics as a function of the number of units. Interestingly, the optimum number of units seems to be different across the model features (CPC, HuBERT) and linguistic levels. HuBERT features are better than CPC features in most cases, and seem to benefit from more clusters than CPC features. In general, we see that the language model scores seem to decrease slowly compared to ABX as the number of clusters becomes bigger. It is also interesting to note that the language model scores become steadily good as soon as the number of clusters is higher than the number of phonemes (40 units). This could be seen in Figure 3, where we observe a limited unit-phoneme correspondence when having only 20 discrete units; but as soon as the number of clusters reaches 50, we see a clear correspondence between the units and the phonemes, although several "hard" phonemes are still dispersed and don't correspond to a single unit (e.g. ch,

id	*Systems* input	target	loss	ABX (target features)↓ within clean	within other	across clean	across other	sWUGGY↑	sBLIMP↑	wSIMI↑
	ZeroSpeech 2021 Best Baseline System [9]									
	CPC-layer2+km50	CPC-layer2+km50	NLL-l	6.71	10.62	8.41	15.06	75.51	56.16	2.05
	ZeroSpeech 2021 Text Topline Systems [9]									
	Forced phones	Forced phones	NLL-l	0.00	0.00	0.00	0.00	91.88	63.16	4.44
	Phones	Phones	NLL-l	-	-	-	-	97.67	66.91	12.80
	BERT Models on CPC-big Features									
2	CPC-layer2+km50	CPC-layer2+km50	NLL-e	6.71	10.62	8.41	15.06	**80.29**	**59.93**	6.56
13	*CPC-layer0*	CPC-layer2+km50	NLL-e	6.71	10.62	8.41	15.06	77.22	55.62	**6.61**
17	*CPC-layer0*	*CPC-layer2*	L1	3.28	4.81	4.31	7.92	68.37	53.95	5.68
9	CPC-layer2+km50	*CPC-layer2*	L1	3.28	4.81	4.31	7.92	74.46	55.38	6.17
	HuBERT Base Models									
23	*waveform*	MFCC+km100	NLL-e	20.22	24.97	33.42	40.45	62.74	54.11	5.58
24	*waveform*	H-iter1-layer6+km500	NLL-e	6.29	7.51	8.76	12.82	79.13	58.89	5.45
25	*waveform*	H-iter2-layer12+km500	NLL-e	5.87	7.15	6.96	10.73	**80.19**	**59.29**	**5.87**
	BERT Models on HuBERT Discrete Units									
26	H-iter2-layer12+km500	H-iter2-layer12+km500	NLL-e	5.87	7.15	6.96	10.73	**83.29**	**61.93**	**9.73**

TABLE V

COMPARISON ON THE TEST SETS OF THE 4 ZEROSPEECH 2021 METRICS OF OUR BERT MODELS TRAINED ON CONTINUOUS OR DISCRETE CPC FEATURES, BERT MODEL TRAINED ON HUBERT DISCRETE UNITS AND HUBERT BASE MODELS WITH ZEROSPEECH 2021 BASELINE AND TOPLINE SYSTEMS. FOR EACH CONTINUOUS/DISCRETE COMBINATION, WE CHOOSE THE BEST PERFORMING MODEL ON THE DEV SET AS REPORTED IN TABLE III. WE TRAINED THE HUBERT MODEL FOR 3 ITERATIONS. THE TARGETS USED TO TRAIN THE 3 ITERATIONS ARE DISCRETIZED MFCC FEATURES (100 UNITS), DISCRETIZED FEATURES FROM TRANSFORMER'S LAYER6 OF 1ST ITERATION (500 UNITS) AND DISCRETIZED FEATURES FROM TRANSFORMER'S LAYER12 OF 2ND ITERATION (500 UNITS) RESPECTIVELY. ALL MODELS WERE TRAINED ON THE LIBRISPEECH 960H DATASET. FOR THE ABX METRICS, WE REPORT THE SCORES ON THE TARGET FEATURES USED TO TRAIN THE MODEL. BEST SCORES IN EACH CATEGORY ARE IN BOLD, BEST SCORES OVERALL ARE UNDERLINED.

oy, th, uh); when there are 500 clusters, there are more units representing a single phoneme, and most "hard" phonemes are now assigned by certain units.

These results support the hypothesis that the superiority of the discrete units is due to the fact that they block the propagation and amplification of non-linguistic signals that may be present (even if attenuated) in continuous representations.

