The effectiveness of self-supervised representation learning in zero-resource subword modeling

Feng, Siyuan; Scharenborg, Odette

DOI
10.1109/IEEECONF53345.2021.9723318

Publication date
2021

Document Version
Accepted author manuscript

Published in
55th Asilomar Conference on Signals, Systems and Computers, ACSSC 2021

The effectiveness of self-supervised representation learning in zero-resource subword modeling

Siyuan Feng
Multimedia Computing Group
Delft University of Technology
Delft, the Netherlands
S.Feng@tudelft.nl

Odette Scharenborg
Multimedia Computing Group
Delft University of Technology
Delft, the Netherlands
O.E.Scharenborg@tudelft.nl

***Abstract*—For a language with no transcribed speech available (the zero-resource scenario), conventional acoustic modeling algorithms are not applicable. Recently, zero-resource acoustic modeling has gained much interest. One research problem is unsupervised subword modeling (USM), i.e., learning a feature representation that can distinguish subword units and is robust to speaker variation. Previous studies showed that self-supervised learning (SSL) has the potential to separate speaker and phonetic information in speech in an unsupervised manner, which is highly desired in USM. This paper compares two representative SSL algorithms, namely, contrastive predictive coding (CPC) and autoregressive predictive coding (APC), as a front-end method of a recently proposed, state-of-the art two-stage approach, to learn a representation as input to a back-end cross-lingual DNN. Experiments show that the bottleneck features extracted by the back-end achieved state of the art in a subword ABX task on the Libri-light and ZeroSpeech databases. In general, CPC is more effective than APC as the front-end in our approach, which is independent of the choice of the out-domain language identity in the back-end cross-lingual DNN and the training data amount. With very limited training data, APC is found similar or more effective than CPC when test data consists of long utterances.**

***Index Terms*—zero-resource, unsupervised subword learning, contrastive predictive coding, autoregressive predictive coding, cross-lingual modeling**

I. Introduction

Conventional automatic speech recognition (ASR) system development relies heavily on annotated speech data for training the acoustic models (AMs). However, for the vast majority of languages not enough annotated training material is available [1]. In the last decade, unsupervised learning of acoustic models for ASR has gained increasing research interests [2]–[4]. It aims at discovering [2], [5], [6] (also referred to as acoustic unit discovery; AUD) or modeling [3], [4], [7] (also referred to as unsupervised subword modeling; USM) a set of basic speech units that represents all the sounds in the language in a *zero-resource* scenario, i.e., with only untranscribed data available. This research field aims to pave the way to developing high-performance ASR systems for languages that have very limited or no transcribed data.

One important research problem is to learn a frame-level feature representation that can distinguish acoustic subword units (phonemes) of a target language and is robust to speaker variation [3]. This problem, denoted as *unsupervised subword modeling* (USM) [3], [8], is the focus of this paper.

There are many interesting approaches to solving the USM problem [7], [9]–[12]. One research line is to apply unsupervised learning techniques, as they naturally fit the zero-resource assumption. Clustering and self-supervised learning (SSL) algorithms are two representatives. In [9], [10], a Dirichlet process Gaussian mixture model (DPGMM) clustering was shown to achieve best performances in ZeroSpeech 2015 [3] and 2017 [8]. Very recently, SSL algorithms, which treat input or modifications of the input as learning targets, were proposed [13]–[15] and extensively investigated for the USM problem [4], [7], [11]. Some studies show that SSL algorithms outperform clustering for USM [7].

In another research line, speech and linguistic resources of non-target, out-of-domain (OOD) languages are leveraged for USM in a cross-lingual knowledge transfer manner [12], [16]. A typical transfer learning approach is to use an OOD ASR system to decode target speech so as to generate phone alignment labels, followed by building a DNN AM of the target language with the OOD phone labels and acoustic data of the target language [17].

