The Zero Resource Speech Benchmark 2021: Metrics and baselines for unsupervised spoken language modeling

Tu Anh Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Evgeny Kharitonov, Alexei Baevski, Ewan Dunbar, Emmanuel Dupoux

The Zero Resource Speech Benchmark 2021: Metrics and baselines for unsupervised spoken language modeling

Tu Anh Nguyen*
Facebook AI Research & EHESS,
ENS-PSL Univ., CNRS, INRIA, France
nguyentuanh208@gmail.com

Maureen de Seyssel*
EHESS, ENS-PSL Univ., CNRS, INRIA
& U. Paris, France
maureen.deseyssel@gmail.com

Patricia Rozé
ENS-PSL Univ., CNRS
France
patricia.roze@ens.fr

Morgane Rivière
Facebook AI Research
France
mriviere@fb.com

Evgeny Kharitonov
Facebook AI Research
France
kharitonov@fb.com

Alexei Baevski
Facebook AI Research
France
abaevski@fb.com

Ewan Dunbar†
U. Paris Diderot, France
& U. Toronto, Canada
ewan.dunbar@utoronto.ca

Emmanuel Dupoux†
Facebook AI Research & EHESS,
ENS-PSL, CNRS, INRIA, France
emmanuel.dupoux@gmail.com

Abstract

We introduce a new unsupervised task, spoken language modeling: the learning of linguistic representations from raw audio signals without any labels, along with the Zero Resource Speech Benchmark 2021: a suite of 4 black-box, zero-shot metrics probing for the quality of the learned models at 4 linguistic levels: phonetics, lexicon, syntax and semantics. We present the results and analyses of a composite baseline made of the concatenation of three unsupervised systems: self-supervised contrastive representation learning (CPC), clustering (k-means) and language modeling (LSTM or BERT). The language models learn on the basis of the pseudo-text derived from clustering the learned representations. This simple pipeline shows better than chance performance on all four metrics, demonstrating the feasibility of spoken language modeling from raw speech. It also yields worse performance compared to text-based 'topline' systems trained on the same data, delineating the space to be explored by more sophisticated end-to-end models.

1 Introduction

In recent work, self-supervised techniques from vision and NLP have been applied to large datasets of raw audio, giving rise to very effective methods of pretraining for downstream ASR tasks, particularly in the low resource scenario (Schneider et al., 2019; Baevski et al., 2019; Chung and Glass, 2019; Baevski et al., 2020b; Rivière et al., 2020; Kawakami et al., 2020; Wang et al., 2020). The approaches based on transformers and masking objectives, strikingly similar to the models used to train language models, are especially intriguing. The fact that these approaches yield excellent ASR performance (less than 10% WER) with as little as 10 minutes of labels plus a language model (LM), or with 10 hours of labels but no LM (Baevski et al., 2020b), suggests that these systems may actually go beyond acoustic modeling, learning their own LM from raw audio. Such work therefore connects with

*Equal contribution as first authors. † Equal contributions as last authors.

Preprint from the Self-Supervised Learning for Speech and Audio Processing Workshop @ NeurIPS 2020.

Table 1: **Summary description of the four Zero Resouce Benchmark 2021 metrics**. The metrics in light blue use a pseudo-distance d between embeddings (d_h being from human judgments), the metrics in light orange use a pseudo-probability p computed over the entire input sequence.

Linguistic level	Metrics	Dataset	Task	Example
acoustic-phonetic	ABX	Libri-light	$d(a,x) < d(b,x)$? $a \in A, b \in B$, $x \neq a \in A$	within-speaker: $(\text{apa}_{s_1}, \text{aba}_{s_1}, \text{apa}_{s_1})$ across-speaker: $(\text{apa}_{s_1}, \text{aba}_{s_1}, \text{apa}_{s_2})$
lexicon	spot-the-word	sWUGGY	$p(a) > p(b)$?	(brick, blick) (squalled, squilled)
lexical semantics	similarity judgement	sSIMI	$d(a,b) \propto d_h(a,b)$?	(abduct, kidnap) : 8.63 (abduct, tap) : 0.5
syntax	acceptability judgment	sBLIMP	$p(a) > p(b)$?	(dogs eat meat, dogs eats meat) (the boy can't help himself, the boy can't help herself)

research into the *zero resource* setting, which aims at learning linguistic representations from scratch for language with little or no textual resources. However, up to now, there exists no established benchmark to analyse the representations learned by such models beyond the acoustic/phonetic level.

Typically, language models trained from text are evaluated using scores like perplexity. Unfortunately, this simple approach cannot be used here, since perplexity scores computed from learned discrete units vary according to granularity, making model comparison impossible. This is why we chose to follow a black-box NLP strategy: our metrics do require expert linguistic labels for the dev and test sets, but are *zero-shot* in that they do not require training a classifier, they use *simple tasks* enabling direct human/machine comparison, and they give *interpretable* scores at each linguistic level. As seen in Table 1, they can be divided into two types: distance-based and probability-based metrics. Distance-based metrics require models to provide a pseudo-distance computed over pairs of embeddings. The ABX score (Schatz et al., 2013), already used for the evaluation of *acoustic/phonetic* representations, falls in this category and provides a measure of how well separated phonetic categories are in a given embedding space. Here, we use the ABX score developed in Libri-light (Kahn et al., 2020). Distance-based methods can also be used to evaluate the *semantic* representation of words, by computing the correlation between these distances and human semantic similarity judgements (see Schnabel et al., 2015; Faruqui et al., 2016). Chung and Glass (2018) adapted this metric to speech, which we compiled into our sSIMI dataset. Probability-based metrics require models to compute a pseudo-probability for a given test input (non-normalized non-negative number for a given input waveform). The pseudo-probabilities are computed over pairs of inputs, one of which is acceptable in the tested language and the other not. Such methods have been used in NLP to evaluate the *syntactic* abilities of language models, by comparing the probabilities of grammatical versus ungrammatical sentences(Warstadt et al., 2019), and we built the sBLIMP dataset upon this work. Finally, in our sWUGGY dataset, we extend this logic to the *lexical* level by comparing the pseudo-probability associated to words and nonwords. The four metrics are presented in more details in Section 3.2.

