Unsupervised Subword Modeling Using Autoregressive Pretraining and Cross-Lingual Phone-Aware Modeling

Feng, Siyuan; Scharenborg, Odette

DOI
10.21437/Interspeech.2020-1170

Publication date
2020

Document Version
Final published version

Published in
Proceedings of Interspeech 2020

INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Unsupervised Subword Modeling Using Autoregressive Pretraining and Cross-Lingual Phone-Aware Modeling

Siyuan Feng, Odette Scharenborg

Multimedia Computing Group, Delft University of Technology, Delft, the Netherlands
{s.feng, o.e.scharenborg}@tudelft.nl

Abstract

This study addresses unsupervised subword modeling, i.e., learning feature representations that can distinguish subword units of a language. The proposed approach adopts a two-stage bottleneck feature (BNF) learning framework, consisting of autoregressive predictive coding (APC) as a front-end and a DNN-BNF model as a back-end. APC pretrained features are set as input features to a DNN-BNF model. A language-mismatched ASR system is used to provide cross-lingual phone labels for DNN-BNF model training. Finally, BNFs are extracted as the subword-discriminative feature representation. A second aim of this work is to investigate the robustness of our approach's effectiveness to different amounts of training data. The results on Libri-light and the ZeroSpeech 2017 databases show that APC is effective in front-end feature pretraining. Our whole system outperforms the state of the art on both databases. Cross-lingual phone labels for English data by a Dutch ASR outperform those by a Mandarin ASR, possibly linked to the larger similarity of Dutch compared to Mandarin with English. Our system is less sensitive to training data amount when the training data is over 50 hours. APC pretraining leads to a reduction of needed training material from over 5,000 hours to around 200 hours with little performance degradation.

Index Terms: unsupervised subword modeling, autoregressive predictive coding, cross-lingual knowledge transfer

1. Introduction

Training a DNN acoustic model (AM) for a high-performance automatic speech recognition (ASR) system requires a huge amount of speech data paired with transcriptions. Many languages in the world have very limited or even no transcribed data [1]. Conventional supervised acoustic modeling techniques are thus problematic or even not applicable to these languages.

Unsupervised acoustic modeling (UAM) refers to the task of modeling basic acoustic units of a language with only untranscribed speech [2–7]. An important task in UAM is to learn frame-level feature representations that can distinguish subword units of the language for which no transcriptions are available, i.e., the *target* language, and is robust to non-linguistic factors, such as speaker change [1, 8]. This problem is referred to as *unsupervised subword modeling*, and is the focus of this study. It is essentially a feature representation learning problem.

There are many interesting attempts to unsupervised subword modeling [2, 3, 6, 9–12]. One research strand is to use purely unsupervised learning techniques [2, 3, 9]. For instance, Chen et al. [2] proposed a Dirichlet process Gaussian mixture model (DPGMM) posteriorgram approach, which performed the best in ZeroSpeech 2015 [8]. Heck et al. extended this approach by applying unsupervised speaker adaptation, which performed the best in ZeroSpeech 2017 [3]. In a recent study [13], a two-stage bottleneck feature (BNF) learning framework was proposed. The first stage, i.e., the front-end, used the factorized hierarchical variational autoencoder (FHVAE) [14] to learn speaker-invariant features. The second stage, the back-end, consisted of a DNN-BNF model [15], which used the FHVAE pretrained features as input features and generated BNFs as the desired subword-discriminative acoustic feature representations. In the case of unsupervised acoustic modeling, no frame labels are available for DNN-BNF model training. In [13], DPGMM was adopted as a building block of the back-end to generate pseudo-phone labels for the speech frames. In another recent study [9], the vector quantized VAE (VQ-VAE) [16] was applied to directly learn the desired feature representation without a back-end model such as the DNN-BNF, and is comparable to state-of-the-art performance [3].

In another research strand, frame-level feature representations that can distinguish subword units in the target language are created using a cross-lingual knowledge transfer approach [10, 11]. Here, out-of-domain (OOD) mismatched language resources are used to train DNN AMs which are further used to extract phone posteriorgrams or BNFs of the target speech. The two research strands mentioned above can also be combined. For instance, [11] proposed to apply the DNN-BNF model, and utilized unsupervised DPGMM and OOD ASR systems to generate two types of frame labels for multi-task DNN-BNF learning. The two label types correspond to the two research strands respectively. The results showed the complementarity of the two label types in unsupervised subword modeling.