The two research lines mentioned above can be combined to achieve a better performance to the USM task. In our recent studies [4], [18], a two-stage bottleneck feature (BNF) learning framework was proposed and achieved state of the art. The first stage adopts an SSL model named autoregressive predictive coding (APC) [15]. The APC model creates features that have the potential to separate phonetic and speaker information in speech. The created features are used as input features to the second stage of the framework, i.e., a cross-lingual DNN AM. The cross-lingual DNN AM uses OOD phone labels provided by a non-target language's ASR system as targets during training, and extracts a BNF representation for the in-domain speech as the learned representation for USM.

Despite the success of APC as the first stage [4], [18], a research question that is still open is whether the choice of APC, from the various SSL algorithms, is optimal for the front-end of our two-stage USM approach. Previous studies on USM and relevant zero-resource speech processing tasks [19], [20] demonstrated the efficacy of contrastive predictive coding (CPC) [14], another SSL algorithm, and reported superiority of CPC over APC. The comparison between CPC and APC as the front-end SSL model in our two-stage approach was

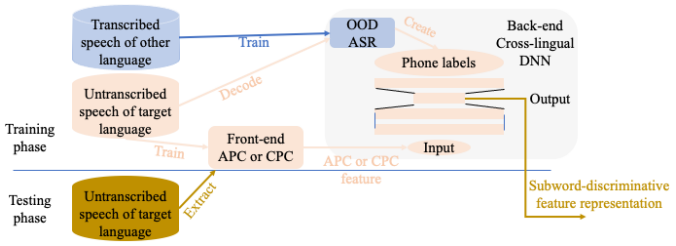

Fig. 1. General framework of the proposed two-stage approach. The two colors in the training phase represent the data used to train each model.

however not investigated before. In this paper, we aim to answer the question whether CPC also outperforms APC in our model, and moreover, we explore as our second research question whether the efficacy of CPC is dependent on the language identity of the OOD ASR system used in the our approach's back-end. Thirdly, we investigate whether any performance differences between CPC and APC are dependent on the amount of available training data. This will provide us insights into the two SSL algorithms' differences in the sensitivity to the amounts of training data (see also [4]), which is a particularly important property in low- and zero-resource speech modeling. We follow the training data amount settings of [4] which ranges between 13 and 526 hours. In short, this paper compares the APC and CPC models across two dimensions: the choice of OOD languages in the back-end model, and the amount of training material. The code is available[1].

II. Proposed Approach

The general framework of our proposed two-stage approach is illustrated in Figure 1. In the first stage, the front-end of our approach pipeline, an APC model or a CPC model creates APC or CPC features. In the second stage, the back-end, a DNN AM extracts the subword-discriminative BNFs.

A. Front-end self-supervised learning (SSL) model

1) Autoregressive predictive coding (APC): An APC model is trained to predict a future speech frame based on current and past speech frames in an utterance. It has an encoder (denoted as Enc$(\cdot)$) which is usually realized as a multi-layer long short-term memory (LSTM) network [15]. Assume a sequence of T unlabeled speech frames for training are denoted as $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}\}$. At time t, Enc$(\cdot)$ outputs a feature vector $\hat{\boldsymbol{x}}_{\boldsymbol{t}}$ (same dimension as $\boldsymbol{x_t}$) based on $\boldsymbol{x_{1:t}} = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_t}\}$, i.e. $\hat{\boldsymbol{x}}_{\boldsymbol{t}} = \text{Enc}(\boldsymbol{x_{1:t}})$. $\hat{\boldsymbol{x}}_{\boldsymbol{t}}$ should be as close as possible to $\boldsymbol{x_{t+n}}$, where n is a pre-defined constant integer, denoted as the *prediction step*. The training objective function of an APC model is defined as: $\text{Loss}_{APC} = \sum_{t=1}^{T-n} |\hat{\boldsymbol{x}}_{\boldsymbol{t}} - \boldsymbol{x_{t+n}}|$.

After APC training, the output of the top LSTM layer is extracted as the learned acoustic representation, and is henceforth referred to as the *APC feature*.

[1]https://github.com/syfengcuhk/libri-light/tree/master/kaldi_related/crs_ling_labeling/cpc_feats.