Next, we apply these metrics to a simple baseline system (Section 3.3), built on contrastive pretraining (Contrastive Predictive Coding, CPC, van den Oord et al., 2018; Rivière et al., 2020), followed by k-means clustering, which we use to decode a speech dataset (LibriSpeech, Panayotov et al., 2015) into pseudo-text. This pseudo-text is used to train a language model varying in compute budget: an LSTM (smaller budget) or BERT (larger budget) model. We show (Section 4) that such simple baseline models give better than chance performance on all 4 metrics, demonstrating that it has learned representations at the four corresponding linguistic levels. However, comparison with a text-based BERT topline system trained on the phonetic transcription of the same training data shows that the speech input raises challenges for the LM component of the model that need to be addressed in further work. Datasets and baselines will be open sourced to encourage bridging the gap between speech and text-based systems.

2 Related work

Zero Resource Speech Challenge Series. Previous work (Versteegh et al., 2016; Dunbar et al., 2017, 2019, 2020) has focused on establishing benchmarks for unsupervised learning of an entire dialogue system, but has so far remained at a rather low level (acoustic, lexical). Acoustic modeling

has used two metrics: ABX, a distance-based metric to be discussed later, and opinion scores on TTS output (whereby the discovered units are used to resynthesize speech). As for the lexical level, past work has focused on using the NLP metrics developed for word segmentation (Ludusan et al., 2014). However, these metrics assume that the models should discover words explicitly. The success of character-based language models suggests that it is possible to learn high-level linguistic concepts without explicitly segmenting words (see Hahn and Baroni, 2019).

Black box NLP. Among the variety of black-box linguistic tasks, psycholinguistically-inspired ones enable direct comparison of models and humans. Grammaticality judgments for recurrent networks have been investigated since Allen and Seidenberg (1999), who use closely matched pairs of sentences to investigate grammatical correctness. This approach has recently been adopted to assess the abilities of RNNs, and LSTMs in particular, in capturing syntactic structures. For instance, Linzen et al. (2016) and Gulordava et al. (2018) use word probes in minimally different pairs of English sentences to study number agreement. To discriminate grammatical sentences from ungrammatical ones, they retrieve the probabilities of the possible morphological forms of a target word, given the probability of the previous words in the sentence. Practically, in the sentence "the boy is sleeping", they assume the network has detected number agreement if $\mathbf{P}(w = is) > \mathbf{P}(w = are)$. This methodology has also been adapted by Goldberg (2019) to models trained with a masked language-modeling objective. Similarly, Ravfogel et al. (2018) use word probes to examine whether LSTMs understand Basque agreement and Godais et al. (2017) to test the lexical level in character-based LM.

3 Methods

3.1 Training set

We used as a training set the LibriSpeech 960h dataset (Panayotov et al., 2015). We also included in this work the clean-6k version of the Libri-light dataset (Kahn et al., 2020) which is a huge collection of speech for unsupervised learning. A phonetic transcription of the LibriSpeech dataset was also employed. To obtain this, we used the original LibriSpeech lexicon, as well as the G2P-seq2seq toolkit[2] to generate the phonetic transcriptions of words lacking from the lexicon. We generated a forced-alignment version of Librispeech using the abkhazia library[3]. This enabled us to provide comparative text-based topline systems along with the speech baseline.

3.2 Metrics

We set up four metrics with their accompanying datasets, to evaluate the sLMs at four levels: phonetic (the Libri-light ABX metrics), lexical (the sWUGGY spot-the-word metrics), syntactic (the sBLIMP acceptability metrics) and semantic (the sSIMI similarity metric). The 4 datasets are composed of speech sounds extracted from LibriSpeech (sSIMI), or synthetic stimuli constructed with the Google API[4] using 4 different voices, two males and two females (sWUGGY, sBLIMP, sSIMI)[5]. When synthetic, the stimuli were subsequently force-aligned to retrieve the phonetic boundaries. The datasets containing words or sentences were filtered to only contain the LibriSpeech vocabulary, and are split into dev and test sets.

Phonetics: Libri-light ABX metrics. The ABX metric consists in computing, for a given contrast between two speech categories A and B (e.g., the contrast between triphones 'aba' and 'apa'), the probability that two sounds belonging to the same category are closer to one another than two sounds that belong to different categories. Formally, we compute an asymmetric score, with a and x, different tokens belonging to category A (of cardinality n_A) and b belonging to B (n_B), respectively:

$$\hat{e}(A,B) := \frac{1}{n_A(n_A-1)n_B} \sum_{\substack{a,x \in A \\ x \neq a}} \sum_{b \in B} \left[\mathbb{1}_{d(b,x)<d(a,x)} + \frac{1}{2}\mathbb{1}_{d(b,x)=d(a,x)} \right] \quad (1)$$

[2]https://github.com/cmusphinx/g2p-seq2seq

[3]https://github.com/bootphon/abkhazia

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

[5]We use WaveNet voices A, C, D and F. All dev set stimuli are synthesised in all four voices. Stimuli in the sSIMI and sBLIMP test sets are split evenly among the four different voices, and sWUGGY uses all four for each test set stimulus.

The score is symmetrized and aggregated across all minimal pairs of triphones like 'aba', 'apa', where the change only occurs in the middle phoneme. This score can be computed within speaker (in which case, all stimuli a, b and x are uttered by the same speaker) or across speaker (a and b are from the same speaker, and x from a different speaker). This score requires a pseudo-distance between acoustic tokens computed by averaging along a dynamic time warping path a framewise distance (KL or angular distance). This metric is agnostic to the dimensionality of the embeddings, can work with discrete or continuous codes, and has been used to compare ASR speech features (Schatz, 2016). Here, we run this metric on the pre-existing Libri-light dev and test sets, which has been already used to evaluate several self-supervised models (Kahn et al., 2020; Rivière et al., 2020).

Lexicon: sWUGGY spot-the-word metrics. We built on Godais et al. (2017) which used the 'spot-the-word' task. In this task, networks are presented with a pair of items, an existing word and a matching nonword, and are evaluated on their capacity to attribute a higher probability to the existing word. The spot-the-word metric corresponds to the average accuracy of classifying the words and nonwords correctly across each pair.

