Improving Unsupervised Sparsespeech Acoustic Models with Categorical Reparameterization

Benjamin Milde, Chris Biemann

Language Technology Group, Dept. of Informatics, Universität Hamburg
{milde,biemann}@informatik.uni-hamburg.de

Abstract

The Sparsespeech model is an unsupervised acoustic model that can generate discrete pseudo-labels for untranscribed speech. We extend the Sparsespeech model to allow for sampling over a random discrete variable, yielding pseudo-posteriorgrams. The degree of sparsity in this posteriorgram can be fully controlled after the model has been trained. We use the Gumbel-Softmax trick to approximately sample from a discrete distribution in the neural network and this allows us to train the network efficiently with standard backpropagation. The new and improved model is trained and evaluated on the Libri-Light corpus, a benchmark for ASR with limited or no supervision. The model is trained on 600h and 6000h of English read speech. We evaluate the improved model using the ABX error measure and a semi-supervised setting with 10h of transcribed speech. We observe a relative improvement of up to 31.3% on ABX error rates within speakers and 22.5% across speakers with the improved Sparsespeech model on 600h of speech data and further improvements when scaling the model to 6000h.

Index Terms: unsupervised learning, unsupervised acoustic models, sparse autoencoders, acoustic unit discovery

1. Introduction

Transcribed and labeled speech data is usually needed to train supervised speech recognition systems, yet it is costly to obtain and transcribe. In contrast, unlabeled speech data can be obtained in vast quantities, even for languages for which much less resources are available as compared to e.g. English.

In recent years, unsupervised acoustic modelling has been gaining traction as viable models emerge to leverage and make use of a treasure trove of unlabeled speech data. The task of acoustic unit discovery has gained significant popularity in unsupervised or zero resource speech processing [1]. Unsupervised unit discovery in isolation can provide insights into datasets, phoneme modelling choices and ultimately provide representations that enables working with raw speech when transcriptions are completely absent.

However, using what unsupervised acoustic models learn and transferring that knowledge in semi-supervised and transfer learning settings is of considerable practical interest. These learning settings hold the promise to boost performance of supervised systems, especially in low-resource settings. In this work, we extend and evaluate an unsupervised acoustic model originally proposed for acoustic unit discovery also in a semi-supervised setting. A large amount of untranscribed speech data (600h-6000h) and only a small amount (10h) of transcribed speech training data is available in this learning setting.

Using the masking technique for unsupervised modelling and then fine-tuning is reminiscent of transformer models such as BERT [2], which are currently very popular on text data. As speech is continuous, using the idea of masking becomes a bit more difficult to transfer directly. In the following, we present our approach that is based on a memory component addressed by Gumbel-Softmax, as part of a larger recurrent encoder/decoder network. We then use the masking technique on the internal representation, i.e. the pseudo-posteriograms used for reconstruction that are generated at each time step are randomly masked.

2. Related work

The ZeroSpeech challenges [3, 1, 4] target speech processing in a zero resource setting, i.e. models that learn from raw speech without transcription. The challenges have also established the use of the ABX discriminability [5, 6] to intrinsically evaluate how well semantically relevant sounds are mapped by a discovered representation in the acoustic unit discovery task. Models trained on untranscribed speech are recently becoming relevant since they can also boost performance in supervised systems:

Schneider et al. [7] showed with wav2vec that pre-training a model on raw speech with a similar binary contrastive loss as word2vec [8] can be effective to improve supervised end-to-end acoustic models. The discrete variant vq-wav2vec [9, 10] of this model with vector quantization into several thousand of units has also been successfully used to pretrain BERT [2] on a sequence masking task, followed by using BERT representations in a wav2letter [11] acoustic model for speech recognition.

Vector quantized variational autoencoders [12] can also be used to learn discrete representations of speech, as demonstrated by the end-to-end system involving attention based ASR and TTS [13] to encode and decode. Wang et al. proposed input masking [14] in recurrent auto-encoders.