2) Contrastive predictive coding (CPC): A CPC model is trained to distinguish a near future speech frame from frames of other utterances or more distant future frames of the input utterance. A typical CPC model consists of a convolutional neural network (CNN) based encoder g_{enc} and a recurrent neural network (RNN) based sequence model g_{seq} [14]. The encoder maps speech frames $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_T}\}$ into a latent representation $\{\boldsymbol{z_1}, \boldsymbol{z_2}, \ldots, \boldsymbol{z_T}\}$ by $\boldsymbol{z_t} = g_{enc}(\boldsymbol{x_t})$. Next, the sequence model converts the encoder output to a context latent representation $\boldsymbol{c_t}$ by $\boldsymbol{c_t} = g_{seq}(\{\boldsymbol{z_1}, \boldsymbol{z_2}, \ldots, \boldsymbol{z_t}\})$. Given $\boldsymbol{c_t}$, CPC distinguishes K positive samples $\{\boldsymbol{z_{t+1:t+K}}\}$ from a random set of negative samples (latent representations of other utterances' frames or more distant frames of the input utterance), denoted as $\mathcal{N}_t$. The training objective function of a CPC model is

$$\text{Loss}_{CPC} = -\frac{1}{K}\sum_{t=1}^{T}\sum_{k=1}^{K}\log\frac{\exp(\boldsymbol{z_{t+k}^{\intercal}} g_p(k, \boldsymbol{c_t}))}{\sum_{n\in\mathcal{N}_t}\exp(\boldsymbol{z_n^{\intercal}} g_p(k, \boldsymbol{c_t}))}, \quad (1)$$

where $g_p(k, \cdot)$ can be a Transformer [11] (adopted in this study) or a linear classifier [14].

After CPC training, the context latent representation $\boldsymbol{c_t}$ is extracted as the learned representation, and is henceforth referred to as the *CPC features*.

B. Back-end cross-lingual DNN AM

The back-end DNN AM contains a low-dimension hidden layer (see Figure 1) which is named the bottleneck layer. The DNN AM is trained to predict OOD phone labels given APC or CPC feature representation of the target untranscribed speech. The OOD phone labels are generated by using an OOD, non-target language's ASR system to decode target speech into lattices, and find the best path for every utterance. Afterwards, each speech frame is assigned with a phone label. After training the DNN AM, subword-discriminative BNF representation for target unlabeled speech data is extracted from the output of the bottleneck layer.

III. Experimental Setup

A. Databases and evaluation metric

English is chosen as the target language while Dutch and Mandarin are chosen as the two OOD languages. The English unlabeled training data for training the front-end APC and CPC models and the back-end DNN AM are taken from Libri-light [21], specifically, the *unlab-600* set from Libri-light. Unlab-600 consists of 526 hours of speech recordings. Additionally, we follow [4] in randomly selecting subsets from unlab-600 with varying sizes to investigate the performances of our approach using APC or CPC front-end with regard to different amounts of training material. These subsets consist of 13 to 209 hours. Details of the training (sub)sets are listed in Table I.

The Dutch and Mandarin data used to build the two OOD ASR systems are the Dutch Spoken Corpus (CGN) [22] and Aidatatang_200zh [23] respectively. The CGN training and test data partition follows [24]. Its training data contains 483 hours

TABLE I
LIBRI-LIGHT TRAINING DATA AND ITS SUBSETS.

	unlab-600	subsets of unlab-600			
#utterances	36,229	14,400	7,200	3,600	900
#speakers	489	438	393	351	244
Hours	526	209	104	52	13

of a wide range of speaking styles including conversational and read speech and broadcast news. Aidatatang_200zh is a read speech corpus. Its training data consists of 140 hours of speech.

Our approach is evaluated on two widely adopted databases, namely the Libri-light and ZeroSpeech 2017 [8] datasets. The Libri-light evaluation sets consist of *dev-clean, dev-other, test-clean* and *test-other*, with *-clean having higher quality and more standard US accents than *-other. The ZeroSpeech evaluation sets of English are organized into three parts that differ in per-utterance lengths (1s, 10s and 120s).