The nonwords are produced with WUGGY(Keuleers and Brysbaert, 2010), which generates for a given word, a list of candidate nonwords best matched in phonotactics and syllabic structure. Because we were aiming at speech stimuli, we needed additional constraints to ensure that (i) the audio synthesis of the pairs would be of good quality, and (ii) that the pairs would have matching unigram and bigram scores relative to their phonemes. On a sample of 100 word/nonword pairs, and with feedback from a native English speaker informant, we designed a synthesis-quality rule. The rule consists of testing whether the original phonetic transcription matches the output of a back-to-back phoneme-to-grapheme (p2g) and grapheme-to-phoneme encoding (g2p).[6] Only pairs where both the words and nonwords passed this test were kept. We added additional constraints using a stochastic sampler to also match unigram and bigram phoneme frequencies (see Supplementary Material A). The final sWUGGY test and development sets consists of 20,000 and 5,000 pairs respectively, with the existing words being part of the LibriSpeech train vocabulary. We also prepared additional OOV-sWUGGY test and development sets consisting of 20,000 and 5,000 pairs respectively, with existing words which do not appear in the LibriSpeech training set.

The spot-the-word accuracy is the average of the indicator function $1_{PP(word_k)>PP(nonword_k)}$ over the set of pairs $(word_k, nonword_k)$, where *PP* is a pseudo-probability (a possibly non-normalized non-negative number) assigned to each input file by the model.

Syntax: sBLIMP acceptability metrics. This part of the benchmark is adapted from BLIMP (Warstadt et al., 2019), a dataset of linguistic minimal sentence pairs of matched grammatical and ungrammatical sentences. Similarly to the preceding test, the task is to decide which of the two members of the pair is grammatical based on the probability of the sentence. We adapted the code used to generate the BLIMP dataset (Warstadt et al., 2019) in order to create sBLIMP, specifically tailored for speech purposes. In BLIMP, sentences are divided into twelve broad categories of syntactic paradigms. These categories are themselves divided into 68 specific paradigms containing 1000 sentence pairs each, automatically generated using an expert hand-crafted grammar (this includes an additional subcategory which was added to the code subsequent to Warstadt et al. (2019).

To make this dataset 'speech-ready,' we discarded five subcategories and slightly modified the grammar for nine additional subcategories in order to ensure sentences had appropriate prosodic contours. We also removed from the vocabulary all words absent from the LibriSpeech train set (Panayotov et al., 2015), as well as compound words and homophones that could cause further comprehension issues once synthesised. 5000 sentence pairs were then generated for each of the 63 remaining subcategories. We sampled sentence pairs from the generated pool to create a development and a test set, ensuring that the larger linguistic categories were sampled so as to balance the n-gram language model scores (see Supplementary Material A). The test and development sets contain 63,000 and 6,300 sentence pairs respectively, with no overlap in sentence pairs. Stimuli were then synthesized and force-aligned as described at the beginning of the section.

Similar to the spot-the-word metric, the acceptability judgment metric requires a pseudo-probability for each given input file. The sentence acceptability accuracy is reported similarly to the spot-the-word accuracy with the pairs of grammatical and ungrammatical sentences in the sBLIMP dataset.

[6] We used the G2P-seq2seq toolkit.

Lexical semantics: sSIMI similarity metrics. Here, the task is to compute the similarity of the representations of pairs of words and compare it to human similarity judgements. Based on previous work (Chung and Glass, 2018), we used a set of 13 existing semantic similarity and relatedness tests to construct our similarity benchmark. The similarity-based datasets include WordSim-353 (Yang and Powers, 2006), WordSim-353-SIM (Agirre et al., 2009), mc-30 (Miller and Charles, 1991), rg-65 Rubenstein and Goodenough (1965), Rare-Word (or rw) (Luong et al., 2013), simLex999 (Hill et al., 2015), simverb-3500 (Gerz et al., 2016), verb-143 (Baker et al., 2014) , YP-130 Yang and Powers (2006) and the relatedness-based datasets include MEN (Bruni et al., 2012), Wordsim-353-REL (Agirre et al., 2009), mturk-287 (Radinsky et al., 2011), mturk-771 (Halawi et al., 2012). All scores were normalised on a 0-10 scale, and pairs within the same dataset containing the same pair of words but in the opposite order were averaged. Pairs containing a word not in the LibriSpeech train set Panayotov et al. (2015) were discarded.

We selected as a development set the mturk-771 dataset, which was, in preliminary study using character- and word-based LMs, both highly correlated with all other datasets and was large enough to be used as a development set. It was also ensured that no pair from the development set was present in any of the test sets. All other twelve datasets were used as test sets. We then created two subsets of audio files, one synthetic, one natural. For the first, we followed the synthesis and forced alignment procedures described at the beginning of the section. For the second, we retrieved the audio extracts from LibriSpeech corresponding to each word, following the process presented in (Chung and Glass, 2018). The natural subset is therefore smaller than its synthesized counterpart as we had to discard pairs from the test and dev sets which were not present in the LibriSpeech test and dev sets respectively. However, in this natural subset, each word may appear in multiple tokens, providing phonetic diversity; duplicated scores are averaged in the analysis step. The synthesised subset is composed of 9744 and 705 word pairs for the test and dev sets respectively, and the LibriSpeech subset is composed of 3753 and 309 pairs for the test and dev sets.

The semantic similarity score is reported as the Spearman's rank correlation coefficient ρ between the semantic distance scores given by the model and the true human scores in the dataset. Note that in this work all the semantic similarity scores are multiplied by 100 for clarity.

3.3 Models

Baseline models. Our baseline models are a composite of three components: an acoustic model (CPC), a clustering module (k-means) and a language model (LSTM or BERT) varying in size.