D. Comparison with state-of-the-art systems

We evaluate the HuBERT model on the zero-shots metrics and compare the results with our trained BERT models. The HuBERT model is trained iteratively, using clustering units from features of previous iteration as the teacher. We trained a HuBERT base model, which comprises a 7-layer CNN Encoder followed by a 12-layer Transformer Encoder, on the Librispeech 960h dataset for 3 iterations. The teachers for each iteration are MFCC features (100 units), Transformer's layer6 of 1st iteration (500 units) and Transformer's layer12 of 2nd iteration (500 units) respectively. Architecturally, the Transformer Encoder of the HuBERT model is very similar to our model 13 (*continuous input layer 0, discrete target layer 2, NLL-e loss*) where they both take as input the continuous features of the CNN Encoder and predict discrete targets obtained from features of a higher level with a NLL-eloss.

Overall performances on the ZeroSpeech 2021 test sets are reported in Table V. For each of discrete/continuous combinations, we choose the best performing model on the dev set as reported in Table III. We also include the discrete-discrete model trained on HuBERT Discrete Units (500 units), which was reported to have the best LM scores in section IV-C. We first observe a huge improvement of model 2 compared with the baseline system, even if they both use the same units for the BERT model. This improvement greatly comes from the reimplementation of the BERT model, which uses the wav2vec2 Transformer Encoder model[4]. Changing the NLL-l loss to NLL-e loss also improves a little bit (cf. Table I).

It seems that using good quality discrete units as targets is very beneficial for the language models, achieving better scores than using continuous targets in all the metrics. The HuBERT model performs surprisingly well, approaching our best model on the language model tasks. This means that the Transformer Encoder of HuBERT acts as a language model as well. We see that as soon as the discrete targets have better quality, the HuBERT model manages to have better results on spoken language modeling metrics. We see that the discrete-discrete model on HuBERT Discrete Units (model 26) further improves the scores on all the metrics, confirming again our finding that it's better to train a discrete-discrete model when we have good quality units.

Comparing the results with the ZeroSpeech 2021 Systems, we observe that our models are closing the gap between spoken and text-based language models.

V. CONCLUSION

This work analyses the importance of discretization in spoken language modeling. We experimentally show that discretization is essential for spoken language modeling, although

[4]One main difference between the two Transformer models is that wav2vec2 uses a Convolutional Positional Embedding instead of the standard Sinusoidal Positional Embedding. However, we did not study the effect of this difference in this paper.

high-quality discrete units are required to obtain good performances. We also show the possibility of learning high-level language properties of a self-supervised speech representation learning model like HuBERT. Finally, we obtain state-of-the-art results on 3 out of 4 metrics of the Zero Resource Speech Challenge 2021 (Track 1 - Speech Only), bridging the gap between speech and text-based systems. Note though that because HuBERT requires a teacher that learns a discrete representation, the overall training of HuBERT is not end-to-end, because the training of the teacher is not (in fact, requires several iterations). Further work is needed to simplify this kind of training loop to learn language directly from speech inputs. Further work is also needed to assess whether the present results can generalize to other languages and datasets.

ACKNOWLEDGMENTS

In this work, ED in his EHESS role was supported by the Agence Nationale pour la Recherche (ANR-17-EURE-0017 Frontcog, ANR-10-IDEX-0001-02 PSL*, ANR-19-P3IA-0001 PRAIRIE 3IA Institute), a grant from CIFAR (Learning in Machines and Brains) and HPC resources from GENCI–IDRIS (Grant 2021-AD011011691R1).

We would like to thank the reviewers for their thoughtful comments on the paper. We also thank Jade Copet, Evgeny Kharitonov, Morgane Riviere, Paden Tomasello, Wei-Ning Hsu, Yossef Mordechay Adi, Abdelrahman Mohamed, Maureen de Seyssel, Marvin Lavechin, Robin Algayres, Xuan-Nga Cao, Nicolas Hamilakis, Hadrien Titeux, Gwendal Virlet, Marianne Metais for helpful discussions on the paper.

REFERENCES

[1] A. Radford, K. Narasimhan *et al.*, "Improving language understanding by generative pre-training," in *arxiv*, 2018.