The created subword-discriminative BNF representation is evaluated using the ABX subword discriminability metric [3]. 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 [8] (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 d. 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/B belong to the same speaker.

B. Front-end SSL model implementation

1) APC: We follow the setup in [4], [18] for training the APC model. The APC encoder is a 5-layer 100-dimension LSTM network with residual connections between two consecutive layers. It takes 13-dimension MFCC features with cepstral mean normalization (CMN) as input. For each training data amount setting, the prediction step n is picked from $\{1, 2, 3, 4, 5\}$ which gives the best ABX performance. The model is trained by an open-source tool [15] for 100 epochs with the Adam optimizer [25], an initial learning rate of 10^{-4} and a batch size of 32. After training, the output of the top LSTM layer is extracted as the APC features.

2) CPC: We mainly follow the setup in [11] in training the CPC model. The CPC encoder is a 5-layer CNN with kernel sizes: 10, 8, 4, 4, 4, and stride sizes: 5, 4, 2, 2, 2. The CPC sequence model is a 2-layer 256-dimension LSTM network. The model is trained by an open-source tool [11] for 200 epochs with Adam, using numbers of positive and negative samples being 12 and 128 respectively, an initial learning rate of 5×10^{-5} and a batch size of 32. After training, the context latent representation $\boldsymbol{c_t}$ of the top LSTM layer is extracted as the CPC features.

C. Back-end DNN AM implementation

1) OOD ASR systems: Two OOD ASR systems, one for Dutch and the other for Mandarin, are trained beforehand in order to generate OOD phone labels for in-domain (English) untranscribed speech. Both OOD ASR systems use a 7-layer time-delay neural network (TDNN) architecture, implemented by Kaldi [26], and trained with a lattice-free maximum mutual information (LF-MMI) criterion [27]. For the Dutch ASR, the input features consist of 40-dimension MFCC, while for the Mandarin ASR, the input features consist of 40-dimension MFCC plus 3-dimension pitch features. Forced alignment used to train the TDNN AM is obtained by a GMM-HMM AM trained using the same training data. A tri-gram language model (LM) trained on training data transcripts is used for both the Dutch and the Mandarin ASR system.

The Dutch ASR system obtained a word error rate (WER) of 8.98% on the CGN broadcast test set. The Mandarin ASR system obtained a character error rate (CER) of 6.37% on the Aidatatang_200zh test set.

2) Cross-lingual DNN AM: Finally, four cross-lingual DNN AMs are trained, two taking the Dutch phone labels as training labels and two taking the Mandarin phone labels as training labels. Within DNN AMs that use the same OOD cross-lingual phone labels, one uses APC features as the input and the other uses CPC features as the input.

All the four DNN AMs use the same architecture and training criterion: 7 feed-forward layers (FFLs) of 450 dimensions except a 40-dimension bottleneck layer located below the top layer and the LF-MMI criterion. The input APC or CPC features are appended with their respective neighboring frames (-3 to +3) to capture temporal information. After training the cross-lingual DNN AMs, 40-dimension BNF representations are extracted and evaluated by the ABX task. Depending on the choices of DNN AM input features and OOD phone labels, the four BNF representations are denoted as **A-BNF-Du** (APC input, Dutch labels), **A-BNF-Ma** (APC input, Mandarin labels), **C-BNF-Du** (CPC input, Dutch labels) and **C-BNF-Ma** (CPC input, Mandarin labels).

IV. RESULTS AND DISCUSSION

A. The effectiveness of the APC and CPC front-ends

In this subsection, all models trained on the Libri-light training data used the full *unlab-600* set (526 hours). ABX error rates (%) of the A-BNF-Du/-Ma and C-BNF-Du/-Ma evaluated on Libri-light are listed in Table II. The results of A-BNF-Du/-Ma are taken from our previous study [4]. Two reference systems (named M-BNF-Du and M-BNF-Ma) that did not use an SSL front-end but did use the same cross-lingual DNN back-end [18], and one reference system that beat the previous SotA [28] are also listed in Table II. The table shows that:

(1) Both SSL algorithms are effective on the subword ABX task as both systems that use SSL outperform the reference systems without the SSL front-end for both Dutch and Mandarin and for both the across-speaker and within-speaker ABX error rates. At the same time, CPC is more effective than APC for both languages and for both the across-speaker and within-speaker error rates.