The acoustic model is built upon Contrastive Predictive Coding (CPC, van den Oord et al. (2018)), where the representation of the audio is learned by predicting the future through an autoregressive model. In more detail, given an input signal $\mathbf{x}$, the CPC model embeds $\mathbf{x}$ to a sequence of embeddings $\mathbf{z} = (z_1, \dots, z_T)$ at a given rate through a non-linear encoder g_{enc}. At each time step t, the autoregressive model g_{ar} takes as input the available embeddings $z_1, \dots, z_t$ and produces a context latent representation $c_t = g_{\text{ar}}(z_1, \dots, z_t)$. Given the context c_t, the CPC model tries to predict the K next future embeddings $\{z_{t+k}\}_{1 \leq k \leq K}$ by minimizing the following constrastive loss:

$$\mathcal{L}_t = -\frac{1}{K} \sum_{k=1}^{K} \log \left[\frac{\exp\left(z_{t+k}^\top W_k c_t\right)}{\sum_{\tilde{z} \in \mathcal{N}_t} \exp\left(\tilde{z}^\top W_k c_t\right)} \right] \tag{2}$$

where $\mathcal{N}_t$ is a random subset of negative embedding samples, and W_k is a linear classifier used to predict the future k-step observation. We used a PyTorch implementation of CPC[7] (Rivière et al., 2020), which is a modified version of the CPC model that stabilizes the CPC training by replacing batch normalization with a channel-wise normalization and improves the CPC model by replacing the linear classifier W_k in equation (2) with a 1-layer Transformer network (Vaswani et al., 2017). The encoder g_{enc} is a 5-layer 1D-convolutional network with kernel sizes of 10,8,4,4,4 and stride sizes of 5,4,2,2,2 respectively, resulting in a downsampling factor of 160, meaning that the embeddings have a rate of 100Hz. The autoregressive model g_{ar} is a multi-layer LSTM network, with the same hidden dimension as the encoder. For this baseline, we trained two different versions of CPC: CPC-small and CPC-big. Details are given in Table 2.

After training the CPC model, we then train a k-means clustering module on the outputs of either the final layer or a hidden layer of the autoregressive model. The clustering is done on the collection of

[7] https://github.com/facebookresearch/CPC_audio

Table 2: **Characteristics of the baseline acoustic CPC models**. We took the last LSTM layer of CPC-small and the second LSTM hidden layer of CPC-big as inputs to the clustering as they give the best ABX scores (Supplementary Table S1).

Model	CPC configuration		Training data	Input to kmeans
	autoregressive	hidden units		
CPC-small	2-layer LSTM	256	LibriSpeech clean-100	LSTM level 2
CPC-big	4-layer LSTM	512	Libri-light clean-6k	LSTM level 2

Table 3: **Characteristics of the baseline LMs.** L refers to the number of hidden layers; ED, HD and FFD refer to the dimension of the embedding layer, hidden layer, and feed-forward output layer respectively; H refers to the number of attention heads in the BERT case.

Model	Architecture					nb parameters	Train data	Compute Budget
	L	ED	HD	FFD	H			
BERT	12	768	768	3072	12	90M	LS960	48h - 32 GPUs
BERT-small	8	512	512	2048	8	28M	LS960	60h - 1GPU
LSTM	3	200	1024	200	-	22M	LS960	60h- 1GPU

all the output features at every time step of all the audio files in a given training set. After training the k-means clustering, each feature is then assigned to a cluster, and each audio file can then be discretized to a sequence of discrete units corresponding to the assigned clusters. The k-means training was done on the subset of LibriSpeech containing 100 hours of clean speech.

Finally, with the discretized version of the audio files, we train language models on the discretized units. We establish two 'low budget' and two 'high budget' baselines, based on the number of parameters and the compute resources necessary to train them. The high budget used a BERT-based architecture (Devlin et al., 2019) trained either on CPC-small or CPC-big plus k-means-50 pretrained units. The low budget architectures were a two-layer LSTM and a small BERT architecture (see Table 3 for details); they both used the units from the CPC-big pretraining. Following Baevski et al. (2020a), we trained the BERT models with only the masked token prediction objective. We also followed Baevski et al. (2020a) by masking a span of tokens in the input sequence instead of a single token (otherwise the prediction would be trivial to the model as discretized units tend to replicate). 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). All models were trained on LibriSpeech 960h. The BERT models were trained with a total batch size of 524k tokens, and the LSTM model was trained with a total batch size of 163k tokens. The learning rate was warmed up to a peak value of 1×10^{-5}. All the implementation was done via fairseq (Ott et al., 2019).

The Topline models. For topline comparison, we trained a BERT model on force-aligned phonemes using the gold transcription of the LibriSpeech dataset. We also employed the span masking similarly to the baseline model. In addition to the BERT trained on forced alignments, we also included a BERT model trained on the gold phonetic transcription of the LibriSpeech dataset, with the difference that we only mask one token instead of a span of tokens. For an absolute topline comparison, we used the pretrained RoBERTa large model (Liu et al., 2019), which was trained on 50K subword units on a huge dataset of total 160GB, 3000 times bigger than the transcription of the LibriSpeech 960h dataset.

4 Results

4.1 Libri-light ABX

Computing distances. We used the average angular distance (arccos of the normalized dot product) of the representations along the DTW-realigned path, as used by default in previous challenges (Versteegh et al., 2016; Dunbar et al., 2017, 2019). For our baseline models, we computed the ABX scores over one-hot representations of discretized units of the audio files.

Results. We first ran experiments varying the number of clusters. As seen in Supplementary Table S2, too few or too many clusters gives rise to worse ABX performance, with a sweet spot at 50 clusters, which is the number we retain for the remainder of the paper. In Table 4, we present the result of the ABX for our two models (CPC-small and CPC-big), before and after clustering. One can see that the CPC-big model yields better performance than the CPC-small model (we retain the big

Table 4: **Within and Across Speaker ABX error** (lower is better) on Libri-light dev-clean and -other for two unsupervised models, before and after clustering (1-hot representations).

	within		across	
Embedding	dev-clean	dev-other	dev-clean	dev-other
MFCC	10.95	13.55	20.94	29.4
CPC-small	6.24	8.48	8.17	13.55
+kmeans-50	10.26	14.24	14.17	21.26
CPC-big	3.41	4.85	4.18	7.64
+kmeans-50	6.38	10.22	8.26	14.86

model for the rest of the experiments), and the clustering step yields an increase in error of between 60-100%. Still, the performances are better than for an MFCC representation, with a much more compact code.

