INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Segmental Contrastive Predictive Coding for Unsupervised Word Segmentation

Saurabhchand Bhati[†], *Jesús Villalba*[†,‡], *Piotr Żelasko*[†,‡], *Laureano Moro-Velazquez*[†], *Najim Dehak*[†,‡]

[†]Center for Language and Speech Processing, Johns Hopkins University, USA
[‡]Human Language Technology Center of Excellence, Johns Hopkins University, USA
{sbhati1,jvillalba,pzelasko,laureano,ndehak3}@jhu.edu

Abstract

Automatic detection of phoneme or word-like units is one of the core objectives in zero-resource speech processing. Recent attempts employ self-supervised training methods, such as contrastive predictive coding (CPC), where the next frame is predicted given past context. However, CPC only looks at the audio signal's frame-level structure. We overcome this limitation with a segmental contrastive predictive coding (SCPC) framework that can model the signal structure at a higher level e.g. at the phoneme level. In this framework, a convolutional neural network learns frame-level representation from the raw waveform via noise-contrastive estimation (NCE). A differentiable boundary detector finds variable-length segments, which are then used to optimize a segment encoder via NCE to learn segment representations. The differentiable boundary detector allows us to train frame-level and segment-level encoders jointly. Typically, phoneme and word segmentation are treated as separate tasks. We unify them and experimentally show that our single model outperforms existing phoneme and word segmentation methods on TIMIT and Buckeye datasets. We analyze the impact of boundary threshold and when is the right time to include the segmental loss in the learning process.

1. Introduction

Unsupervised discovery of phoneme or word-like units forms the core objective of Zero Resource speech processing [1–9]. The representation techniques employed to characterize the speech signal dictate the quality of the segmentation and clustering and, in turn, the entire process of discovering linguistic units from the speech signal. A good speech representation becomes crucial for unsupervised systems' good performance. Self-supervised methods have emerged as a promising technique for representation learning from unlabeled speech data [10–13]. In the self-supervised learning (SSL) scenario, the 'self' part refers to the generation of pseudo-labeled training data for an auxiliary task, and 'supervised' refers to the supervised training of the underlying neural model. While most SSL work in speech focuses on learning representations [10–12], SSL has recently been used to identify the spectral changes from the raw waveform and detect phoneme boundaries [13].

The SSL framework exploits the temporal structure present in the speech to learn latent representations. In [13], the auxiliary task is to identify the next frame's latent representation given the latent representation of a reference frame. A feature extractor, e.g., Convolutional Neural Network (CNN), that maps the speech signal to a latent space is optimized using the Noise Contrastive Estimation (NCE) [14] to correctly identify the next frame within a set of frames that includes random distractor frames. The speech signal is composed of underlying linguistic units like phonemes or words. Thus it would be beneficial to learn from the speech structure at phoneme level. But phoneme segmentation is not straightforward and phoneme length is variable, which makes working at phoneme level difficult. That is why current SSL frameworks remain limited to structure at the frame level.

In this work, we present a new method for unsupervised phoneme and word segmentation, for which we propose Segmental Contrastive Predictive Coding (SCPC). This framework can exploit the structure in speech signal at a high scale, i.e., phonemes, during the learning process. A word can be thought of as a sequence of segments. For a certain language, and considering the phonemic domain, segments in a word will occur together frequently and will have a higher likelihood of following each other than segments across words. By locating lower prediction probability time points between adjacent segments, we can get the word boundary candidates.

In the proposed approach, we start from the raw waveform and train an encoder via the next frame prediction to extract frame-level latent representations. We then use a differentiable boundary detector to extract variable-length segments. We develop a simple yet efficient differentiable boundary detection that can detect boundaries without any constraints on segment length or the number of segments in an utterance. Then, we obtain the segment level representation by encoding the averages of the variable-length segments through a segment encoder. The differentiable boundary detector allows information to flow between frame and segment encoder. The model is trained in an end-to-end manner.

Our proposed methods enable boundary detection and segment representation computation in a batch manner for faster training. At the frame level, the model is optimized to predict the next frame. At the segment level, the auxiliary task becomes the next segment prediction. Unlike frames, for segments, the previous segment might not be enough to predict the next segment. We need to learn the context or the order in which the segments occur in. We use a Recurrent neural network to capture the segment context. The joint training allows the model to capture the structure present in the speech at multiple levels, i.e., frames and phonemes level, and these two mutually benefit from each other. We evaluate our proposed methods on TIMIT [15] and Buckeye [16] datasets for phoneme and word boundary detection. Our proposed method outperforms state-of-the-art phoneme and word segmentation methods.

