TRANSCRIPTION FREE FILLER WORD DETECTION WITH NEURAL SEMI-CRFS

Ge Zhu[1], *Yujia Yan*[1], *Juan-Pablo Caceres*[2] *and Zhiyao Duan*[1]

[1]University of Rochester, [2]Adobe Research
{ge.zhu,zhiyao.duan,yujia.yan}@rochester.edu   caceres@adobe.com

ABSTRACT

Non-linguistic filler words, such as "uh" or "um", are prevalent in spontaneous speech and serve as indicators for expressing hesitation or uncertainty. Previous works for detecting certain non-linguistic filler words are highly dependent on transcriptions from a well-established commercial automatic speech recognition (ASR) system. However, certain ASR systems are not universally accessible from many aspects, e.g., budget, target languages, and computational power. In this work, we investigate filler word detection system[1] that does not depend on ASR systems. We show that, by using the structured state space sequence model (S4) and neural semi-Markov conditional random fields (semi-CRFs), we achieve an absolute F1 improvement of 6.4% (segment level) and 3.1% (event level) on the PodcastFillers dataset. We also conduct a qualitative analysis on the detected results to analyze the limitations of our proposed system.

***Index Terms*—** filler word detection, speech disfluency, neural semi-CRFs

1. INTRODUCTION

Speech disfluencies, including repetitions, word fragments, repairs and filler words, are prevalent in spontaneous speech. Content creators often seek to remove these unwanted contents, filler words in particular. Automatically detecting filler words can speed up this laborious process. Such system is also helpful with other speech analysis tasks. For example, in deception detection [1, 2], it is shown that the instances of "um" occur less frequently and appear to be shorter during lying [1]. A preliminary investigation [3] finds that people with Alzheimer's Disease (AD) use more "uh" but less "um". Filler word detection may also be helpful for automatic speech transcription (ASR), as it is shown that filler words can lead to an increase in ASR error rates [4, 5].

Due to the broad interest in detecting filler words, many studies have been proposed in recent years. In natural language processing (NLP), disfluency correction of spontaneous speech transcription is heavily investigated for downstream language processing tasks [6, 7, 8]. However, obtaining an accurate transcription that includes filler words from speech directly is a nontrivial task; it requires either human effort or a sophisticated verbatim ASR system developed on a large annotated dataset. There are also studies on filler word detection on speech signals by integrating with ASR systems. Inaguma et al. [9] proposed to integrate event detection modules to simultaneously recognize speech and detect and fillers. All these above-mentioned systems depend on a well-performing ASR system, which may be inaccessible for applications with limited budget and computational power [10]. In addition, many applications requires the filler words detection to be performed on-device. Therefore, it would be desirable to develop automatic filler word detection methods that do not require verbatim transcription.

Filler detection based on low-level acoustic features [11] have been built using Hidden Markov Models [12], Conditional Random Fields [13] and neural networks [14, 15]. In previous neural network based systems, frame-level classifiers are first trained using convolutional neural networks (CNN), and then the raw probability outputs are fed into a separate temporal post-processor, which involves careful design and tuning. Due to the lack of public datasets, previous works were mainly developed on well-controlled small scale datasets. Moreover, they only evaluate their systems at the frame level without reporting performance at the event level [16]. The *event-based* metrics [17] measure the ability to detect the correct holistic event at the correct temporal position. The performance of the event-based metrics is usually much lower than *segment-based* metrics. We believe that event-based metrics are more suitable for measuring the performance in practice because of its discrete nature.

In this paper, we propose a system that improves transcription free filler word detection: we incorporate a recently proposed lightweight S4 backbone [18] and shows that it improves the overall detection performance; we further use neural semi-CRFs adapted for temporal event detection [19] as a direct formulation for the final output that avoids manually designed post processing steps.

2. PROPOSED SYSTEM

In the problem of filler word detection from speech signals, we are given n frames of input features $x = (x_1, ..., x_n)$, such

This work is partially supported by a New York State Center of Excellence in Data Science award.

[1] Codebase: https://github.com/gzhu06/Filler-semi-CRF

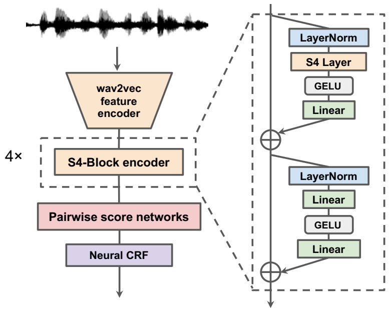

Fig. 1. Proposed transcription-free filler word detector.

as mel-frequency cepstral coefficients (MFCC), the target is to output the intervals $\{(u_i, v_i)\}$ for the filler events, where u_i and v_i indicate the starting time and ending time, respectively. In our proposed system shown in Fig. 1, we feed wav2vec encoded feature maps into a context encoder, followed by a pairwise frame index score network. Eventually, the semi-CRF layer directly outputs the target event time interval.