4.2 sWUGGY spot-the-word

Computing the pseudo-probability. Given an audio file x, we first discretize x into a sequence of discretized units $q_1...q_T$. Then, following Salazar et al. (2020), we propose the following pseudo-probability score for our BERT models trained with a span-masked token prediction objective:

$$\text{span-PP}_{M_d,\Delta t}(q_1..q_T) = \prod_{\substack{i=1+j\Delta t \\ \lfloor (T-1)/\Delta t \rfloor \geq j \geq 0}} P(q_i..q_{i+M_d}|q_1..q_{i-1}q_{i+M_d+1}..q_T),$$

where M_d is a chosen decoding span size, and Δt is a temporal sliding size. For the LSTM model, we computed the probability of the discretized sequence with the classic left-to-right scoring style obtained by the chain rule: $P(q_1..q_T) = \prod_{i=1}^{T} P(q_i|q_1..q_{i-1})$.

Results. We determined the optimal masking (Supplementary Table S3) to be $\Delta t = 5$ and $M_d = 15$. We kept this setting for all other experiments involving pseudo-probabilities. Table 5 presents the average of the four baseline systems and in Figure S1, the detailed performances of the baseline compared to n-gram controls and toplines. The performance of all four baselines is consistently better than chance and n-gram controls.

4.3 sBLIMP acceptability

Computing the pseudo-probability. We computed the pseudo-probability as in Section 4.2.

Results. The aggregate results are shown in Table 5 and the detailed ones on the best system in Table S4. The results of this test, while above chance are considerably lower than the text-based toplines.

4.4 sSIMI semantic similarity

Computing the distance. We computed the semantic distance between two audio files x and y as the similarity between the two corresponding discretized sequences $q_1^x...q_T^x$ and $q_1^y...q_S^y$. To obtain this, we extracted outputs from a hidden layer of the LM to the two discretized sequences, aggregating them with a pooling function to produce a fixed-length representation vector for each sequence, and computed the cosine similarity between the two representation vectors:

$$d_{SEM}(x,y) = sim\left(f_{pool}\left(h^{(i)}(q_1^x...q_T^x)\right), f_{pool}\left(h^{(i)}(q_1^y...q_S^y)\right)\right),$$

where f_{pool} is the pooling function and $h^{(i)}(\cdot)$ is the output of the i^{th} hidden layer of the LM.

As each word consists of possibly several voices, we averaged the similarity distance over pairs of the same voice for the synthetic subset, and all possible pairs for the LibriSpeech subset.

Results. For each model, we chose the pooling function and the hidden level that give the best score on the dev set, and computed the score on the corresponding test set. The aggregate results are in Table 5, and a detailed layer-by-layer analysis in Table S5. The scores for semantic similarity are overall modest, compared to BERT systems trained on larger units (BPE). However, one can observe that the best layers for semantic similarity occur towards the first third of the transformer, and that max pooling seems to be best. This contrasts with the best layers for acoustic similarity (as indexed by ABX), which occur at the extremities.

4.5 Model comparison

The overall results are in Table 5. They show that the four baseline models are above chance in the four tasks, even low budget ones, although there is substantial variation between tasks. While task at the lexical level is substantially above chance, the syntactic and semantic tasks show room for improvement compared to text-based toplines trained on similar amounts of data.

5 Discussion

Table 5: Overall performance of our baseline and topline models on dev and test sets on our four zero-shot metrics. For baseline models, the k-means training (k=50) was performed on LibriSpeech clean-100h, and the LSTM/BERT models was trained on discretized units of LibriSpeech 960h. For topline comparisons, we included a BERT model trained on the forced aligned frames of LibriSpeech 960h, a BERT model trained on the gold phonetic transcription of LibriSpeech 960h, and a RoBERTa large model pretrained on a text dataset 3000 times bigger than the transcription of LibriSpeech 960h.

		ABX within		ABX across		sWUGGY	sBLIMP	sSIMI	
System	Set	clean	other	clean	other			synth.	libri.
Low budget baseline systems									
CPC-big+km50+BERT-small	dev	6.38	10.22	8.26	14.86	65.81	52.91	3.88	5.56
	test	6.71	10.62	8.41	15.06	65.94	53.02	3.02	0.06
CPC-big+km50+LSTM	dev	6.38	10.22	8.26	14.86	66.13	53.32	4.42	7.56
	test	6.71	10.62	8.41	15.06	66.22	52.89	7.35	6.66
High budget baseline systems									
CPC-small+km50+BERT	dev	10.26	14.24	14.17	21.26	70.69	54.26	2.99	6.68
	test	10.07	14.71	13.45	22.42	70.50	54.61	8.96	-1.55
CPC-big+km50+BERT	dev	6.38	10.22	8.26	14.86	75.56	56.14	6.25	8.72
	test	6.71	10.62	8.41	15.06	75.51	56.16	5.17	1.75
Topline systems									
Forced align BERT	dev	0.00	0.00	0.00	0.00	92.19	63.72	7.92	4.54
	test	0.00	0.00	0.00	0.00	91.88	63.16	8.52	2.41
Phone BERT	dev	-	-	-	-	97.90	66.78	9.86	16.11
	test	-	-	-	-	97.67	66.91	12.23	20.16
RoBERTa large	dev	-	-	-	-	96.58	81.56	32.28	28.96
	test	-	-	-	-	96.25	82.11	33.16	27.82

We introduced the new Zero Resource Speech Benchmark 2021 for spoken language models. It is composed of 4 zero-shot tests probing 4 linguistic levels: acoustic, lexical, syntactic and semantic. We showed that a simple CPC+clustering+LM trained on LibriSpeech can perform above chance on all of these tests, outperforming n-gram models, while being worse than text-based models trained on the same data. This shows both that the spoken LM task is feasible, and that there is room for improvement.

Obvious directions for research include improving the representation learning component, the clustering methods, and the transformer, which have not been particularly tuned for this benchmark. There are also end-to-end models like wav2vec (Baevski et al., 2020b) and other masking systems (Wang et al., 2020) that could be tried in this context. The performance gap between the RoBERTa large system and our toplines trained on LibriSpeech suggest that much is to be gained by increasing the size of the training set, which can be obtained by large unlabelled audio datasets like LibriVox. Finally, even though this benchmark is intended for developing speech technology for low resource languages, significant resources are still required to construct the test sets and metrics (phonetic dictionary, aligned speech, grammar, TTS or trained speakers to make the stimuli). More work is needed to reduce this footprint and scale up this benchmark to languages other than English.