TABLE II

ABX ERROR RATES OF BNF REPRESENTATIONS BY ADOPTING AN APC [4] OR CPC FRONT-END, TWO REFERENCE SYSTEMS WITHOUT ADOPTING A FRONT-END [18] AND A PREVIOUS BEST REFERENCE SYSTEM [28] ON LIBRI-LIGHT. MODELS ARE TRAINED WITH UNLAB-600.

System	dev-clean	dev-other	test-clean	test-other	Avg.
			Across-speaker ABX error rate		
A-BNF-Du [4]	6.18	11.02	6.03	10.94	8.54
C-BNF-Du	**5.49**	**9.53**	**5.26**	**9.72**	**7.50**
M-BNF-Du [18]	6.67	11.65	6.64	12.00	9.24
A-BNF-Ma [4]	7.00	11.80	6.84	11.81	9.36
C-BNF-Ma	6.64	10.86	6.39	10.95	8.71
M-BNF-Ma [18]	7.92	12.71	7.74	13.23	10.40
MR [28]	5.89	10.60	5.78	11.00	8.32
			Within-speaker ABX error rate		
A-BNF-Du [4]	4.77	6.69	4.49	6.43	5.60
C-BNF-Du	**4.37**	**5.92**	**4.04**	**5.92**	**5.06**
M-BNF-Du [18]	4.97	6.94	4.73	6.86	5.88
A-BNF-Ma [4]	5.25	7.14	5.21	7.09	6.17
C-BNF-Ma	5.16	6.87	4.93	6.71	5.92
M-BNF-Ma [18]	6.06	7.71	5.62	7.82	6.80
MR [28]	4.67	6.66	4.49	6.81	5.66

TABLE III

ABX ERROR RATES OF BNF REPRESENTATIONS BY ADOPTING AN APC [4] OR CPC FRONT-END, AND A REFERENCE SYSTEM [16] ON THE ZEROSPEECH 2017 ENGLISH EVALUATION SETS. MODELS ARE TRAINED WITH UNLAB-600 IN LIBRI-LIGHT.

	Across-speaker				Within-speaker			
	1s	10s	120s	Avg.	1s	10s	120s	Avg.
A-BNF-Du [4]	7.65	6.69	6.66	7.00	5.52	4.77	4.68	4.99
C-BNF-Du	**6.38**	**6.16**	**6.14**	**6.23**	**4.58**	**4.40**	**4.41**	**4.46**
A-BNF-Ma [4]	8.19	7.33	7.30	7.61	5.97	5.39	5.37	5.58
C-BNF-Ma	7.37	7.20	7.21	7.26	5.37	5.24	5.24	5.28
SH [16]	7.9	7.4	6.9	7.40	5.5	5.2	4.9	5.20

(2) The advantage of CPC over APC as a front-end model is greater when Dutch labels are used compared to when Mandarin labels are used. The absolute ABX error rate reductions from A-BNF-Du to C-BNF-Du are 1.04% (across-speaker) and 0.54% (within-speaker) respectively, both larger than the absolute error rate reductions from A-BNF-Ma to C-BNF-Ma which are 0.65% and 0.25%. On the other hand, looking at the performance difference of the proposed approach with and without adopting a CPC front-end, the ABX error rate difference of C-BNF-Du vs. M-BNF-Du (1.74% across-speaker and 0.82% within-speaker) is highly similar to the difference of C-BNF-Ma vs. M-BNF-Ma (1.69% and 0.88%). This indicates that the effectiveness of the CPC front-end to the proposed two-stage approach is insensitive to the choice of OOD language identity in back-end model training. This is different from when an APC front-end is applied (where a larger improvement was found when using Mandarin labels than when using Dutch labels).