The present study adopts a two-stage BNF learning framework similar to [13], and aims at combining unsupervised learning techniques, specifically autoregressive predictive coding (APC) as a front-end, with cross-lingual knowledge transfer in the back-end. Recently, APC has been shown [17] to learn speech feature representations that are beneficial to various downstream tasks, and outperform other effective unsupervised methods such as contrastive predictive coding (CPC) [18] in ASR, speech translation and speaker verification [19]. APC preserves phonetic (subword) and speaker information from the original speech signal, while the two information types are more separable. This makes APC a possibly interesting method for unsupervised subword modeling. In this paper, we investigate the effectiveness of APC in this task for the first time.

In the second stage, a DNN-BNF back-end is trained, using the APC pretrained features as input features. Frame labels required for DNN-BNF model training are obtained using an OOD ASR system as was done in [11]. By doing so, cross-lingual phonetic knowledge is exploited. Two OOD ASR systems trained on different OOD languages are employed for comparison, in order to study the effect of target and OOD language similarity on the performance of the proposed approach.

For low-resource languages for which transcribed data are absent, even unlabeled speech can be costly to collect. The robustness of unsupervised subword modeling methods against

http://dx.doi.org/10.21437/Interspeech.2020-1170

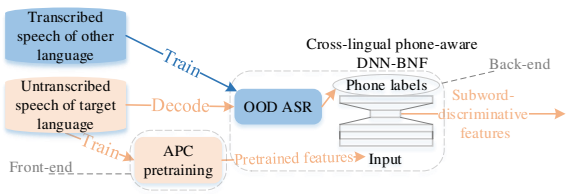

Figure 1: *General framework of the proposed approach to unsupervised subword modeling.*

limited amounts of training material is therefore an important topic, however has received little attention in the literature so far. The second aim of this work is therefore to systematically investigate the robustness of the proposed approach's effectiveness to different amounts of training data. Specifically, we varied the amount of training data from 10 hours to over 500 hours.

2. Proposed approach

The general framework of our proposed approach is illustrated in Figure 1. Given untranscribed speech data of a target language, an APC model is pretrained in the front-end. Next, an OOD ASR system trained on a language different from the target language assigns a phone label to every frame of the target language's speech data through decoding. Pretrained features created by the APC model and the cross-lingual phone labels created by the OOD ASR are then used to train a DNN-BNF model in the back-end, from which BNFs are extracted as the subword-discriminative representation in the final step.

Front-end APC pretraining will be compared with an FHVAE approach [14] which was used in related previous work [13]. The whole pipeline of our approach will be compared with a system consisting of only the back-end DNN-BNF model, and a CPC approach [18] applied in the same task [20]. Moreover, two different languages will be used to train two different OOD ASR systems for comparison.

2.1. APC pretraining

In our concerned task, previously adopted feature learning methods usually target suppressing speaker variation, such as FHVAE [13] and speaker adaptation [3]. In contrast, APC aims at learning a representation that keeps information from speech, while phonetic information is made more separable from speaker information. The learned representation is considered less risky of losing phonetic information than representations learned by methods in [3, 13].

Let us assume a set of unlabeled speech frames $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}\}$ for training, where T is the total number of frames. At each time step t, the encoder of APC model $\mathrm{Enc}(\cdot)$ reads as input a feature vector $\boldsymbol{x_t}$, and outputs a feature vector $\hat{\boldsymbol{x}}_{\boldsymbol{t}}$ (same dimension as $\boldsymbol{x_t}$) based on all the previous inputs $\boldsymbol{x_{1:t}} = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_t}\}$,

$$\hat{\boldsymbol{x}}_{\boldsymbol{t}} = \mathrm{Enc}(\boldsymbol{x_{1:t}}). \tag{1}$$

The goal of APC is to let $\hat{\boldsymbol{x}}_{\boldsymbol{t}}$ be as close as possible to $\boldsymbol{x_{t+n}}$, where n is a pre-defined constant positive integer, denoted as *prediction step*. The loss function during APC training is defined as: $\mathrm{Loss} = \sum_{t=1}^{T-n} |\hat{\boldsymbol{x}}_{\boldsymbol{t}} - \boldsymbol{x_{t+n}}|$. Intuitively, increasing n encourages the encoder to capture contextual dependencies in speech, while a small n focuses more on local smoothness.