Broader Impact

The metrics developed here may help improve interpretability of unsupervised systems. Research within the Zero Resource setting may help for developing speech technology for low resourced languages, or for languages with no textual resources, which cannot be addressed in the supervised setting. Even for high resource languages, learning a language model from raw speech would help address dialect variation, including minorities, making speech technology more inclusive. Broadening the reach of speech technology might be used to increase the economic dominance of already-large actors if developed with proprietary resources. To minimize this, we engage the community through an open source benchmark.

Acknowledgments

The work for MS, PR and for EDupoux and TAN in their EHESS role was supported by the Agence Nationale de la Recherche (ANR-17-EURE-0017 Frontcog, ANR-10-IDEX-0001-02 PSL*, ANR-19-P3IA-0001 PRAIRIE 3IA Institute) and grants from CIFAR (Learning in Minds and Brains) and Facebook AI Research (Research Grant). The work for EDunbar was supported by a Google Faculty Research Award and by the Agence Nationale de la Recherche (ANR-17-CE28-0009 GEOMPHON, ANR-18-IDEX-0001 U de Paris, ANR-10-LABX-0083 EFL).

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches.

Joseph Allen and Mark S Seidenberg. 1999. The emergence of grammaticality in connectionist networks. *The emergence of language*, pages 115–151.

Alexei Baevski, Michael Auli, and Abdelrahman Mohamed. 2019. Effectiveness of self-supervised pre-training for speech recognition. *arXiv preprint arXiv:1911.03912*.

Alexei Baevski, Steffen Schneider, and Michael Auli. 2020a. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020b. wav2vec 2.0: A framework for self-supervised learning of speech representations. *arXiv preprint arXiv:2006.11477*.

Simon Baker, Roi Reichart, and Anna Korhonen. 2014. An unsupervised model for instance level subcategorization acquisition. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 278–289.

Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. 2012. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145.

Yu-An Chung and James Glass. 2018. Speech2vec: A sequence-to-sequence framework for learning word embeddings from speech. *arXiv preprint arXiv:1803.08976*.

Yu-An Chung and James Glass. 2019. Generative pre-training for speech with autoregressive predictive coding. *arXiv preprint arXiv:1910.12607*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*.

Ewan Dunbar, Robin Algayres, Julien Karadayi, Mathieu Bernard, Juan Benjumea, Xuan-Nga Cao, Lucie Miskic, Charlotte Dugrain, Lucas Ondel, Alan W. Black, Laurent Besacier, Sakriani Sakti, and Emmanuel Dupoux. 2019. The zero resource speech challenge 2019: Tts without t.

Ewan Dunbar, Xuan Nga Cao, Juan Benjumea, Julien Karadayi, Mathieu Bernard, Laurent Besacier, Xavier Anguera, and Emmanuel Dupoux. 2017. The zero resource speech challenge 2017.

Ewan Dunbar, Julien Karadayi, Mathieu Bernard, Xuan-Nga Cao, Robin Algayres, Lucas Ondel, Laurent Besacier, Sakti Sakriani, and Emmanuel Dupoux. 2020. The zero resource speech challenge 2020: Discovering discrete subword and word units. In *INTERSPEECH, perception;bootstrapping/modeling;clustering/bootphon*.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*.

Daniela Gerz, Ivan Vulić, Felix Hill, Roi Reichart, and Anna Korhonen. 2016. Simverb-3500: A large-scale evaluation set of verb similarity. *arXiv preprint arXiv:1608.00869*.

Gaël Godais, Tal Linzen, and Emmanuel Dupoux. 2017. Comparing character-level neural language models using a lexical decision task. pages 125–130.

Yoav Goldberg. 2019. Assessing bert's syntactic abilities. *arXiv preprint 1901.05287*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically.

Michael Hahn and Marco Baroni. 2019. Tabula nearly rasa: Probing the linguistic knowledge of character-level neural language models trained on unsegmented text. *Transactions of the Association for Computational Linguistics (Accepted)*.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1406–1414.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

J. Kahn, M. Riviere, W. Zheng, E. Kharitonov, Q. Xu, P.E. Mazare, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, and et al. 2020. Libri-light: A benchmark for asr with limited or no supervision. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. van den Oord. 2020. Learning robust and multilingual speech representations.

Emmanuel Keuleers and Marc Brysbaert. 2010. Wuggy: A multilingual pseudoword generator. *Behavior research methods*, 42(3):627–633.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *TACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Bogdan Ludusan, Maarten Versteegh, Aren Jansen, Guillaume Gravier, Xuan-Nga Cao, Mark Johnson, and Emmanuel Dupoux. 2014. Bridging the gap between speech technology and natural language processing: an evaluation toolbox for term discovery systems. In *Proceedings of LREC*, pages 560–567.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 104–113.

George A Miller and Walter G Charles. 1991. Contextual correlates of semantic similarity. *Language and cognitive processes*, 6(1):1–28.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th international conference on World wide web*, pages 337–346.

Shauli Ravfogel, Francis M Tyers, and Yoav Goldberg. 2018. Can LSTM learn to capture agreement? the case of basque. *arXiv preprint 1809.04022*.

Morgane Rivière, Armand Joulin, Pierre-Emmanuel Mazaré, and Emmanuel Dupoux. 2020. Unsupervised pretraining transfers well across languages.

Herbert Rubenstein and John B Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, Online. Association for Computational Linguistics.

T. Schatz, V. Peddinti, F. Bach, A. Jansen, H. Hermansky, and E. Dupoux. 2013. Evaluating speech features with the minimal-pair abx task: Analysis of the classical mfc/plp pipeline. *INTERSPEECH*.

Thomas Schatz. 2016. *ABX-discriminability measures and applications*. Ph.D. thesis, Paris 6.

Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. 2015. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 298–307.

S. Schneider, A. Baevski, R. Collobert, and M. Auli. 2019. wav2vec: Unsupervised pre-training for speech recognition. *arXiv:1904.05862*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Maarten Versteegh, Xavier Anguera, Aren Jansen, and Emmanuel Dupoux. 2016. The zero resource speech challenge 2015: Proposed approaches and results. *Procedia Computer Science*, 81:67–72.