2. Related Work

Most previous works reduce the phoneme boundary detection task to a boundary classification task at each time step. Michel et al. [17] use HMM or RNN to predict the subsequent frame. A peak detection algorithm identifies the high prediction error regions, which are then used as phoneme boundaries. Wang et

http://dx.doi.org/10.21437/Interspeech.2021-1874

al. [18] train an RNN autoencoder and track the norm of intermediate gate values over time e.g., forget gate for LSTM. A peak detection algorithm identified the boundaries from the gate norm over time. Kreuk et al. [13] train a CNN encoder to distinguish between adjacent frame pairs and random pair of distractor frames. To detect the phoneme boundaries, a peak detection algorithm is applied over the model outputs. This method achieves state-of-the-art phoneme segmentation on TIMIT and Buckeye dataset. All these methods try to exploit the structure at frame level to detect phoneme boundaries.

Word segmentation is an important problem in Zero resource speech processing. Bayesian Segmental GMM [6] and Embedded Seegmental K-Means [8] both start from an initial set of subword boundaries and then iteratively eliminate some of the boundaries to arrive at frequently occurring longer word patterns. The initial subword boundaries are not adjusted during the process. As a result, the performance of the ES-Kmeans critically depends on the initial boundaries [9]. Kamper et al. [19] proposed methods to constrain the outputs of vector-quantized (VQ) neural networks and assign blocks of consecutive feature vectors to the same segment. First, vector quantized variational autoencoder (VQ-VAE) [20] and vector quantized contrastive predictive coding (VQ-CPC) [21] models are trained to encode the speech signal into discrete latent space. Then, dynamic programming (DP) is used to merge the frames to optimize a squared error with a length penalty term to encourage longer but fewer segments. The DP segmentation output is further segmented using word segmentation algorithms, e.g. adopter grammar [22], or the Dirichlet process model [23], typically used for segmenting sequences of space-removed characters or phonemes.

The tasks of phoneme segmentation and word segmentation are often done via different methods [6, 8, 9, 24]. Some methods use phoneme segmentation as a starting point for word boundaries [9, 24]. However, none of the methods jointly do phoneme and word segmentation. Here, we propose a single system capable of doing both and able to exploit the two tasks' inter-dependencies.

3. Segmental Contrastive Predictive Coding

We train the Segmental Contrastive Predictive Coding (SCPC) system, depicted in Figure 1, by solving contrastive tasks at multiple scales: at the frame level, $\mathcal{L}_{\text{NFC}}$, and at the segment level, $\mathcal{L}_{\text{NSC}}$, which require the model to identify the true latent representation within a set of distractors. The system's final objective is to minimize the overall loss ($\mathcal{L}$), composed of both the next frame prediction and the next segment prediction losses:

$$\mathcal{L} = \mathcal{L}_{\text{NFC}} + \mathcal{L}_{\text{NSC}} \tag{1}$$

3.1. Next Frame Classifier

Let the sequence $\mathbf{X} = (x_1, x_2, ..., x_T)$ represent a waveform. We learn an encoding function $f_{\text{enc}} : \mathbf{X} \rightarrow \mathbf{Z}$ that maps audio sample sequences to latent spectral representations, $\mathbf{Z}(\in \mathbb{R}^{p \times L}) = (\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_L)$ at low frequency. Each p-dimensional vector $\mathbf{z}_i$ corresponds to a 30 ms audio frame extracted with 10 ms shift. Given frame $\mathbf{z}_t$, the model is trained to identify the next frame $\mathbf{z}_{t+1}$ correctly within a set of $K+1$ candidate representations $\tilde{\mathbf{z}} \in \mathcal{Z}_t$, which includes $\mathbf{z}_{t+1}$ and K distractors–randomly sampled from the same utterance, as

$$\mathcal{L}_{\text{NFC}} = -\log \frac{\exp(\text{sim}(\mathbf{z}_t, \mathbf{z}_{t+1}))}{\sum_{\tilde{\mathbf{z}} \in \mathcal{Z}_t} \exp(\text{sim}(\mathbf{z}_t, \tilde{\mathbf{z}}))} \tag{2}$$