Contrastive Predictive Coding (CPC) [15] is a representation learning model trained by predicting future hidden states, which can be applied to raw speech in the time domain. In [16], Kahn et al. created a benchmark for ASR with no or limited supervision, based on English audio books and Librispeech data, also providing results for using CPC. We use the Libri-Light corpus for training and evaluating our models, as it provides a good benchmark for unsupervised acoustic models that can scale well to large amounts of (untranscribed) speech data.

3. Sparsespeech model

In [17], we previously proposed an approach to train unsupervised bi-directional recurrent neural network (RNN) acoustic models that learn discrete representations, with a memory-augmented auto encoder. The Sparsespeech model also uses sequence masking (sequence dropout) on a quasi-symbolic representation that the network generates. The model consists of an encoder that generates the quasi-sparse representation of speech and a decoder that reconstructs the input features from embeddings of a memory component addressed with this quasi-sparse representation. Encoder and decoder are each a bi-directional stacked Long Short-Term Memory (LSTM) [18]. A continuous context vector is also an additional input to the decoder, with the idea to capture and entangle variability of utterance global factors such as speaker identity or the environment. It can be an explicit context representation [19] or speaker vector [20]; in this paper we use an implicit context vector that is the mean of all encoder states, as also evaluated in [17]. This also has the advantage that no separate model needs to be trained. When Sparsespeech representations are generated, we use output of the encoder's softmax. A sparsity constraint and diversity constraint on the encoders softmax output values σ is used in the original model to train the model on continuous approximations of one-hot vectors:

$$\text{Sparsity-}L = 1 - \sup_{n} \sigma_i \tag{1}$$

$$\text{Diversity-}L = \frac{1}{m}\sum_{j=1}^{m} D_{KL}(\sigma_j||U) \tag{2}$$

where m are time steps of an utterance with n softmax outputs per timestep using Kullback-Leibler (KL) divergence [21] and $U(x) = \frac{1}{n}$. A sparsity weight is multiplied with Sparsity-L and a diversity weight is multiplied with Diversity-L, these terms are then added to the mean squared error (MSE) reconstruction loss function. In this paper we also explore using Huber loss [22] as reconstruction loss, as it gives less weight to outliers.

One drawback of the original sparsity constraint is that it cannot be changed at generation time, as it is a hyper-parameter at training time. In this paper, we extend Sparsespeech to model symbolic self-labeling as an (approximated) discrete distribution, introducing an additional parameter that can be used to control the sparseness of the pseudo-posteriorgram representations that our model generates after training.

4. Categorical reparameterization

Discrete variables are difficult to train directly in a neural network, as the backpropagation algorithm cannot by applied to a non-differentable layer. We use categorical reparameterization [23] by Gumbel-Softmax [24] to implement approximate discrete inference within the network while training it. This uses the softmax function as a differentiable approximation to argmax as follows:

We sample a noise vector $\boldsymbol{g} = g_1 \dots g_k$ from a Gumbel distribution with a uniform random sampler U:

$$\boldsymbol{g} = -log(-log(U(0,1))) \tag{3}$$

Where k is the number of elements in the softmax. We then compute the Gumbel-Softmax as:

$$softmax(\frac{logits + \omega \cdot \boldsymbol{g}}{\tau}) \tag{4}$$

Where ω is a noise weight parameter. We set $\omega = 1$ while training the network and $\omega = 0$ after the training is completed to disable the Gumbel noise. The temperature parameter τ controls the amount of sparsity of the sample drawn from the distribution provided by the (unscaled) input logits. We illustrate this in Figure 1 with example samples drawn from the same distribution with varying τ. Lower temperatures (0.05, 0.1, 0.2) tend to make the drawn samples sparse, approximating a one hot vector, while higher temperatures (2.0, 5.0) increase denseness and approximate a uniform distribution.