(3) The best performance was obtained with the CPC frontend and the Dutch OOD labels (C-BNF-Du). Moreover, this best performance (C-BNF-Du) outperforms the previous state of the art (shown as MR in Table II) by Rivière and Dupoux [28]. Notably, the system MR is purely trained with Libri-light unlab-600 data, without using OOD data.

The ABX error rates (%) of the A-BNF-Du/-Ma and C-BNF-Du/-Ma evaluated on the ZeroSpeech 2017 corpus are listed in Table III. The results of A-BNF-Du/-Ma are taken from our previous study [4]. The best-performing system in the ZeroSpeech Challenge which exploited OOD cross-lingual resources [16] is also listed as a reference (*SH*). Note that unlike our approach, *SH* [16] used the target language's (English) transcribed data during model training. The total amount of labeled data used in the *SH* system is over 1,000 hours. Similar to what was observed in Table II, Table III shows that adopting a CPC front-end outperforms the APC front-end which is consistent over all the evaluation sets and conditions. The advantage of adopting a CPC front-end over an APC front-end is consistent irrespective of the Dutch or Mandarin OOD phone labels used at the back-end. Moreover, our best performing system (C-BNF-Du) outperforms the current state-of-the-art. This further confirms the effectiveness of CPC in learning front-end features for the USM task.

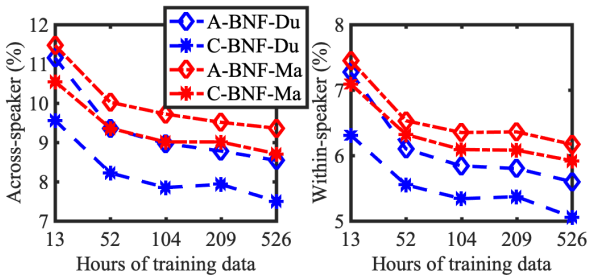

Fig. 2. ABX error rates of the BNF representations when adopting the APC or CPC front-end with respect to the amount of training material on Libri-light (averaged over the 4 test sets). The results of A-BNF-Du and A-BNF-Ma are taken from our previous work [4].

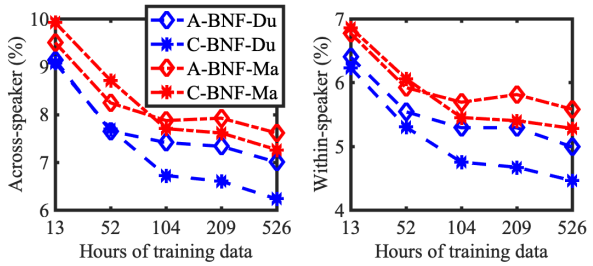

Fig. 3. ABX error rates of the BNF representations when adopting the APC or CPC front-end with respect to the amount of Libri-light training material on ZeroSpeech 2017 (averaged over the 1s, 10s and 120s test sets). The results of A-BNF-Du and A-BNF-Ma with 104, 209 and 526 hours of training data are taken from our previous work [4].

B. *Effect of the amount of training data*

The proposed approach's performances with regard to the amount of Libri-light training material is evaluated on Libri-light and ZeroSpeech 2017. The ABX error rate results of the BNF representations by adopting the APC or CPC front-end on Libri-light are illustrated in Figure 2. The results are averaged over the 4 evaluation sets in Libri-light. Please note that the results related to A-BNF-Du and A-BNF-Ma in Figure 2 are taken from our previous study [4]. All panels in Figure 2 show that the performances of all the systems improve when more

training data becomes available. Comparing the two front-end SSL algorithms shows that CPC ($*$) is consistently more effective than APC ($\diamond$). When training data is more than 52 hours, the performance difference between A-BNF-Du and C-BNF-Du seems to be insensitive to the amount of training material. A similar finding is observed for Mandarin when comparing A-BNF-Ma with C-BNF-Ma. However, for both languages, when the training data amount is very limited (13 hours), the performance gap between C-BNF-Du and A-BNF-Du is larger. This indicates that the advantage of CPC over APC is more prominent in a very low-resource setting.