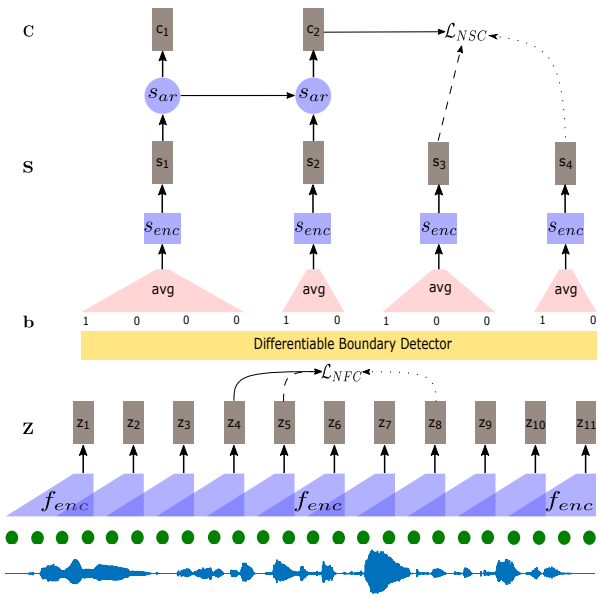


Figure 1: *Overview of the Segmental Contrastive Predictive coding architecture. The solid line represents the reference frame (segment)* $\mathbf{z}_4(\mathbf{c}_2)$*, the dashed line represents the positive frame (segment)* $\mathbf{z}_5(\mathbf{s}_3)$ *and the dotted line represent the negative distractor randomly sampled from the signal*

where $\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}\mathbf{y}^T}{\|\mathbf{x}\|\|\mathbf{y}\|}$ denotes the cosine similarity.

3.2. Differentiable Boundary detection

A phoneme segment can be thought of as a sequence of frames. The frames that follow each other with high similarity are likely to be from the same segment. If there is a high dissimilarity between adjacent frames, then it might indicate a segment change. $\mathbf{d} = (d_1, d_2, ..., d_{L-1})$ captures the dissimilarity between two adjacent frames,

$$\mathbf{d} = 1 - (\mathbf{d}^{\mathbf{s}} - \min(\mathbf{d}^{\mathbf{s}}))/(\max(\mathbf{d}^{\mathbf{s}}) - \min(\mathbf{d}^{\mathbf{s}})) \tag{3}$$

where $\mathbf{d}^{\mathbf{s}} = (d_1^s, d_2^s, ..., d_{L-1}^s)$ s.t. $d_t^s = \text{sim}(\mathbf{z}_t, \mathbf{z}_{t+1})$ captures similarity between adjacent frames. To locate the segment boundaries, we need to find the frame indexes with high dissimilarity values. We can do that by finding peak locations in $\mathbf{d}$. Equation 4 defines the peak detectors $\mathbf{p}^{(1)}$, $\mathbf{p}^{(2)}$ and $\mathbf{p}$ whose t^{th} entries are

$$\begin{aligned} p_t^{(1)} &= \min(\max(d_t - d_{t-1}, 0), \max(d_t - d_{t+1}, 0)) \\ p_t^{(2)} &= \min(\max(d_t - d_{t-2}, 0), \max(d_t - d_{t+2}, 0)) \\ p_t &= \min(\max(\max(p_t^{(1)}, p_t^{(2)}) - \text{thres}, 0), p_t^{(1)}) \end{aligned} \tag{4}$$

$p_t^{(1)}$ captures if d_t is greater than both d_{t-1} and d_{t+1} or not. It will be zero if it is smaller than either d_{t-1} or d_{t+1} and non-zero otherwise. By itself, $p_t^{(1)}$ might be noisy and over-segment. We amend that by introducing $p_t^{(2)}$, which compares d_t with d_{t-2} and d_{t+2}. A peak in the low range can be more informative than one that is higher but otherwise is a trivial member of a tall range. So we check if the height of the peak as compared to its neighbors is more than a threshold or not instead of the peak height directly.