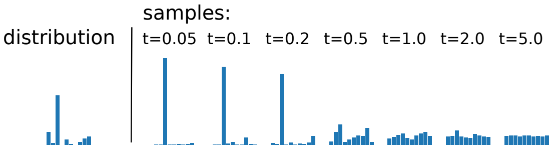

Figure 1: *Drawing samples with different temperatures with the Gumbel-Softmax from a discrete distribution.*

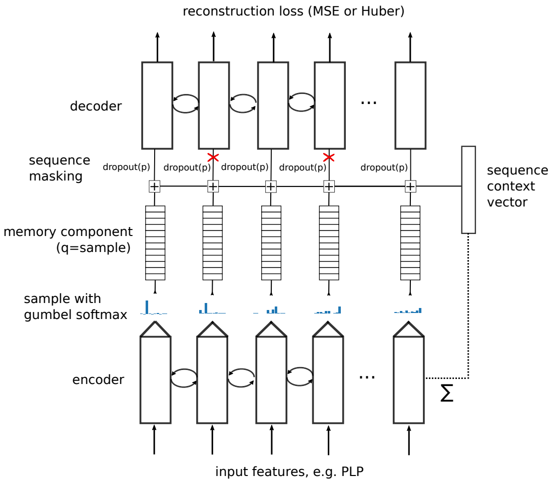

Figure 2: *The Sparsespeech unsupervised acoustic model with Gumbel-Softmax.*

While training the network we use annealing, starting with a higher temperature and slowly decreasing it to a cutoff value below 0.5, for example $\tau = 2 \rightarrow 0.2$. In the Sparsespeech model, the Gumbel-Softmax replaces the regular softmax with the sparsity constraint. Figure 2 illustrates the complete Sparsespeech model with the added Gumbel-Softmax.

5. Setup

We use the newest version of Sparsespeech[1], Tensorflow 1.8 and Python 3.6.9. The relevant changes necessary for the categorical reparameterization have been added to the original model and repository. For evaluation we use the supplied auxiliary scripts of the Libri-Light corpus[2], with minor enhancements, such as the possibility to use Kullback-Leibler (KL) [21] as a distance function in the ABX evaluation[3]. KL is a better metric to compare pseudo-spectograms such as the ones our model generates, while the default cosine distance function of the Libri-Light scripts are better suited for comparing embedding representations.

We use 4-layer stacked BiLSTM decoder/encoders in our Sparsespeech models, with a width of 256 neurons for all experiments. Perceptual linear predictive (PLP) [25] input features are computed with Kaldi [26] using the standard settings of 13 dimensions and 100 frames per second on (downsampled if necessary) 16kHz audio. The sparsity constraint of Sparsespeech is disabled (sparsity weight set to 0) for all experiments with the new model, while the diversity constraint of original model is kept and the diversity weight set to 100. We keep the

[1] https://gitlab.com/milde/sparsespeech/

[2] https://github.com/facebookresearch/libri-light

[3] This change has been merged into the Libri-Light repository.

2-stage training approach of the original model where the model is pre-trained without the memory component. For the second training stage, we use temperature annealing while training the network: the τ parameter for Gumbel-Softmax is set to 2.0 and then slowly decreases by multiplying with an annealing factor (0.9999) every x batches. A cut off parameter, 0.1 or 0.2 in most experiments is set after which the annealing scheme stops.

6. Evaluation

We evaluate on two proposed evaluation tasks in Libri-Light [16]: completely unsupervised and semi-supervised with limited supervision on English audio book read speech. We currently focus on the 600h (small) and 6000h (medium) subsets of untranscribed speech to train our models. In the unsupervised evaluation, we measure ABX error rates [5, 6]. This provides an error rate that measures how well the trained unsupervised representation can differentiate between same/different tri-phones within and across speakers, for example ”bit” vs. ”bat”. It is also agnostic to the representation and can be evaluated on pseudo-labels as well as dense representations. We use the dev sets to calibrate parameters and test the best performing models on the test set. The ABX error measure uses DTW to compare two segments of different length, we use symmetric KL divergence as local comparison function. This is the recommended distance function for posteriorgram-like representations [1, 4]. In the semi-supervised setting, we first train a Sparsespeech model on the unannotated data from Libri-Light. We then follow [16] and evaluate with a simple phoneme classifier with Connectionist Temporal Classification (CTC) loss [27] that is trained on the representation with 10h of limited-resource phone labels.