Weiran Wang, Qingming Tang, and Karen Livescu. 2020. Unsupervised pre-training of bidirectional speech encoders via masked reconstruction. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6889–6893. IEEE.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R Bowman. 2019. Blimp: A benchmark of linguistic minimal pairs for english. *arXiv preprint arXiv:1912.00582*.

Dongqiang Yang and David Martin Powers. 2006. *Verb similarity on the taxonomy of WordNet*. Masaryk University.

Supplementary Materials

A Sampling method to balance ngram scores

We describe here our sampling method to balance ngram scores for sWUGGY and sBLIMP datasets. We first show the algorithm that we applied to sWUGGY, then we just modify slightly the algorithm for the sBLIMP dataset.

For sWUGGY, let's assume that we have N words $w_1, \ldots, w_N$; and for each word w_i, we have a list of K matching nonword candidates $nw_i^1, \ldots, nw_i^K$. We also assume that each word or nonword w has M scores $s_1(w), \ldots, s_M(w)$ (this might be unigram/bigram char/phone scores). We aim to choose, for each word w_i, a matching nonword nw_i^* such that the proportion of the pairs where the score of the word is higher than the score of nonword is close to 50% as possible, for each of M scores.

In other words, we want to build a list of word-nonword pairs $L = \{(w_1, nw_1^*), \ldots, (w_N, nw_N^*)\}$ such that the objective function

$$\text{obj}(L) = \sum_{m=1}^{M} |\text{accuracy\_of\_score\_m(L)-0.5}| \tag{S1}$$

is as close to zero as possible.

We thus deduce a simple sampling method as follows: We first initialize a list L of chosen pairs of word and nonword. At each iteration, we randomly choose an unchosen word. Then we sample a nonword candidate in the list of matching nonword candidates, update the list with the new pair, and compute the objective function of the new list as given in S1. If the objective increases, we remove this newly added element, and resample a new nonword from the list of candidates. If we encounter all the nonword candidates but cannot find a new pair, we randomly choose a nonword from the list of candidates. We then continue to the next word until all the words are chosen.

We found afterwards that if we sample all the words at the same time, we can obtain an overall score very close to 50%, but then words with high frequency or with short length tended to have higher accuracy than others. We then decided to divide the words into sub-categories by frequency and word length, and then do the sampling on each of the sub-categories, which gives a more balanced score on all the length and frequency levels.

For sBLIMP, the candidates are slightly different. We now have a list of N pairs of grammatical and non-grammatical sentences and we want to choose K pairs among them such that the accuracy of the chosen pairs is as close to 50% as possible as for sWUGGY. We can then use the same sampling method as described above, with the exception that instead of choosing a word and sampling the nonword candidates at each iteration, we sample an unchosen pair in the list of candidates, and add that pair to the chosen list if we succeed to decrease the objective function.

As we also found that there is a huge difference in the accuracy scores of linguistic paradigms, we tried to do the sampling by each sub-paradigm. However, there were still some paradigms for which we were not able to perfectly balance the score.

B Supplementary ABX methods and results

Given two sounds x and y with two sequences of representations $\mathbf{r}^x = r_1^x, \ldots, r_T^x$ and $\mathbf{r}^y = r_1^y, \ldots, r_S^y$ respectively, the ABX distance between x and y is computed as follows:

$$d_{ABX}(x, y) = \frac{1}{|\text{path}_{\text{DTW}}(\mathbf{r}^x, \mathbf{r}^y)|} \sum_{(i,j) \in \text{path}_{\text{DTW}}(\mathbf{r}^x, \mathbf{r}^y)} sim(r_i^x, r_j^y). \tag{S2}$$

where $sim(x, y)$ is the arc cosine of the normalized dot product between the embeddings x and y.

Table S1 shows the ABX error on Libri-light dev-clean as a function of different hidden layer of the autoregressive network. We found that as long as we have a big autoregressive network, it is generally not the last layer that brings the best phonetic information of the audio file.

Table S1: Within and Across Speaker ABX error (lower is better) on Libri-light dev-clean at different level of the autoregressive network of CPC-small and CPC-big models. Best layer for each model in bold.

	CPC-small		CPC-big			
LSTM layer	1	2	1	2	3	4
within	10.26	**6.24**	9.62	**3.41**	4.65	9.50
across	14.17	**8.17**	14.73	**4.18**	5.40	9.95

Table S2 reports the ABX scores for different number of clusters, we also included multiple-group clustering in our experiences as similar to Baevski et al. (2020a). We found that the best score is obtained with 50 clusters. Using multiple groups do not further improve the quality of the discretized units, this may be due to the fact that we only used one-hot information of the multiple groups (for example, the two codes 26-20 and 26-10 represent two different one-hot units without any correlation).

Table S2: Within and Across-Speaker ABX error rate (lower is better) on the LibriSpeech dev-clean dataset for CPC-small+kmeans (one-hot vectors embeddings) with different number of units (clusterings). Optimal number of clusters in bold.

nunits	20	50	200	500	2000	50 x 2gr	320 x 2gr
within	11.3	**10.3**	12.5	13.4	17.0	12.6	18.3
across	14.5	**14.2**	16.8	19.9	27.2	17.7	27.7

C Supplementary spot-the-word results

Table S3: **Spot-the-word accuracy** (higher is better) on sWUGGY dev as a function of the masking parameters to compute the pseudo-probabilities. The runtime is estimated based on the evaluation time with the base parameters $M_d = \Delta t = 10$. In bold the compromise we selected between accuracy and speed.

M_d	5		10			15			20		
Δt	5	1	10	5	1	15	5	1	20	5	1
scores	59.14	62.59	64.59	68.23	70.85	66.45	**70.69**	72.52	64.38	69.04	71.33
runtime (est.)	$\times$ 2	$\times$ 10	$\times$ 1	$\times$ 2	$\times$ 10	$\times$ 0.66	$\times$ **2**	$\times$ 10	$\times$ 0.5	$\times$ 2	$\times$ 10

Table S3 investigates the effect of the masking parameters M_d and Δt to the spot-the-word metrics. We found that the way of computing log-probability can greatly influence the evaluation scores. We see that as long as we overlap the masking spans more, the performance is better. In addition, given that we masked spans of $M \sim \mathcal{N}(10, 10)$ tokens during training, the best decoding masking size was found to be 15. Considering the evaluation time, it is theoretically inversely proportional to Δt, and we thus decided to choose $M_d = 15$ and $\Delta t = 5$ for an accuracy and speed trade-off.