The vector $\mathbf{p}$ will have zeros wherever there are no boundaries and non-zero if there is a boundary. The non-zeros are

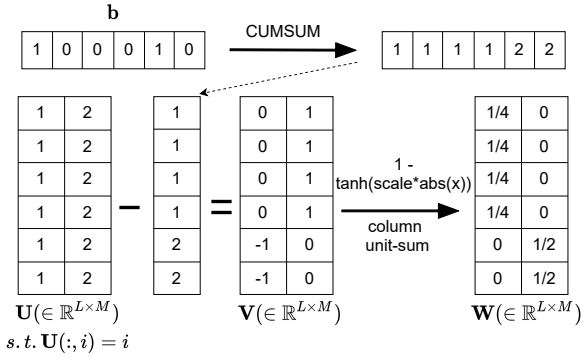


Figure 2: *Weight generation for obtaining segment means in a differentiable manner. The number of segments M can be computed by taking the sum of* $\mathbf{b}$.

values different across peaks, so we scale them to make all the non-zero values consistently 1. We can do that by taking $\tanh$ of a large scalar multiple e.g. 100 of p, but this might result in vanishing gradients. We use a gradient straight-through estimator [25, 26] for obtaining the boundary variables,

$$\begin{aligned} \mathbf{b}_{\text{soft}} &= \tanh(10\ \mathbf{p}) \\ \mathbf{b}_{\text{hard}} &= \tanh(1000\ \mathbf{p}) \\ \mathbf{b} &= \mathbf{b}_{\text{soft}} + \text{sg}(\mathbf{b}_{\text{hard}} - \mathbf{b}_{\text{soft}}) \end{aligned} \tag{5}$$

where sg is the stopping gradient function that avoids gradients to flow through $\mathbf{b}_{\text{hard}}$, which has exploding gradient at 0. The $\mathbf{b}$ is 1 at boundaries and 0 elsewhere. This method can detect segments in a batch manner without any assumptions on segment length or the number of segments.

3.3. Segment representations

After the boundary detection step, the feature sequence $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_L)$ is segmented into disjoint contiguous segments $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_M)$. The next step is to obtain representations for these variable-length segments. A fixed-size representation for segments would be easier to work with and would allow different segments to be compared straightforwardly. Since we decide a segment boundary based on high dissimilarity between frames, the frames that lie within a segment will be close to each other. We represent a segment via the average of constituting frames and feed the averages through a segment encoder s_{enc} to generate the segment representations. From Figure 1, we observe that the first four frames, $\mathbf{z}_{1:4}$ belong to the first segment $\mathbf{s}_1$, and the next two frames $\mathbf{z}_{5:6}$ belong to the second segment $\mathbf{s}_2$. The segment representations are given as $\mathbf{s}_1 = \text{s}_{\text{enc}}(\frac{1}{4}\sum_{i=1}^{4}\mathbf{z}_i)$, $\mathbf{s}_2 = \text{s}_{\text{enc}}(\frac{1}{2}\sum_{i=4}^{5}\mathbf{z}_i)$. We could iteratively compute those averages segment by segment but that would be prohibitively slow. Instead, we propose to vectorize those computations following the steps in Figure 2. Segment representation can be obtained by multiplying the $\mathbf{Z}(\in \mathbb{R}^{p\times L})$ and $\mathbf{W}(\in \mathbb{R}^{L\times M})$ and feeding it through s_{enc}.

3.4. Next Segment Classifier

The segment encoder s_{enc} takes the segment averages as input and outputs the segment representations. We use a recurrent neural network (RNN), $s_{\text{ar}} : \mathbf{S} \rightarrow \mathbf{C}$, to build a contextual representation $(\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_M)$ computed as $c_i = s_{\text{ar}}(s_i)$. Given a reference representation $\mathbf{c}_t$ the model needs to identify the next segment $\mathbf{s}_{t+1}$ correctly from a set of $K+1$ candidate representations, $\tilde{\mathbf{s}} \in \mathcal{S}_t$ which includes $\mathbf{s}_{t+1}$ and K distractors.

$$\mathcal{L}_{\text{NSC}} = -\log \frac{\exp(\text{sim}(\mathbf{c}_t, \mathbf{s}_{t+1}))}{\sum_{\tilde{\mathbf{s}}\in\mathcal{S}_t}\exp(\text{sim}(\mathbf{c}_t, \tilde{\mathbf{s}}))} \tag{6}$$

where sim denotes the cosine similarity.