The ABX error rate results of A-BNF-Du/-Ma and C-BNF-Du/-Ma on ZeroSpeech 2017 English sets are illustrated in Figure 3. The results are averaged values over the 3 English evaluation sets in ZeroSpeech. Part of the results of A-BNF-Du and A-BNF-Ma in Figure 3 (training data: 104 $\sim$ 526 hours) are taken from our previous paper [4]. Figure 3 shows with 104 hours or more training material available, our approach with the CPC front-end consistently outperforms that with the APC front-end. While when limited training data (13 $\sim$ 52 hours) is available, CPC is close to, or worse than APC as the front-end of our approach - opposite to what we observed for Libri-light. Upon looking at the ABX difference per evaluation set (not reported due to page limit), we found APC to be better on the sets with long utterances (10s and 120s) mostly, and CPC always better on the set with short utterances (1s). This is believed to be caused by the use of CMN in APC but not in CPC: a system could greatly benefit from the CMN based speaker normalization if test utterances are very long.

V. Conclusion

This study compares two representative SSL algorithms (APC and CPC) as a front-end method in a recently proposed two-stage unsupervised subword modeling (USM) approach. The experiments on the Libri-light and the ZeroSpeech 2017 databases show that both APC and CPC are effective, with CPC consistently outperforming APC irrespective of the OOD language chosen for back-end model training and the amount of training material. The advantage of CPC is more prominent when the training data amount is very limited for Libri-light. Results on the ZeroSpeech evaluation sets show that with very limited training data, CPC could be worse than APC on very long test utterances (over 10s), which is likely caused by the CMN speaker normalization used in APC but not CPC. The superiority of CPC over APC is however language-dependent: it was greater when Dutch was used as the OOD language than when Mandarin is chosen.

References

[1] O. Scharenborg, L. Besacier, A. Black, M. Hasegawa-Johnson, F. Metze, G. Neubig, S. Stüker, P. Godard, M. Müller, L. Ondel *et al.*, "Speech technology for unwritten languages," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 964–975, 2020.

[2] C.-y. Lee and J. Glass, "A nonparametric bayesian approach to acoustic model discovery," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2012, pp. 40–49.

[3] 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.

[4] S. Feng and O. Scharenborg, "Unsupervised subword modeling using autoregressive pretraining and cross-lingual phone-aware modeling," in *Proc. INTERSPEECH*, 2020, pp. 2732–2736.

[5] L. Ondel, L. Burget, and J. Černockỳ, "Variational inference for acoustic unit discovery," *Proc. SLTU*, vol. 81, pp. 80–86, 2016.

[6] S. Feng, P. Zelasko, L. Moro-Velázquez, and O. Scharenborg, "Unsupervised acoustic unit discovery by leveraging a language-independent subword discriminative feature representation," in *Proc. ICASSP*, 2021.

[7] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using wavenet autoencoders," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2041–2053, 2019.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

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

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

[15] 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.

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

[17] S. Feng and T. Lee, "Exploiting speaker and phonetic diversity of mismatched language resources for unsupervised subword modeling," in *Proc. INTERSPEECH*, 2018, pp. 2673–2677.

[18] S. Feng and O. Scharenborg, "The effectiveness of unsupervised subword modeling with autoregressive and cross-lingual phone-aware networks," *IEEE Open Journal of Signal Processing*, 2021.

[19] M. A. C. Blandón and O. Räsänen, "Analysis of predictive coding models for phonemic representation learning in small datasets," *ICML workshop SAS*, 2020.

[20] L. van Staden and H. Kamper, "A comparison of self-supervised speech representations as input features for unsupervised acoustic word embeddings," in *Proc. SLT*, 2021, pp. 927–934.

[21] 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.

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

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

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

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

[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 *Proc. ASRU*, 2011.

[27] 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*, 2016, pp. 2751–2755.

[28] M. Rivière and E. Dupoux, "Towards unsupervised learning of speech features in the wild," in *Proc. SLT*, 2021, pp. 156–163.