[2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[3] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[5] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "RoBERTa: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[6] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 12 449–12 460. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf

[7] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9585401

[8] Y. Chung, Y. Zhang, W. Han, C. Chiu, J. Qin, R. Pang, and Y. Wu, "W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," *CoRR*, vol. abs/2108.06209, 2021. [Online]. Available: https://arxiv.org/abs/2108.06209

[9] T. A. Nguyen, M. de Seyssel, P. Rozé, M. Rivière, E. Kharitonov, A. Baevski, E. Dunbar, and E. Dupoux, "The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling," in *Neural Information Processing Systems Workshop on Self-Supervised Learning for Speech and Audio Processing Workshop*, 2020. [Online]. Available: https://arxiv.org/pdf/2011.11588.pdf

[10] E. Dunbar, M. Bernard, N. Hamilakis, T. A. Nguyen, M. De Seyssel, P. Rozé, M. Rivière, E. Kharitonov, and E. Dupoux, "The Zero Resource Speech Challenge 2021: Spoken language modelling," in *Interspeech 2021 - Conference of the International Speech Communication Association*, Brno, Czech Republic, Aug. 2021. [Online]. Available: https://hal.inria.fr/hal-03329301

[11] B. van Niekerk, L. Nortje, M. Baas, and H. Kamper, "Analyzing speaker information in self-supervised models to improve zero-resource speech processing," 08 2021, pp. 1554–1558.

[12] A. Baevski, M. Auli, and A. Mohamed, "Effectiveness of self-supervised pre-training for ASR," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*. IEEE, 2020, pp. 7694–7698. [Online]. Available: https://doi.org/10.1109/ICASSP40776.2020.9054224

[13] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: https://openreview.net/forum?id=rylwJxrYDS

[14] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, G. Kubin and Z. Kacic, Eds. ISCA, 2019, pp. 3465–3469. [Online]. Available: https://doi.org/10.21437/Interspeech.2019-1873

[15] K. Lakhotia, E. Kharitonov, W.-N. Hsu, Y. Adi, A. Polyak, B. Bolte, T.-A. Nguyen, J. Copet, A. Baevski, A. Mohamed, and E. Dupoux, "On Generative Spoken Language Modeling from Raw Audio," *Transactions of the Association for Computational Linguistics*, Feb. 2021. [Online]. Available: https://hal.inria.fr/hal-03329219

[16] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[18] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. [Online]. Available: http://arxiv.org/abs/1807.03748

[19] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing*. Barcelona / Virtual, Spain: IEEE, May 2020, pp. 7414–7418. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02959418

[20] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019. [Online]. Available: https://aclanthology.org/N19-4009

[21] A. Wang and K. Cho, "BERT has a mouth, and it must speak: BERT as a Markov random field language model," in *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 30–36. [Online]. Available: https://aclanthology.org/W19-2304

[22] J. Salazar, D. Liang, T. Q. Nguyen, and K. Kirchhoff, "Masked language model scoring," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association

for Computational Linguistics, Jul. 2020, pp. 2699–2712. [Online]. Available: https://aclanthology.org/2020.acl-main.240

[23] E. Kharitonov, J. Copet, K. Lakhotia, T. A. Nguyen, P. Tomasello, A. Lee, A. Elkahky, W.-N. Hsu, A. Mohamed, E. Dupoux, and Y. Adi, "textless-lib: a library for textless spoken language processing," 2022.

APPENDIX

A. Hyper-parameters of trained models

Table VI shows the hyperparameters of all trained models in the paper.

B. Unit-Phone alignments for CPC Discrete Units

We analyse to what extent the discrete units obtained with different numbers of clusters correlate with the gold phonemes. Using the phoneme alignments of Librispeech available from [9], we collect all unit-phoneme pairs from the utterances of the dev-clean subset and compute the probability of each phoneme given a discrete unit. Figure 3 (top, middle, bottom) shows this unit-phone alignment for discrete units obtained from CPC features with 20, 50 and 500 clusters respectively. The phoneme order is obtained by clustering the rows of the 50-unit model with a hierarchical clustering method.

C. Layer-wise analysis of BERT models on ABX metrics

In addition to performing ABX on input and target features of the BERT models, we also compute the ABX error on the features extracted from hidden Transformer layers of the trained BERT/HuBERT models. Table VII reports the average (within-across) ABX errors on the Librispeech dev-clean subset for all the trained models in the paper.

We see that well-trained models with good language modeling scores (models 1,2,13,15,17,8,9,10; cf. Table III) seem to have very good ABX errors compared to the others. By looking at the best hidden features of each model, we see that models with discrete targets (models 1,2,13) are able to reconstruct hidden features which are better than the targets, while this is not the case for most models with continuous targets (except model 10).