Table 1: *ABX error rates on features/posteriograms generated by our model for the* ***Libri-Light dev set****, with varying temp. τ.*

Model or features	Temp.	within speaker		across speaker	
	τ	clean	other	clean	other
PLP Features	-	11.12	15.08	25.87	33.74
S6000h-n42-$\tau 2 \rightarrow 0.1$	0.2	12.66	15.52	18.86	24.84
”	0.8	11.04	13.65	17.02	23.01
”	1.0	10.66	13.25	16.34	22.55
”	2.0	9.57	12.15	14.73	20.68
”	**3.0**	**9.51**	**12.15**	**14.41**	**20.25**
”	5.0	10.48	12.94	15.28	20.87

In Table 1 we generate pseudo-posteriorgrams with different temperatures τ from the same model. This model has been trained on 6000h, with 42 components in the memory bank and output representation (n42) and a temperature annealing training scheme of $\tau 2 \rightarrow 0.1$. The sparseness of the output can also be controlled with τ after training. The ABX error measure is sensitive to too sparse representations, as a different scaling with a higher τ significantly reduces the error measure. Temperatures 2.0 and 3.0 produced the lowest ABX error rates.

In Table 2 we mainly evaluate different n in the memory component of Sparsespeech, trained on the Libri-Light small subset of 600h. Using n=100 or n=128 components produced good within speakers results, $n = 42$ performed better on the across speakers ABX error. All models have been trained for 3 epochs, with a training time ranging from 23.8h to 30.43h for the second stage of training (with the memory component) on a single Nvidia Titan XP GPU. We have experimented with different annealing schemes, but settled on $\tau = 2.0 \rightarrow 0.2$ for most experiments. Most models have been trained using MSE as reconstruction loss. For $n = 20$ and $n = 42$ we also trained models with Huber loss, this improved ABX error rates further. Additionally we trained Sparsespeech models ranging from $n = 20$ to $n = 128$ using the original sparsity loss training method without Gumbel-Softmax. For $n = 20$, the sparsity loss constraint weight needed to be increased as the recommended default (2.0) produced a degenerate solution. The original models did not show good ABX error rates on the Libri-Light dev set, in fact only accross speaker ABX error improved over the PLP features baseline. The new models trained with Gumbel-Softmax show significant relative error rate improvements over the original Sparsespeech model, with nearly all tested representations better than PLP features in all settings.

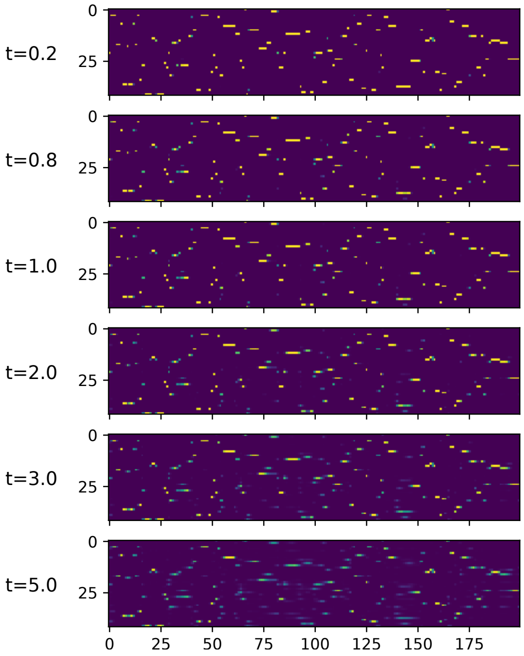

Figure 3: *Example feature representations generated by the Sparsespeech model ”S6000h-n42-$\tau 2 \rightarrow 0.1$” with varying temperature.*

In Table 3 we compare ABX error rates with some selected models on the Libri-Light test set. We compare against baseline PLP features, a baseline Sparsespeech model trained with the original sparsity loss method without Gumbel-Softmax and representations from Contrastive Predictive Coding (CPC) [15] as reported in [16]. While the Sparsespeech models are trained on PLP features (n=13) as input, the CPC are trained on raw 16kHz speech in the time domain. Like on the dev set, there is a large reduction in error rates when training with Gumbel-Softmax. On the larger medium subset, a second training run using the Huber loss did also further improve ABX error rates. However, the dense embedding representations trained with the CPC model show lower error rates than the best Sparsespeech model that we trained on the 6000h medium subset.