S4 encoder. During classification, partial word frames with similar phonemes as 'um' or 'uh' can be misclassified without referring to neighboring contexts. One typical example could be the leading part of 'um-brella'. A network architecture that is able to model temporal characteristics is crucial to avoid such false positives. In VC-FillerNet [16], convolutional recurrent neural networks (CRNN) is used as the classifier backbone, which is widely used in sound event detection (SED). In CRNN, the stacked convolutional layers on the top act as feature extractors to learn discriminative time-frequency features. The recurrent layers integrate the extracted features over time to model the context information.

We propose to apply the structured state space sequence (S4) model [18] to replace the CRNN backbone for a fast and accurate model and potentially for real-time applications. S4 is a special case of the discretized state space model (SSM). The discretized state space model inherits the properties of both CNNs and RNNs [20]:

$$h_k = \bar{A}h_{k-1} + \bar{B}x_k, \quad y_k = \bar{C}h_k + \bar{D}x_k, \tag{1}$$

where $\bar{A}, \bar{B}, \bar{C}, \bar{D}$ are discretized SSM parameters. We can rewrite Eq. (1) in the form of a standard RNN setting $\bar{D} = 0$:

$$h_k = f(h_{k-1}, x_k), \quad y_k = g(h_k), \tag{2}$$

where f and g are linear transforms. We can also rewrite Eq. (1) in the form of convolution mimicking CNN by unrolling all of the states defined in Eq. 1, keeping $\bar{D} = 0$:

$$y = \bar{K} * x, \quad \bar{K} = (\overline{CB}, \overline{CAB}, ..., \overline{CA}^{n-1}\overline{B}), \tag{3}$$

where $\bar{K}$ can be viewed as the convolutional kernel. Unlike vanilla CNNs that commonly utilize small kernels and deep layers, the size of $\bar{K}$ is set with length of the entire sequence, achieving a long receptive field with only one layer.

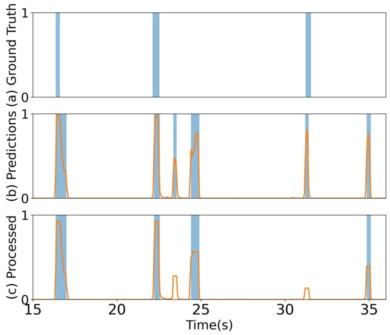

Fig. 2. Examples of the post-processing pipeline of frame-level filler prediction outputs. Top row: blue columns are ground-truth fillers. Middle row: the orange curve is the filler prediction confidence from VC-FillerNet and the blue columns are detected fillers. Bottom row: median filtered confidence curve and resulting fillers.

Specifically, S4 applies a special parameterization of $\bar{A}$ using a diagonal plus low-rank matrix for the SSM, which has two key properties: First, this structured representation is small in size and allows the faster computation of the convolution kernel $\bar{K}$. Second, it allows the SSM to capture long-range sequence information and achieves strong results in sequence classification [18]. To overcome the limited representation ability induced by the linear transform in Eq. (2), we employ the S4 block [21] with an additional feed-forward network using the pre-layernorm design [22], shown in the right part of Fig. 1. The advantages of the proposed backbone is that it achieves a large receptive field over the entire audio feature sequence with a smaller amount of parameters.

Semi-CRF layer. Frame-level filler detectors, which output the framewise likelihood of a position being a filler, usually rely on ad hoc post processing pipelines. Those post-processing steps are usually heuristics-based, typically including smoothing, thresholding, etc, for transforming noisy framewise estimations into clean event intervals. This process is shown in Fig. 2, in this example, we manually tune the median filtering size to 9 to filter the raw predictions for fillers from VC-FillerNet [16]; we can see that true and false positives are filtered out.

Semi-CRFs [23] was originally proposed to address sequence tagging problems in NLP, and recently it was successfully applied to distinguishing disfluent chunks for transcriptions [24] and acoustic modeling [25]. Similarly, we can use semi-CRFs to output the filler word intervals from a sequence of features from the encoder network. Previous approaches using semi-CRFs are meant to find a segmentation of input sequences, of which each segment is labeled. For the scenario of finding disfluent chunks from audio frames, of which positions are discretized from continuous positions, we incorporate a specially adapted version of neural semi-CRFs as proposed in [19]. The difference between this variant and the original versions are as follows. The original semi-CRF models the conditional probability over all possible segmentations; while the one in [19] is over all possible sets of non-overlapping intervals, which contains no specific segment for non-events. Moreover, intervals are allowed to overlap on endpoints (onset/offset) in [19] for allowing an offset and onset to be discretized in the same frame.