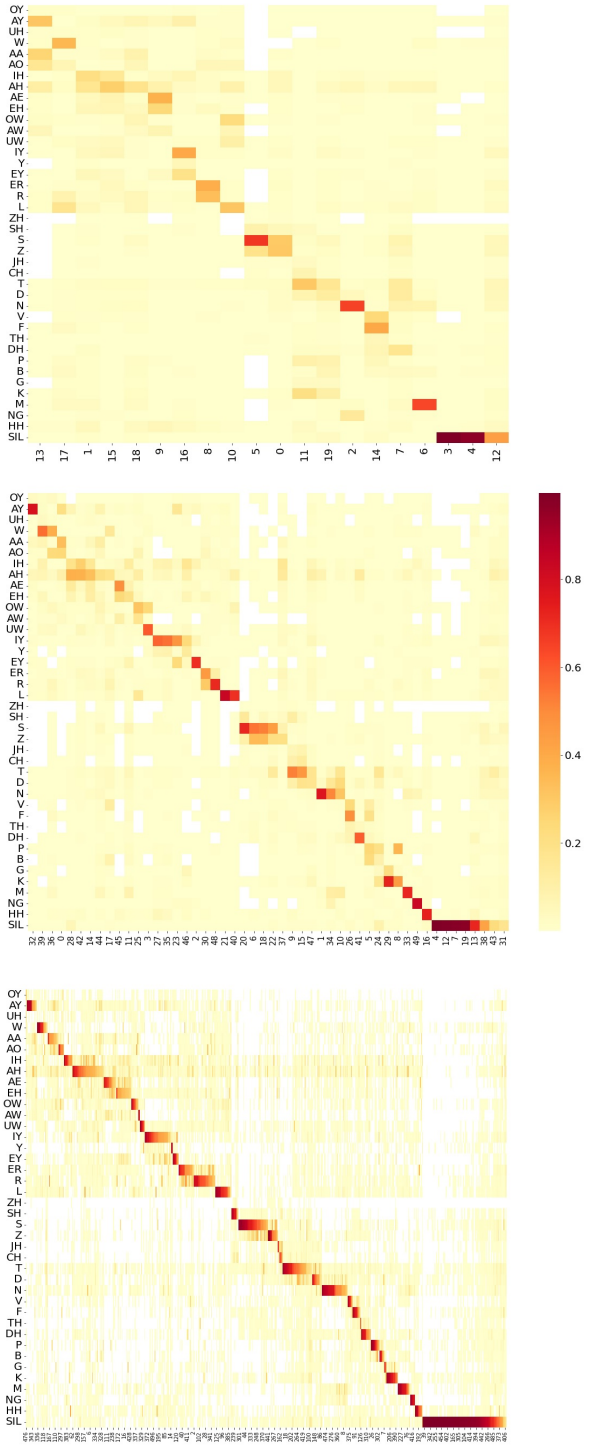

Fig. 3. Probability that each discrete unit belongs to possible phonemes $P\,(phoneme \mid unit)$ for discrete units obtained by clustering CPC features with different numbers of clusters: 20 (top), 50 (middle) and 500 (bottom). Unit-Phoneme alignments are collected on Librispeech dev-clean subset. The phoneme order is obtained by clustering the rows of the 50-unit model with a hierarchical clustering method.

	Training hyperparameters						Inference hyperparameters					
	n units (if discrete)		feat.	masking		num	prob. est.		SIMI librispeech		SIMI synthetic	
id	input	target	stride	length	prob.	updates	M	Δt	layer	pooling	layer	pooling
	BERT Models on CPC-big Features											
1	50	50	10ms	10	0.5	250k	15	5	11	max	1	min
2	50	50	10ms	10	0.5	250k	15	5	5	min	7	mean
11	50	50	10ms	10	0.5	250k	15	5	11	mean	10	min
12	50	50	10ms	10	0.5	250k	15	5	9	mean	1	min
3	-	50	10ms	10	0.5	250k	35	5	6	min	4	max
4	-	50	10ms	10	0.5	250k	35	5	8	mean	11	min
13	-	50	10ms	10	0.5	250k	25	5	10	max	10	min
14	-	50	10ms	10	0.5	250k	35	5	4	max	6	mean
5	-	-	10ms	10	0.5	250k	45	5	8	max	1	mean
15	-	-	10ms	10	0.5	250k	35	5	11	max	11	mean
16	-	-	10ms	10	0.5	250k	45	5	12	mean	12	mean
6	-	-	10ms	10	0.5	250k	35	5	1	mean	12	mean
17	-	-	10ms	10	0.5	250k	25	5	12	min	12	max
18	-	-	10ms	10	0.5	250k	35	5	1	mean	12	mean
7	-	-	10ms	10	0.5	250k	35	5	3	mean	1	mean
8	50	-	10ms	10	0.5	250k	25	5	12	mean	5	min
19	50	-	10ms	10	0.5	250k	25	5	8	mean	4	min
20	50	-	10ms	10	0.5	250k	25	5	8	mean	11	mean
9	50	-	10ms	10	0.5	250k	15	5	12	max	12	min
21	50	-	10ms	10	0.5	250k	15	5	11	mean	7	min
22	50	-	10ms	10	0.5	250k	15	5	4	mean	12	max
10	50	-	10ms	10	0.5	250k	15	5	10	max	12	max
	HuBERT Base Models											
23	-	100	20ms	10	0.65	250k	15	5	2	max	5	min
24	-	500	20ms	10	0.65	400k	15	5	1	mean	9	max
25	-	500	20ms	10	0.65	400k	15	5	10	mean	6	max
	BERT Models on HuBERT Discrete Units											
26	-	500	20ms	10	0.5	250k	10	5	7	mean	8	mean