Here, the encoder of APC $\mathrm{Enc}(\cdot)$ is realized by a long short-term memory (LSTM) [21] RNN. Let L denote the number of LSTM layers, Equation (1) is formulated as,

$$\boldsymbol{h_0} = \boldsymbol{x_{1:t}}, \tag{2}$$

$$\boldsymbol{h_l} = \mathrm{LSTM}^l(\boldsymbol{h_{l-1}}), l \in \{1, 2, \ldots, L\}, \tag{3}$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{t}} = \boldsymbol{W}\boldsymbol{h_L}, \tag{4}$$

where $\boldsymbol{W}$ is a trainable projection matrix. The equations that form $\mathrm{LSTM}(\cdot)$ can be found in [22].

After APC training, the output of the top hidden layer $\boldsymbol{h_L}$ is extracted as the learned acoustic representation, and is henceforth referred to as the *APC feature*. Although in principle, $\boldsymbol{h_l}$ of any layer l could be used as the learned representation, we follow [17] in using the output of the top layer as they showed that this gave the best results in phone classification tasks.

2.2. Cross-lingual phone-aware DNN-BNF

As shown in Figure 1, the DNN-BNF back-end is a DNN with a bottleneck layer in the middle [23]. To obtain cross-lingual phone labels, the OOD ASR is used to decode target speech utterances into lattices, and find the best path for every utterance. Afterwards, each speech frame is assigned with a triphone HMM state modeled by the OOD ASR. These state labels provide phonetic representation for the target speech from a cross-lingual perspective.

After obtaining triphone HMM state labels as cross-lingual phone labels, the DNN-BNF is trained using the pretrained APC features and the cross-lingual phone labels in a supervised manner [24], and used to extract BNFs as the desired subword-discriminative feature representation.

3. Experimental setup

3.1. Databases and evaluation metric

English is chosen as the target language while Dutch and Mandarin are chosen as the two OOD languages. Training data for APC pretraining and DNN-BNF model training are taken from Libri-light [20], a newly published English database to support unsupervised subword modeling. The *unlab-600* and *unlab-6K* sets from Libri-light are adopted. Unlab-600 is used in both APC pretraining and DNN-BNF model training, while unlab-6K is used only in DNN-BNF model training. Unlab-600 consists of 526 hours of speech excluding silence. Additionally, we randomly select four subsets of utterances from unlab-600 to investigate the robustness of our approach to different amounts of training material. These subsets consist of 900 (i.e., 13 hours), 3.6K (52 hours), 7.2K (104 hours), and 14.4K (209 hours) utterances. Unlab-6K set consists of 5,273 hours of speech excluding silence. Details of the training sets are listed in Table 1.

Table 1: *Libri-light training data and its subsets.*

	unlab-6K	unlab-600	subsets of unlab-600			
#utterances	362,817	36,229	14,400	7,200	3,600	900
#speakers	1,742	489	438	393	351	244
Hours	5,273	526	209	104	52	13

The Dutch and Mandarin corpora used for training the two OOD ASR systems are the CGN [25] and Aidatatang_200zh [26], respectively. The CGN training and test data partition follows [27]. Its training data contains 483 hours of speech, covering speaking styles including conversational and read speech and broadcast news. Aidatatang_200zh is a read speech corpus. Its training data contains 140 hours of speech.

Evaluation data are taken from Libri-light and ZeroSpeech 2017 [1]. Libri-light evaluation sets consist of *dev-clean, dev-*

other, test-clean and *test-other*. , with $*$-clean having higher recording quality and accents closer to US English than $*$-other [28]. They are used to evaluate the effectiveness of both front-end pretrained features and BNFs learned by the back-end.

Evaluation on ZeroSpeech 2017 aims to better compare our approach with previous research in this area. English evaluation data from ZeroSpeech 2017 are used to evaluate the effectiveness of BNFs learned by the the back-end. These data are organized into subsets of differing lengths (1s, 10s & 120s) [1].