Each set of intervals for a specific event type e is modeled as the following conditional probability given the input x:

$$\log p_\theta(Y_e|x) = \log \sum_{(i,j,e)\in Y_e} \exp(f(i,j,e)) - \log(Z(e)), \tag{4}$$

where Y_e denotes the set of events of a specific event type e that are non-overlapping except for the endpoints, and $Z(e)$ is the normalization factor. The function $f(i,j,e)$ assigns a score to an interval $[i,j]$ that is labeled as the event type e. For simplicity we omit the skip score in [19].

We follow the same scoring module design as [19], which is a three-layer feed-forward neural network followed by a three-layer 2-d CNNs that outputs a score tensor with shape $T_1 \times T_0 \times N$. Here T_1 and T_0 are the ending and beginning positions of the entire sequence respectively, and N corresponds to the number of event types. The final training objective is to maximize the total conditional log-likelihood of target event types:

$$\log p_\theta(Y|x) = \sum_{e}^{N} \log p_\theta(Y_e|x). \tag{5}$$

For inference, we use the same Viterbi algorithm as in [19] to infer the most likely set of intervals. More specifically,

$$Y_e^* = arg \max_{Y_e} \log p_\theta(Y_e|x). \tag{6}$$

We refer the readers to [19] for more details.

3. EXPERIMENTAL DESIGN

Dataset. We train and evaluate our proposed system on the PodcastFillers dataset [16] that consists of 145 hours of audio from over 350 speakers in English. The annotations consist of over 85k manually annotated audio events with onsets and offsets including approximately 35k filler words and 50k non-filler events. Practically, other than the target "Filler" class, we also include the events from categories of "Speech", "Laughter", "Breath" and "Music" to train the frame-level encoder on top of the semi-CRF layer to stablize the training. We follow the original dataset split which is based on podcast shows to avoid speaker identity overlapping.

During training, we filtered out apparently non-voiced regions from PodcastFillers with a voice activity detector (VAD) [26]. This filtering procedure was performed with a small threshold to achieve a high recall. As a result, the majority of silence, music and noise were removed.

Training details. Based on the fact that the actual fillers usually have variable length, ranging from 50ms to nearly 1s [14, 27], we use the label resolution of 50ms and perform training on audio segments that are 2s long. As a comparison, the system proposed in [16] is trained on 1s speech segments with labels at a temporal resolution of 100ms.

To achieve better performance, we use the pretrained feature encoder from wav2vec [28] that encodes 16khz audio into a sequence of temporal embeddings with a hop size of 10ms. The receptive field for this pretrained encoder network is 30ms. To make the system deployable, we demand a small-sized model, therefore we do not use the full encoder in [28]. Instead, we only use the first few layers, because we believe that the low-level acoustic features are sufficient for detecting fillers. We use a batch size of 64 and train for 30 epochs using AdamW optimizer with a weight decay of 1e-2 at an initial learning rate of 1e-3 using the cosine annealing scheduler.

Evaluation. We use segment-based and event-based metrics originally proposed for SED [17] to evaluate the filler word detection performance. Segment-based metrics compare the system outputs and the ground truth on a fixed time grid. Event-based metrics directly compare the estimated sound events with the ground-truth events by using a maximum cardinality matching between predictions and ground truths. A predicted event is considered a true positive if and only if it is assigned with the correct label and its onset and offset are within a deviation threshold (200 ms in this work) from the reference event's onset and offset.

Ablations. We run ablation studies, with model configurations shown in Table 1, on the validation split of PodcastFillers to understand the impact of label resolution, model backbones and the semi-CRF layer. We first compare the temporal label resolution at 100ms and 50ms respectively. For backbone comparison, we include CRNN, S4 block and encoder-only transformer with a similar model size. When assessing effects of the semi-CRF layer, to make the ablations comparable, we add an additional median filter of size 3 for the frame-wise classifier without the semi-CRF layer. For the systems with the semi-CRF layer, we directly evaluate on the test split because there is no explicit hyperparameters involved for tuning in this case.