3.5. Inference

During inference, for a new utterance $\mathbf{X}$ we first extract the frame-level features, $\mathbf{Z}$, using f_{enc}. For phoneme segmentation, the model outputs the dissimilarity between adjacent frames. The frames with high dissimilarity are considered segment boundary candidates. Similar to [13, 17, 18], we apply a peak detection algorithm to find the final segment boundaries. The peak prominence value for the peak detection algorithm is fined-tuned on the validation dataset [13]. For word segmentation, the model outputs dissimilarity score between context representation and the segment representation. This can be interpreted as how likely the model thinks $\mathbf{s}_t$ and $\mathbf{s}_{t+1}$ occur together. Segments that form a word are more likely to be occur together. A peak detection algorithm locates the times with lower prediction which are used as word boundary candidates.

4. Experiments

4.1. Datasets and Evaluation Metrics

We evaluate the proposed model on both TIMIT [15] and Buckeye [16] datasets. For the TIMIT dataset, the standard train/test split is used, with 10% of the train data randomly sampled employed as the validation subset. For Buckeye, we split the data into 80/10/10 for train, test, and validation sets, similarly to [13]. The longer recordings are divided into smaller ones by cutting during untranscribed fragments, noises and silences for ease of training. Each smaller sequence started and ended with a maximum of 20 ms of non-speech. To make results more comparable to [19], we also train our word segmentation system on the English training set from the ZeroSpeech 2019 Challenge [27] and test on Buckeye. ZeroSpeech English set contains around 15 hours of speech from over 100 speakers. Using two different training and evaluation datasets allows us analyzing how well our method generalizes across corpora.

For both the phoneme and word segmentation tasks, we measure the performance with precision (P), recall (R) and F-score with a tolerance of 20 ms [13, 17, 18]. F-score is not sensitive enough to capture the trade-off between recall and Over Segmentation (OS), defined as $R/P - 1$, i.e., even a segmentation model that predicts a boundary every 40 ms can achieve high F1-score by maximizing recall at the cost of low precision. This motivated a more robust metric, R-value [28], more sensitive to OS and the only way to obtain a perfect score (1) is to have perfect recall (1) and perfect OS (0). All the results reported here are an average of three different runs with different seeds unless stated otherwise.

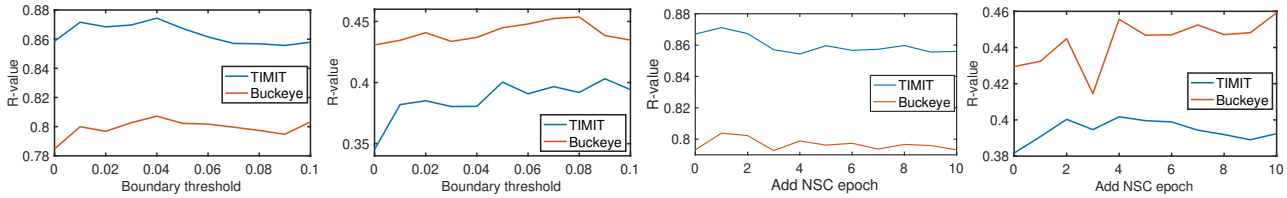
4.2. Effect of Segmental CPC on phoneme segmentation

As observed from Table 1, SCPC outperformed all the baselines. Our proposed approach, SCPC, extracts more structure from the speech data i.e. at frame-level and at phoneme-level, whereas the baselines rely on just the frame-level structure.

The CPC+ line denotes the CPC system trained with additional unlabelled training data from Librispeech [13]. For TIMIT, the 100 hours partition and for Buckeye 500 hours partition are added to their respective training sets. SCPC out-

Table 1: *Comparison of phoneme segmentation performance on TIMIT and Buckeye test sets. All the results use 20 ms tolerance window.*

	TIMIT				Buckeye			
	Precision	Recall	F1	R-val	Precision	Recall	F1	R-val
RNN [17]	74.80	81.90	78.20	80.10	69.34	65.14	67.18	72.13
RNN Gate [18]				83.16	69.61	72.55	71.03	74.83
CPC [13]	83.89	83.55	83.71	86.02	75.78	76.86	76.31	79.69
CPC+ [13]	84.11	84.17	84.13	86.40	74.92	**79.41**	77.09	79.82
SCPC	**84.63**	**86.04**	**85.33**	**87.44**	**76.53**	78.72	**77.61**	**80.72**



(a) *phoneme segmentation* (b) *word segmentation* (c) *phoneme segmentation* (d) *word segmentation*

Figure 3: *Segmentation performance on test portions from TIMIT and Buckeye vs boundary threshold value and add nsc epoch*

performs the CPC trained with just the TIMIT and Buckeye datasets and with data augmentations from Librispeech.