The created BNFs, as well as APC pretrained features, are evaluated in terms of the ABX subword discriminability [1]. In the ABX task, A, B and X are three speech segments, and x and y are two different phonemes. $A \in x$, $B \in y$, $X \in x$ or y. Following [1] (see also for more details), an error occurs if given a pre-defined distance measure d, $d(A, X) > d(B, X)$, given $X \in x$, or $d(A, X) < d(B, X)$, given $X \in y$. Dynamic time warping is chosen as the distance measure. Segments A and B belong to the same speaker. ABX error rates for *within-speaker* and *across-speaker* are evaluated separately, depending on whether X and A belong to the same speaker.

3.2. Front-end

The APC model is implemented as a multi-layer LSTM network. Residual connections are made between two consecutive layers. Each LSTM layer has 100 dimensions. Unless specified explicitly, the number of LSTM layers is 3. For each training data amount setting, the prediction step n (in Section 2.1) is picked from $\{1, 2, 3, 4, 5\}$ which gives the best ABX performance. Our preliminary experiments showed that increasing n to larger than 5 would lead to rapid degradation in ABX error rate. The input features to APC are 13-dimension MFCCs with cepstral mean normalization (CMN) at speaker level. The model is trained with the open-source tool by [17] for 100 epochs with the Adam optimizer [29], an initial learning rate of 10^{-4}, and a batch size of 32. After training, the top LSTM layer's output is extracted as the APC feature representation.

The performance of front-end APC pretraining is compared against FHVAE [14], which was used in related previous work [13]. The latent representation $\boldsymbol{z_1}$ of FHVAE is known to be preserving linguistic content while suppressing speaker variation [14], and is compared with the APC feature representation. The model architecture of FHVAE and its training procedure follow those in [13]. The FHVAE models are trained using an open-source tool [14], and take the same input features and training data (i.e., Libri-light) as the APC models. After training, the FHVAE encoder's output $\boldsymbol{z_1}$ is extracted.

3.3. Back-end

3.3.1. OOD ASR systems

We trained two OOD ASR systems, i.e., a Dutch ASR and a Mandarin ASR. The OOD ASR systems use a chain-time delay NN (TDNN) AM [30] trained using Kaldi [31], containing 7 layers. The TDNN is trained based on the lattice-free maximum mutual information (LF-MMI) criterion [30]. For Dutch, the input features consist of 40-dimension high-resolution (HR) MFCCs. For Mandarin, the input features consist of HR MFCCs appended by pitch features [32]. Frame labels required for TDNN training are obtained by forced-alignment with a GMM-HMM AM trained beforehand. For both systems, a trigram LM is trained using training data transcriptions.

The Dutch ASR obtained a word error rate (WER) of 8.98% on the CGN broadcast test set. (This WER could be improved upon by integrating an RNN LM. However, as Dutch ASR performance is not the focus of this study, an RNN LM is not applied.) The Mandarin ASR obtained a character error rate (CER) of 6.37% on the Aidatatang_200zh test set. The two ASR systems are used to generate cross-lingual phone labels for Libri-light training speech frames.

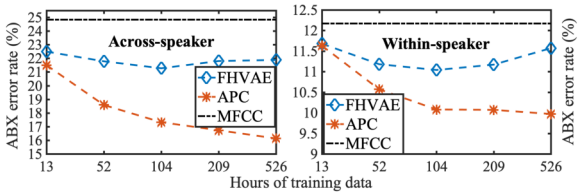

Figure 2: *ABX error rates of APC features, FHVAE features and official MFCC baseline on Libri-light (Avg. over 4 sets).*

3.3.2. DNN-BNF setup

Two DNN-BNF models are trained, one taking the Dutch cross-lingual phone labels as training labels and one taking the Mandarin phone labels as training labels.

The DNN-BNF consists of 7 feed-forward layers (FFLs). Each layer has 450 dimensions except a 40-dimension bottleneck layer, which is located below the top FFL. The DNN-BNF uses a chain model [30] which is trained based on the LF-MMI criterion. The inputs to DNN-BNF are the APC feature with its neighboring (-3 to $+3$) frames. After DNN-BNF training, 40-dimension BNFs are extracted as the learned subword-discriminative representation and evaluated with the ABX task.

For the purpose of comparison, two more DNN-BNF models are trained using the 40-dimension HR MFCC with its neighboring (-3 to $+3$) frames as input features. One model takes the Dutch labels and the other takes the Mandarin labels. Other training and model settings are unchanged. After training, BNFs are extracted and also evaluated with the ABX task.