TABLE VI
HYPERPARAMETERS OF ALL TRAINED MODELS.

id	**input feature**	**target feature**	**loss**	input	layer 3	layer 6	layer 9	layer 12	target
	BERT Models on CPC-big Features								
1	CPC-l2+km50	CPC-l2+km50	NLL-l	7.3	5.90	6.13	5.64	3.87	7.3
2	CPC-l2+km50	CPC-l2+km50	NLL-e	7.3	5.36	6.97	5.40	3.95	7.3
11	CPC-l0+km50	CPC-l2+km50	NLL-e	26.11	15.08	12.16	10.63	7.01	7.3
12	CPC-l4+km50	CPC-l2+km50	NLL-e	22.1	13.57	9.40	10.05	6.06	7.3
3	*CPC-l2*	CPC-l2+km50	NLL-l	3.81	8.87	15.30	14.91	9.10	7.3
4	*CPC-l2*	CPC-l2+km50	NLL-e	3.81	8.47	16.82	21.47	12.84	7.3
13	*CPC-l0*	CPC-l2+km50	NLL-e	15.01	6.57	4.74	4.04	4.07	7.3
14	*CPC-l4*	CPC-l2+km50	NLL-e	9.75	7.03	8.78	10.49	4.98	7.3
5	*CPC-l2*	*CPC-l2*	NCE	3.81	6.47	6.83	6.09	5.98	3.81
15	*CPC-l0*	*CPC-l2*	NCE	15.01	7.45	5.63	4.73	5.93	3.81
16	*CPC-l4*	*CPC-l2*	NCE	9.75	6.82	7.19	5.46	5.50	3.81
6	*CPC-l2*	*CPC-l2*	L1	3.81	7.41	7.81	10.31	12.87	3.81
17	*CPC-l0*	*CPC-l2*	L1	15.01	6.64	4.73	4.71	5.48	3.81
18	*CPC-l4*	*CPC-l2*	L1	9.75	5.89	5.27	4.80	5.38	3.81
7	*CPC-l2*	*CPC-l2*	L2	3.81	6.35	6.58	7.65	9.20	3.81
8	CPC-l2+km50	*CPC-l2*	NCE	7.3	6.49	5.50	6.87	5.37	3.81
19	CPC-l0+km50	*CPC-l2*	NCE	26.11	15.93	12.85	10.55	9.09	3.81
20	CPC-l4+km50	*CPC-l2*	NCE	22.1	12.88	10.17	10.18	8.81	3.81
9	CPC-l2+km50	*CPC-l2*	L1	7.3	5.25	5.71	4.89	10.96	3.81
21	CPC-l0+km50	*CPC-l2*	L1	26.11	13.13	10.87	9.81	15.40	3.81
22	CPC-l4+km50	*CPC-l2*	L1	22.1	11.44	9.91	7.65	11.88	3.81
10	CPC-l2+km50	*CPC-l2*	L2	7.3	5.42	5.09	4.20	3.68	3.81
	HuBERT Base Models								
23	*waveform*	MFCC+km100	NLL-e	-	7.13	4.18	5.03	9.05	27.88
24	*waveform*	H1-l6+km500	NLL-e	-	6.83	4.71	4.42	3.53	6.97
25	*waveform*	H2-l12+km500	NLL-e	-	6.66	4.43	4.48	3.78	6.26
	BERT Models on HuBERT Discrete Units								
26	H2-l12+km500	H2-l12+km500	NLL-e	6.26	5.32	6.51	6.74	4.43	6.26

TABLE VII
AVERAGE (WITHIN AND ACROSS) DEV-CLEAN ABX ERROR OF INPUT FEATURES, TARGET FEATURES AND FEATURES FROM DIFFERENT HIDDEN LAYERS OF TRANSFORMER MODEL. CPC-LX STANDS FOR LAYER X OF CPC-BIG, HJ-LX STANDS FOR LAYER X OF HUBERT J'TH ITERATION.