In Table 4 we compare the representations of our models in terms of how well a simple phoneme recognizer can classify phonemes with it as input. The phoneme recognizers are trained on 10h of representations with phone labels. They are trained without explicit alignments using the CTC loss. Using only a linear 1D-convolution on posteriorgram-like representa-

Table 2: *ABX error on features/posteriograms generated by our model for the* ***Libri-Light dev set*** *with different n (components in the memory bank). Best results of each section in bold.*

Model or features	**Temp.**	**within speaker**		**across speaker**	
	τ	clean	other	clean	other
PLP Features (n=13)	-	**11.12**	**15.08**	25.87	33.74
S600h-n20-sparsityloss-2.0	-	14.65	17.37	27.09	32.43
S600h-n20-sparsityloss-10.0	-	13.96	17.04	**21.48**	**26.09**
S600h-n42-sparsityloss-10.0	-	14.23	16.16	22.38	27.24
S600h-n100-sparsityloss-2.0	-	14.03	16.63	24.80	29.67
S600h-n128-sparsityloss-2.0	-	15.55	17.94	26.82	31.56
S600h-n20-$\tau 2 \rightarrow 0.5$	2.0	11.56	13.75	21.18	26.66
S600h-n20-$\tau 2 \rightarrow 0.2$	2.0	11.57	13.76	21.12	26.65
S600h-n42-$\tau 2 \rightarrow 0.2$	3.0	11.38	13.49	**17.64**	**22.46**
S600h-n100-$\tau 2 \rightarrow 0.2$	3.0	10.43	13.00	18.86	24.08
S600h-n128-$\tau 2 \rightarrow 0.2$	3.0	**10.00**	**12.46**	17.97	23.16
S600h-n256-$\tau 2 \rightarrow 0.2$	3.0	11.41	14.02	22.73	27.55
S600h-n20-$\tau 2 \rightarrow 0.2$-huber	2.0	11.11	13.45	16.37	**21.34**
S600h-n42-$\tau 2 \rightarrow 0.2$-huber	3.0	**10.30**	**12.82**	**16.34**	21.68

Table 3: *ABX error on features/posteriograms generated by our model for the* ***Libri-Light test set****. CPC results are from [16].*

Model or features	**Temp.**	**within speaker**		**across speaker**	
	τ	clean	other	clean	other
PLP Features (n=13)	-	**10.46**	**14.69**	23.78	34.15
S600h-n20-sparsityloss-10.0	-	13.66	16.83	**19.78**	**26.56**
S600h-n100-sparsityloss-2.0	-	14.12	16.97	22.86	30.53
S600h-n20-$\tau 2 \rightarrow 0.5$	2.0	10.92	14.06	18.86	27.16
S600h-n42-$\tau 2 \rightarrow 0.2$	3.0	10.59	13.78	15.68	23.16
S600h-n128-$\tau 2 \rightarrow 0.2$	3.0	9.39	12.42	16.01	23.49
S6000h-n42-$\tau 2 \rightarrow 0.1$	3.0	**9.33**	**12.05**	**13.53**	**20.60**
S600h-n20-$\tau 2 \rightarrow 0.2$-huber	2.0	10.4	13.82	15.32	22.43
S600h-n42-$\tau 2 \rightarrow 0.2$-huber	3.0	9.69	12.79	14.68	22.05
S6000h-n42-$\tau 2 \rightarrow 0.2$-huber	3.0	**8.79**	**11.62**	**12.55**	**19.84**
CPC-600h (n=256)	-	6.90	9.59	9.00	15.10
CPC-6000h (n=256)	-	6.22	8.55	8.05	13.81
CPC-60000h (n=256)	-	**5.83**	**8.14**	**7.56**	**13.42**

tions as in [16] proved to be challenging, as the most frequent emission symbol per timestep with the CTC loss is the blank label. Adding a simple 1-layer LSTM makes sure that the network can learn when to emit a label other than the blank label and also keep track of context. The 1D convolution has a kernel size of 8 (default in the Libri-Light evaluation script) and the number of output channels of the convolution is set to match the number of phones in the transcription plus the blank label (45). The 1-layer LSTM has a fixed hidden size of 100.