4. Results and discussion

4.1. Effectiveness of APC features

In this subsection, the APC features and FHVAE features in the front-end ($\boldsymbol{z_1}$) are directly evaluated using the ABX task, without being modeled by the DNN-BNF back-end. ABX error rates (%) of the APC and FHVAE features with respect to different hours of training data are shown in Figure 2. ABX results in this figure are averaged values over the 4 evaluation sets in Libri-light. The official MFCC baseline [20] is also shown in this figure. It can be observed that both the APC features and the FHVAE features outperform the MFCC features. The APC features are consistently superior to the FHVAE features in both the across- and the within-speaker conditions irrespective of the amount of training data. Figure 2 (left) indicates that the APC features are more robust to speaker variation than the FHVAE features, even though the APC model is not explicitly suppressing speaker variation as FHVAE is.

4.2. Effectiveness of BNF representation

In this subsection, all models are trained with unlab-600 (526 hours). ABX error rates (%) of BNFs extracted by the back-end DNN-BNF model are listed in Table 2. The second and third columns denote input feature types and frame labels for training DNN-BNF models. 'Du' and 'Ma' stand for Dutch and Mandarin. Two front-end features, i.e. APC and CPC (in [20]),

Table 2: *ABX error rates of BNFs, APC and CPC on Libri-light. Models are trained with unlab-600. APC has 5 layers, as this gives the best performance among different nos. of APC layers.*

Feature	Input	Label	Across-speaker dev-clean	dev-other	test-clean	test-other	Avg.
BNF	APC	Du	**6.18**	**11.02**	**6.03**	**10.94**	**8.54**
	MFCC	Du	6.67	11.65	6.64	12.00	9.24
	APC	Ma	7.00	11.80	6.84	11.81	9.36
	MFCC	Ma	7.92	12.71	7.74	13.23	10.40
APC	-		12.64	19.00	12.19	18.75	15.65
CPC [20]	-		9.58	14.67	9.00	15.10	12.09
			Within-speaker				
BNF	APC	Du	**4.77**	**6.69**	**4.49**	**6.43**	**5.60**
	MFCC	Du	4.97	6.94	4.73	6.86	5.88
	APC	Ma	5.25	7.14	5.21	7.09	6.17
	MFCC	Ma	6.06	7.71	5.62	7.82	6.80
APC	-		8.83	11.07	8.36	11.48	9.94
CPC [20]	-		7.36	9.39	6.90	9.59	8.31

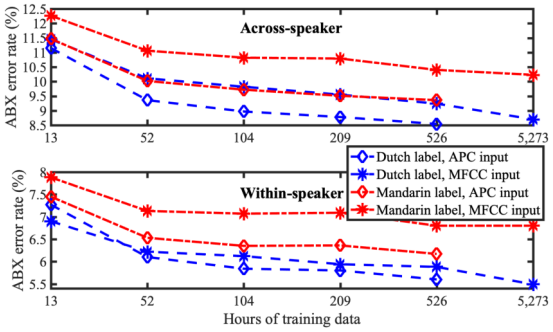

Figure 3: *ABX error rates of BNFs w.r.t amount of training data on Libri-light (Avg. over 4 sets).*

are also listed as references. From this table, it is observed that:

(1) DNN-BNF trained with APC features performs better than that trained with MFCC features in all the evaluation sets. This demonstrates the effectiveness of front-end APC pretraining in our proposed two-stage system framework.

(2) The BNFs obtained from the back-end DNN-BNF model outperform the APC features from the front-end. In other words, the results show that back-end DNN-BNF modeling with cross-lingual phone labels outperforms front-end pretrained features for unsupervised subword modeling, similar to what has been observed by [10, 11]. BNF also performs better than the CPC feature [20]. Note that CPC does not require OOD resources during training while BNF in this study does.

(3) The performance achieved by adopting Dutch labels in DNN-BNF model training is slightly better than that by adopting Mandarin labels. This can possibly be explained by the similarity between the OOD language and target in-domain language, i.e., Dutch and English, respectively, which are both West Germanic languages, while Mandarin is not. Although one could possibly attribute the superiority of adopting Dutch labels over Mandarin labels to the larger amount of training data for Dutch (483 hours) than for Mandarin (140 hours), this is not a likely explanation because both models achieved fairly similar results on their respective in-domain test sets (in Section 3.3.1).