Figure S1 shows the performance of the CPC-big system on the BERT-large architecture: they are worse than the toplines but well above chance. We reproduce the frequency effects (more frequent words giving rise to better accuracies) and the length effect (longer words giving rise to better accuracies). This may be due to the fact that the phonetic space is sparser for long than for short words. As a consequence, a short nonword like "tup" could be continued as a real word in multiple ways ("tuple", "tupperware", etc.). In contrast, a long nonword can rarely be salvaged into a word (eg, 'rhanoceros' is a nonword very early on).

D Supplementary grammaticality results

Table S4 shows the detailed results on the various subsets of sBLIMP of our best model. Almost all of the subsets show better than chance scores (11/12), and of the phoneme ngrams controls (11/12),

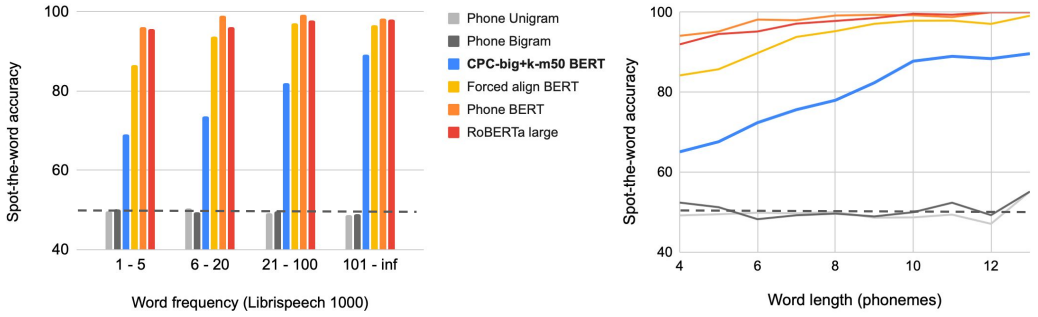

Figure S1: **Spot-the-word accuracy** (sWUGGY dev set, higher is better, chance level at 50%) for our best CPC+clustering+BERT model (blue), compared to phone ngram baselines (gray) and text-based transformer toplines (orange). Left, word frequency effect. Right, word length effect.

and most are better than the word ngrams controls (9/12 for unigram models, and 10/12 for bigram models).

Table S4: **Sentence acceptability accuracy** (sBLIMP dev set, higher is better, chance level at 50%) for our best CPC+kmeans 50+BERT model, compared to phone ngram baselines, text-based transformer toplines, and human scores (from Warstadt et al., 2019).

	Overall	Ana. Agr.	Agr. Str.	Binding	Ctrl. Rais.	D-N Agr.	Ellipsis	Fill. Gap.	Irregular	Island	NPI Li.	Quantifiers	S-V Arg.
Phone Unigram	**48.29**	50.00	50.00	52.90	50.00	50.00	50.00	50.00	45.50	50.00	38.36	39.33	50.00
Phone Bigram	**50.20**	50.50	50.11	52.40	49.80	50.12	50.00	49.88	50.00	49.93	50.00	50.00	50.00
Word Unigram	**54.40**	50.50	50.06	65.20	49.90	50.06	49.50	75.00	51.00	50.00	49.79	50.00	49.92
Word Bigram	**51.64**	50.00	50.06	66.50	50.00	50.06	49.00	50.00	50.00	50.07	50.00	57.00	49.92
CPC-big+km50 BERT	**56.14**	61.50	51.10	62.30	51.62	60.66	74.75	59.91	55.44	56.64	48.29	63.25	51.62
Forced phone BERT	**63.72**	72.62	56.40	63.80	54.90	80.47	69.00	66.34	79.94	58.71	54.29	61.00	65.12
Phone BERT	**66.78**	72.50	59.89	54.40	62.20	92.25	75.00	63.75	82.50	57.71	54.57	81.67	70.17
RoBERTa large	**81.56**	98.50	74.33	80.40	78.20	95.88	99.00	73.62	89.50	68.71	80.71	90.67	87.83
Human (on BLIMP original)	**88.60**	97.50	90.00	87.30	83.90	92.20	85.00	86.90	97.00	84.90	88.10	86.60	90.90

E Supplementary semantic similarity results

Table S5 shows the detailed sSIMI results, layer by layer of the best BERT model together with the detailed ABX results on the same layers. This shows a complementarity of these two metrics (the best layers for acoustics/phonetics are the worst for semantics and vice versa).

Table S5: Comparison of **Semantic similarity scores** (Spearman's correlation with human judgement, higher is better) on the sSIMI synthetic dev set and **ABX scores** on Libri-light dev-clean on different embedding levels of our CPC-big+kmeans50+BERT model. *CPC* refers to the outputs of the second LSTM hidden layer of the CPC-big model, *kmeans* and *outs* refers to 1-hot representations before and after the BERT model repsectively. The semantic similarity scores are also evaluated with different pooling function (mean, max, min). Higher error rates than MFCC baseline in ABX and negative SIMI scores are in red. Note that all the semantic similarity scores are multiplied by 100.

	Score	CPC	kmeans	BERT Layer 0	1	2	3	4	5	6	7	8	9	10	11	12	logits	outs
ABX	within	3.41	6.38	11.82	21.97	35.02	42.54	47.40	44.46	43.71	41.73	33.76	19.67	15.91	15.93	3.30	3.65	5.65
	across	4.18	8.26	13.77	24.59	36.95	43.90	47.94	45.52	44.76	43.12	36.29	23.13	18.92	18.84	4.11	4.59	7.32
sSIMI	mean	-	-	-0.58	-1.97	-1.54	0	1.47	-0.38	1.04	2.26	1.71	2.26	1.47	2.96	-0.57	-	-
	max	-	-	-1.79	0.25	0.51	5.02	6.25	4.03	2.61	1.86	1.69	0.83	1.78	1.78	0.09	-	-
	min	-	-	-3.3	-1.12	-0.93	0.86	6.21	1.9	0.96	0.12	3.53	5.03	0.71	3.41	-0.9	-	-