A simple decoder with beam search generates the hypothesized phone sequence. Phoneme error rate (PER) is then computed by comparing the sequence to the Libri-Light transcriptions on the dev and test set (these sets are the same as the ones in Librispeech [28]). With the original Sparsespeech model we do not significantly surpass the PER results of the PLP baseline, but with the improved Sparsespeech model the phoneme recognizer can improve PER by 14.1% relative over the PLP baseline on test-clean and by 8.1% relative on test-other.

Table 4: *PER error for training a very simple phoneme recognizer with 10h of data on: PLP features, CPC model features or Sparsespeech model features.*

Model or features	**Temp.**	**dev PER**		**test PER**	
	τ	clean	other	clean	other
PLP Features (n=13)	-	**52.44**	62.36	50.96	**63.13**
S600h-n100-sparsityloss-2.0	-	52.48	**61.65**	**50.48**	63.23
S600h-n20-sparsityloss-10.0	-	57.22	64.84	55.40	65.95
CPC-600h (n=256)	-	40.21	51.80	38.18	53.85
CPC-6000h (n=256)	-	34.40	47.60	34.44	49.40
CPC-60000h (n=256)	-	**31.16**	**46.67**	**32.67**	**48.93**
S600h-n100-$\tau 2 \rightarrow 0.2$	3.0	50.39	59.82	48.29	61.75
S600h-n128-$\tau 2 \rightarrow 0.2$	3.0	50.56	60.20	48.05	61.69
S600h-n42-$\tau 2 \rightarrow 0.2$-huber	3.0	49.80	59.09	47.16	60.36
S6000h-n42-$\tau 2 \rightarrow 0.1$	3.0	47.77	57.77	46.61	59.61
S6000h-n42-$\tau 2 \rightarrow 0.2$-huber	3.0	**45.18**	**56.31**	**43.77**	**58.02**

7. Conclusion

We proposed to improve the Sparsespeech model with Gumbel-Softmax and Huber loss. On representations with $n = 20$, this yields a relative reduction of 22.5% in ABX error rates on the test set (with clean speech) accross speakers compared to the original model in [17]. Using Gumbel-Softmax in the Sparsespeech model is an effective improvement, as the dimensionality of the learned representations of the new Sparsespeech model can now also be scaled up to representations with bigger n. This did also improve ABX error rates further. So far $n = 100$ and $n = 128$ yielded the best results for within speaker ABX when trained on 600h of untranscribed speech. Representations with bigger n also show a relative improvement of up to 31.3% on ABX error rates within speakers on the clean test set compared to the best original model. Our first results on training on the 6000h medium subset of the Libri-Light corpus further improved error rates and shows that the model is scaling. Currently, tuning the temperature parameter after a Sparsespeech model has been trained seems to be important to reduce ABX error rates, but higher temperatures when generating Sparsespeech features such as 2.0 and 3.0 seem to work well across models with different hyperparameters.

PER error rates also show an 14.1% improvement over a PLP baseline with the new model when a simple phoneme recognizer is trained on the representations. The generated representations from the new model are still relatively compact and sparse (see also Figure 3) with better phoneme discriminability as measured by ABX and PER than PLP features. However when we compare ABX and PER error to unsupervised dense embedding representations such as the ones generated by CPC (n=256), there still a relatively large gap in error rates on the Libri-Light test set. One difference is the type of input features; CPC uses raw waveforms in the time domain while we have used PLP features. We also plan to try out end-to-end learning on raw waveforms, as this could show how much of the performance gap can be attributed to this difference. CPC representations could potentially also be used as input features to the Sparsespeech model or the models could be combined.