Table 2 also shows CPC outperforms APC. We plan to replace the front-end APC with CPC and study its efficacy in combination with the back-end DNN-BNF model in the future.

4.3. Effect of amount of training data

ABX error rates (%) of BNFs extracted by DNN-BNF models with respect to different amounts of training data in hours are illustrated in Figure 3. The results are averaged values over the 4 evaluation sets in Libri-light. Unlab-6K (5,273 hours) is only adopted in training DNN-BNF models with MFCC input features (marked as "$*$"). For models trained with APC features as input features ("$\diamond$"), the data amount for APC pretraining and DNN-BNF model training is the same for each run. From Figure 3, it can clearly be seen that performance improves as more training data is available, with the largest improvement when the training data increases from 13 hours to 52 hours, and less improvement for any additional training material.

Secondly, across the different data amounts, the DNN-BNF models trained with APC features as input features are almost consistently better than those with MFCC input features. Interestingly, with Dutch labels, the model that uses APC features and is trained with 209 hours of data achieves a similar across-speaker error rate (8.78%) to the model trained with MFCCs with 5,273 hours of data (8.70%). This implies that APC pretraining "saves" around 5,000 hours (i.e. 96%) of training data, making APC pretraining highly appealing in low-resource speech modeling. The effect of pretraining on the needed amount of training data is even larger when Mandarin labels are used (saving over 99% of the training data).

Table 3: *ABX error rates of BNFs on ZeroSpeech 2017 English evaluation sets. Models are trained with Libri-light.*

System	Hours	Across-speaker 1s	10s	120s	Avg.	Within-speaker 1s	10s	120s	Avg.
Proposed-Du	526	7.65	6.69	6.66	**7.00**	5.52	4.77	4.68	**4.99**
	209	8.11	6.99	6.90	7.33	5.83	5.06	4.97	5.29
	104	8.14	7.07	7.03	7.41	5.89	4.99	5.00	5.29
Proposed-Ma	526	8.19	7.33	7.30	7.61	5.97	5.39	5.37	5.58
	209	8.62	7.61	7.52	7.92	6.31	5.52	5.60	5.81
	104	8.47	7.62	7.52	7.87	6.13	5.49	5.44	5.69
Topline [1]		8.6	6.9	6.7	7.40	6.5	5.3	5.1	5.63
[10]		7.9	7.4	6.9	7.40	5.5	5.2	4.9	5.20

4.4. ZeroSpeech 2017 results

We also evaluated the performance of our approach on the ZeroSpeech 2017 English evaluation sets. The results are shown in Table 3, which also includes the official topline [1] and the best-performing system (using OOD data) [10]. Note that, unlike our approach, these two systems employed English labeled data. The total amount of labeled training data used in [10] is 1,327 hours (including 80-hour English data). In this table, "Proposed-Du/-Ma" denotes our proposed approach by adopting Dutch or Mandarin labels respectively. Interestingly, using Dutch labels, our system trained with 526 hours of data outperforms the topline and [10] systems, and is comparable to the two reference systems when trained with only 104 hours of data. Table 3 also shows the proposed approach by adopting Dutch labels performs better than that by adopting Mandarin labels, which is consistent with observations in Section 4.2.

5. Conclusions

This study addresses unsupervised subword modeling, and proposes a two-stage system that consists of APC pretraining and cross-lingual phone-aware DNN-BNF modeling. Experimental results on Libri-light and ZeroSpeech 2017 databases demonstrate the effectiveness of APC in front-end feature pretraining. It surpasses a previously adopted FHVAE approach. Our whole system outperforms the state of the art on both databases. Cross-lingual phone labeling for English data by a Dutch ASR is slightly better than by a Mandarin ASR. This is possibly linked to the larger similarity of Dutch than Mandarin with English. The proposed approach benefits from increasing training data amount, and is less sensitive to data amount when the training data is over 50 hours. When using APC pretraining, 4% of the training material could result in a similar performance to using the full training set without APC pretraining.

6. 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 *Proc. ASRU*, 2017, pp. 323–330.

[2] H. Chen, C.-C. Leung, L. Xie, B. Ma, and H. Li, "Parallel inference of Dirichlet process Gaussian mixture models for unsupervised acoustic modeling: A feasibility study," in *Proc. INTERSPEECH*, 2015, pp. 3189–3193.