4.3. Word segmentation performance

The word segmentation performance on the Buckeye dataset is shown in Table 2. SCPC clearly outperforms both the neural [19] and non-neural [6, 8] methods for word segmentation. Unlike ES K-Means [8] and BES GMM [6], our proposed method does not require an initial segmentation method and can generate and adjust the boundaries during the learning process. VQ-CPC and VQ-VAE are pretrained without the segmentation task [29] and then optimized for word segmentation, where SCPC is trained jointly. Our model is explicitly encouraged to assign blocks of feature frames to the same segment during the learning process. These approaches have different steps for feature extraction, feature learning, initial segmentation, and then word segmentation. In contrast, in our case, everything is done by a single model jointly with feedback from each other to improve performance.

Buckeye_SCPC denotes the SCPC system trained on Buckeye dataset and tested on same, and ZS_SCPC denotes SCPC system trained on Zerospeech 2019 dataset. The validation and test performance are very close, which shows that the model is generalizing well. The high performance across datasets shows the robustness of this approach. The slight difference in performance is because ZS_SCPC is trained on more data.

4.4. Effect of boundary threshold

To analyze the impact of boundary threshold on system performance, we varied the threshold from 0 to 0.1 with 0.01 step size and trained SCPC models. The segmental loss is added after two epochs. As observed in the Figure 3(a,b) the phoneme segmentation performs better with lower thresholds. Both datasets achieve the best phoneme segmentation performance with a peak threshold of 0.04. The optimal threshold for the word segmentation is higher around 0.09 and 0.08 for TIMIT and Buckeye respectively. We hypothesize that a lower peak threshold allows the model to generate more boundaries which can match better with the higher number of ground truth phoneme boundaries present where word boundaries are fewer, so a higher peak threshold suppresses the unnecessary boundaries for improved performance.

Table 2: *Word boundary segmentation performance on Buckeye development dataset. Parenthesis show performance on test set. All the results use a 20 ms tolerance window.*

Model	P	R	F1	OS	R-value
ES K-Means [8]	30.7	18.0	22.7	-41.2	39.7
BES GMM [6]	31.7	13.8	19.2	-56.6	37.9
VQ-CPC DP [19]	15.5	**81.0**	26.1	421.4	-266.6
VQ-VAE DP [19]	15.8	68.1	25.7	330.9	-194.5
AG VQ-CPC DP [19]	18.2	54.1	27.3	196.4	-86.5
AG VQ-VAE DP [19]	16.4	56.8	25.5	245.2	-126.5
ZS_SCPC	**36.9**	29.9	**33.0**	-19.1	**45.6**
	(34.8)	(31.0)	(32.8)	(-10.8)	(44.5)
Buckeye_SCPC	35.0	29.6	32.1	**-15.4**	44.5
	(33.3)	(29.7)	(31.4)	(-10.8)	(43.4)

4.5. When to add NSC loss

To analyze the importance of the epoch at which the segmental loss is added to the model objective, we trained SCPC where the segmental loss is added after i^{th} epoch and i is varied from 0 to 10. The boundary threshold is kept at 0.05. As observed, in Figure 3(c,d) the phoneme segmentation performs better if segmental loss is added early ($i = 1$) and word segmentation performs better when segmental loss is added late ($i = 4$). We hypothesize that this allows the model to learn better frame-level features before combining them into segments.

5. Conclusions

This work proposes methods to exploit speech signal structure at the segment level and extend the self-supervised learning framework beyond frame-level auxiliary tasks. We propose a differentiable boundary detector to find variable-length segments. Our experimental results indicate that the learned segments correspond to phoneme-like units. Our model is trained to jointly optimize frame and segment level representations from raw speech waveform. Our model outperforms existing phoneme and word segmentation methods on TIMIT and Buckeye datasets.