Another major difference is structural in the type of the generated representations. There might be a trade-off in ABX error rates between low-bitrate sparse representations and higher bitrate dense representations. The results from last year's Zero Resource Challenge [4] support this hypothesis with systems with higher ABX errors having lower bit rate representations. The organizers concluded that ”this suggests that discretizing learned speech embeddings well is hard”. The pseudo-posteriorgrams that our Sparsespeech model can generate have the advantage over embeddings that they can directly be interpreted as a (soft) clustering of phoneme-like units. They can also be easily discretized and translated to symbolic pseudo transcriptions, where the ABX discriminability is still largely preserved (see [17]).

8. References

[1] E. Dunbar, X. N. Cao, J. Benjumea, J. Karadayi, M. Bernard, L. Besacier, X. Anguera, and E. Dupoux, "The Zero Resource Speech Challenge 2017," in *Proceedings of Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 323–330.

[2] 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, MN, USA, 2019, pp. 4171–4186.

[3] M. Versteegh, R. Thiolliere, T. Schatz, X. N. Cao, X. Anguera, A. Jansen, and E. Dupoux, "The Zero Resource Speech Challenge 2015," in *Proceedings of Interspeech*, Dresden, Germany, 2015, pp. 3169–3173.

[4] E. Dunbar, R. Algayres, J. Karadayi, M. Bernard, J. Benjumea, X.-N. Cao, L. Miskic, C. Dugrain, L. Ondel, A. W. Black *et al.*, "The zero resource speech challenge 2019: TTS without T," in *Proceedings of Interspeech*, Graz, Austria, 2019, pp. 1088–1092.

[5] T. Schatz, V. Peddinti, F. Bach, A. Jansen, H. Hermansky, and E. Dupoux, "Evaluating speech features with the minimal-pair ABX task: Analysis of the classical MFC/PLP pipeline," in *Proceedings of Interspeech*, Lyon, France, 2013, pp. 1781–1785.

[6] T. Schatz, "ABX-discriminability measures and applications," Ph.D. dissertation, Université Paris 6 (UPMC), 2016.

[7] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised Pre-Training for Speech Recognition," in *Proceedings of Interspeech*, Graz, Austria, 2019, pp. 3465–3469.

[8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, USA, 2013, pp. 3111–3119.

[9] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *Proceedings of International Conference on Learning Representations (ICLR)*, Virtual Addis Ababa, Ethopia, 2020.

[10] A. Baevski and A. Mohamed, "Effectiveness of self-supervised pre-training for ASR," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7694–7698.

[11] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[12] A. van den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," in *Proceedings of Advances in Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017, pp. 6306–6315.

[13] A. Tjandra, B. Sisman, M. Zhang, S. Sakti, H. Li, and S. Nakamura, "VQVAE Unsupervised Unit Discovery and Multi-Scale Code2Spec Inverter for Zerospeech Challenge 2019," in *Proceedings of Interspeech*, Graz, Austria, 2019, pp. 1118–1122.

[14] W. Wang, Q. Tang, and K. Livescu, "Unsupervised pre-training of bidirectional speech encoders via masked reconstruction," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Virtual Barcelona, Spain, 2020, pp. 6889–6893.

[15] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[16] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A.-r. Mohamed, and E. Dupoux, "Libri-light: A benchmark for asr with limited or no supervision," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Virtual Barcelona, Spain, 2020, pp. 7669–7673.

[17] B. Milde and C. Biemann, "Sparsespeech: Unsupervised acoustic unit discovery with memory-augmented sequence autoencoders," in *Proceedings of Interspeech*, Graz, Austria, 2019, pp. 256–260.

[18] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] B. Milde and C. Biemann, "Unspeech: Unsupervised speech context embeddings," in *Proceedings of Interspeech*, Hyderabad, India, 2018, pp. 2693–2697.

[20] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[21] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[22] P. J. Huber, "Robust estimation of a location parameter," *Breakthroughs in statistics*, vol. 53, no. 1, pp. 73–101, 1964.

[23] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *Proceedings of International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[24] E. J. Gumbel, *Statistical theory of extreme values and some practical applications: a series of lectures*. US Government Printing Office, 1948, vol. 33.

[25] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[26] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Waikoloa Village, HI, USA, 2011.

[27] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, PA, USA, 2006, pp. 369–376.

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