[3] M. Heck, S. Sakti, and S. Nakamura, "Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to zerospeech 2017," in *Proc. ASRU*, 2017, pp. 740–746.

[4] H. Kamper, A. Jansen, and S. Goldwater, "A segmental framework for fully-unsupervised large-vocabulary speech recognition," *Computer Speech & Language*, vol. 46, pp. 154–174, 2017.

[5] 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 *Proc. INTERSPEECH*, 2019, pp. 1118–1122.

[6] S. Feng, T. Lee, and Z. Peng, "Combining adversarial training and disentangled speech representation for robust zero-resource subword modeling," in *Proc. INTERSPEECH*, 2019, pp. 1093–1097.

[7] L. Ondel, H. K. Vydana, L. Burget, and J. Cernocký, "Bayesian subspace hidden Markov model for acoustic unit discovery," in *Proc. INTERSPEECH*, 2019, pp. 261–265.

[8] M. Versteegh, R. Thiollière, T. Schatz, X.-N. Cao, X. Anguera, A. Jansen, and E. Dupoux, "The zero resource speech challenge 2015." in *Proc. INTERSPEECH*, 2015, pp. 3169–3173.

[9] J. Chorowski, R. J. Weiss, S. Bengio, and A. v. d. Oord, "Unsupervised speech representation learning using wavenet autoencoders," *arXiv preprint arXiv:1901.08810*, 2019.

[10] H. Shibata, T. Kato, T. Shinozaki, and S. Watanabe, "Composite embedding systems for zerospeech2017 track 1," in *Proc. ASRU*, 2017, pp. 747–753.

[11] S. Feng and T. Lee, "Exploiting cross-lingual speaker and phonetic diversity for unsupervised subword modeling," *IEEE/ACM Trans. Audio, Speech & Language Processing*, vol. 27, no. 12, pp. 2000–2011, 2019.

[12] M. Rivière, A. Joulin, P. Mazaré, and E. Dupoux, "Unsupervised pretraining transfers well across languages," in *Proc. ICASSP*, 2020, pp. 7414–7418.

[13] S. Feng and T. Lee, "Improving unsupervised subword modeling via disentangled speech representation learning and transformation," in *Proc. INTERSPEECH*, 2019, pp. 281–285.

[14] W.-N. Hsu, Y. Zhang, and J. R. Glass, "Unsupervised learning of disentangled and interpretable representations from sequential data," in *Advances in NIPS*, 2017, pp. 1876–1887.

[15] H. Chen, C.-C. Leung, L. Xie, B. Ma, and H. Li, "Multilingual bottle-neck feature learning from untranscribed speech," in *Proc. ASRU*, 2017, pp. 727–733.

[16] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Advances in NIPS*, 2017, pp. 6306–6315.

[17] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. INTERSPEECH*, 2019, pp. 146–150.

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

[19] Y.-A. Chung and J. R. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *Proc ICASSP*, 2020, pp. 3497–3501.

[20] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for ASR with limited or no supervision," in *Proc. ICASSP*, 2020, pp. 7669–7673.

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

[22] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." in *Proc. INTERSPEECH*, 2014, pp. 338–342.

[23] F. Grézl, M. Karafiát, and L. Burget, "Investigation into bottleneck features for meeting speech recognition," in *Proc. INTERSPEECH*, 2009, pp. 2947–2950.

[24] F. Grézl, M. Karafiát, S. Kontár, and J. Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc. ICASSP*, vol. 4, 2007, pp. IV–757.

[25] N. Oostdijk, "The spoken Dutch corpus. overview and first evaluation." in *LREC*. Athens, Greece, 2000, pp. 887–894.

[26] Beijing DataTang Technology Co., Ltd, "Aidatatang 200zh, a free Chinese Mandarin speech corpus."

[27] L. van der Werff, "kaldi_egs_CGN." [Online]. Available: https://github.com/laurensw75/kaldi_egs_CGN

[28] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv*, vol. abs/1412.6980, 2014.

[30] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for ASR based on lattice-free MMI," in *Proc. INTERSPEECH*, N. Morgan, Ed., 2016, pp. 2751–2755.

[31] 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 *Proc. ASRU*, 2011.

[32] P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, "A pitch extraction algorithm tuned for automatic speech recognition," in *Proc. ICASSP*, 2014, pp. 2494–2498.