6. References

[1] A. Jansen and B. Van Durme, "Efficient spoken term discovery using randomized algorithms," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, pp. 401–406, IEEE, 2011.

[2] L. Badino, C. Canevari, L. Fadiga, and G. Metta, "An auto-encoder based approach to unsupervised learning of subword units," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pp. 7634–7638, IEEE, 2014.

[3] M. Huijbregts, M. McLaren, and D. Van Leeuwen, "Unsupervised acoustic sub-word unit detection for query-by-example spoken term detection," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pp. 4436–4439, IEEE, 2011.

[4] 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: Long Papers-Volume 1*, pp. 40–49, Association for Computational Linguistics, 2012.

[5] M.-h. Siu, H. Gish, A. Chan, W. Belfield, and S. Lowe, "Unsupervised training of an hmm-based self-organizing unit recognizer with applications to topic classification and keyword discovery," *Computer Speech & Language*, vol. 28, no. 1, pp. 210–223, 2014.

[6] 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.

[7] S. Bhati, S. Nayak, and K. S. R. Murty, "Unsupervised speech signal to symbol transformation for zero resource speech applications," *Proc. Interspeech 2017*, pp. 2133–2137, 2017.

[8] H. Kamper, K. Livescu, and S. Goldwater, "An embedded segmental k-means model for unsupervised segmentation and clustering of speech," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 719–726, IEEE, 2017.

[9] S. Bhati, H. Kamper, and K. S. R. Murty, "Phoneme based embedded segmental k-means for unsupervised term discovery," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5169–5173, IEEE, 2018.

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

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

[12] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[13] F. Kreuk, J. Keshet, and Y. Adi, "Self-supervised contrastive learning for unsupervised phoneme segmentation," *arXiv preprint arXiv:2007.13465*, 2020.

[14] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 297–304, JMLR Workshop and Conference Proceedings, 2010.

[15] J. S. Garofolo, "Timit acoustic phonetic continuous speech corpus," *Linguistic Data Consortium, 1993*, 1993.

[16] M. A. Pitt, K. Johnson, E. Hume, S. Kiesling, and W. Raymond, "The buckeye corpus of conversational speech: Labeling conventions and a test of transcriber reliability," *Speech Communication*, vol. 45, no. 1, pp. 89–95, 2005.

[17] P. Michel, O. Räsänen, R. Thiolliere, and E. Dupoux, "Blind phoneme segmentation with temporal prediction errors," *arXiv preprint arXiv:1608.00508*, 2016.

[18] Y.-H. Wang, C.-T. Chung, and H.-y. Lee, "Gate activation signal analysis for gated recurrent neural networks and its correlation with phoneme boundaries," *arXiv preprint arXiv:1703.07588*, 2017.

[19] H. Kamper and B. van Niekerk, "Towards unsupervised phone and word segmentation using self-supervised vector-quantized neural networks," *arXiv preprint arXiv:2012.07551*, 2020.

[20] 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.

[21] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *arXiv preprint arXiv:1910.05453*, 2019.

[22] M. Johnson, T. L. Griffiths, S. Goldwater, *et al.*, "Adaptor grammars: A framework for specifying compositional nonparametric bayesian models," *Advances in neural information processing systems*, vol. 19, p. 641, 2007.

[23] S. Goldwater, T. L. Griffiths, and M. Johnson, "A bayesian framework for word segmentation: Exploring the effects of context," *Cognition*, vol. 112, no. 1, pp. 21–54, 2009.

[24] S. Bhati, S. Nayak, K. S. R. Murty, and N. Dehak, "Unsupervised acoustic segmentation and clustering using siamese network embeddings," *Proc. Interspeech 2019*, pp. 2668–2672, 2019.

[25] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *arXiv preprint arXiv:1711.00937*, 2017.

[26] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[27] 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," *arXiv preprint arXiv:1904.11469*, 2019.

[28] O. J. Räsänen, U. K. Laine, and T. Altosaar, "An improved speech segmentation quality measure: the r-value," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[29] B. van Niekerk, L. Nortje, and H. Kamper, "Vector-quantized neural networks for acoustic unit discovery in the zerospeech 2020 challenge," *arXiv preprint arXiv:2005.09409*, 2020.