4. RESULTS

Ablation studies. We begin with analyzing the influence of the label resolution by comparing the solid lines versus dashed lines on the precision-recall (PR) curves shown in Fig. 3. For the segment-based metrics, we can see that for different backbone model classes, the systems with a fine label resolution almost consistently outperforms its counterpart with a coarse label resolution. This improvement shows that a finer label resolution is helpful in locating the boundaries of the filler words. For event-based metrics, finer labels tend to have higher recall but lower precision. Since a finer label

Table 1. Segment- and event-based results (%) of our proposed systems on the test split of PodcastFillers. All trainable parameters are single precision floating points and the numbers of parameters exclude the 5M parameters from the wav2vec feature encoder. MHA (multi-head self-attention) stands for transformer encoder.

System	**Label Res.**	**Backbone**	**Semi-CRF**	**# Params**	**Segment-based (%)**			**Event-based (%)**		
	(ms)				Precision	Recall	F1	Precision	Recall	F1
Transcription-based										
AVC [16]	-	CNN		-	**93.0**	**95.4**	**94.2**	**91.7**	**94.0**	**92.8**
Transcription-free										
VC-100-CRNN [16]	100	CRNN		344k	78.4	69.7	73.8	74.8	76.9	73.8
VC-50-CRNN	50	CRNN		344k	79.5	75.4	77.4	72.7	77.1	74.8
VC-50-CRNNCRF	50	CRNN	✓	357k	78.0	79.2	78.6	75.3	74.9	75.1
VC-100-MHA	100	Transformer		213k	72.4	65.9	69.0	68.8	66.2	67.5
VC-50-MHA	50	Transformer		213k	77.4	72.7	75.0	72.4	73.1	72.7
VC-50-MHACRF	50	Transformer	✓	226k	64.1	84.2	72.7	62.0	80.2	69.9
VC-100-S4	100	S4		215k	76.2	75.1	75.6	75.4	74.6	75.0
VC-50-S4	50	S4		215k	82.1	74.3	78.0	75.0	76.4	75.7
VC-50-S4CRF (Ours)	50	S4	✓	221k	80.2	80.1	**80.2**	79.5	74.5	**76.9**

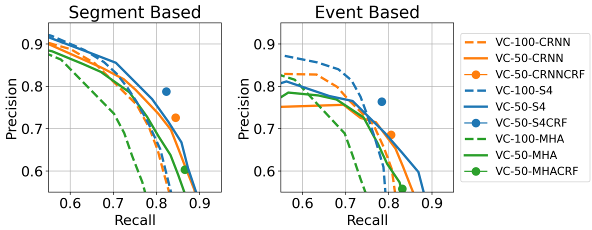

Fig. 3. PR curves for the ablations on the validation split of PodcastFillers. Systems with the semi-CRF layer do not encode threshold for tuning explicitly during training, therefore the result is only one value instead of a curve.

resolution yields more predictions, therefore, it is more prone to misclassification and leads to lower precision.

In the comparison of the backbones differently colored in Fig. 3, we see that when fixing the label resolution, the PR curves of S4 based systems marginally outperforms the CRNN backbone in both metrics, indicating its efficiency and effectiveness in detecting fillers. In S4-based models, the computation of convolution in Eq. (3) requires fast Fourier transform (FFT) and inverse FFT. As a result, they are not as efficient as CRNN-based models. It is shown that such convolution can be replaced with parallel scan to improve efficiency [29], but this optimization is beyond the scope of this paper. Among all of the backbones, the transformer underperforms both CRNN and S4.

After inserting the semi-CRF layer, we observe that it improves over the vanilla systems using S4 and CRNN backbones on both evaluation metrics without any tuning or threshold calibration. But the transformer variant still makes no improvement on the event-based detection.

Comparison to baselines. We compare the detection performance of VC-FillerNet variants on the test split of PodcastFillers, shown in Table 1. We see that VC-50-S4CRF consistently achieves the best performance in both metrics. Although there is still a large gap from the transcription-based system, VC-50-S4CRF achieves 6.4% absolute improvement in segment-based F1 and 3.1% in event-based F1 with a smaller model size over the VC-100-CRNN baseline.

Limitations. We conduct a qualitative analysis on the whole test split to find out the mistakes that transcription-free systems tend to make during the detection. First of all, we track all of the false positives and misses respectively by collecting the events that failed to be matched during the event-based metrics evaluation. Then by looking at the false positives, we analyze the words that are easy to be misclassified as fillers; Similarly, we collect the surrounding words in the misses to analyze.

We observe that the major source of false positives comes from the confusion between words with similar phonemes to fillers: The top five cases are “a”, “the”, “uh-huh”, “oh” and “I”. An ASR system, on the other hand, can provide more context for distinguishing those ambiguous cases with the help of language modeling and alignment. For miss-detected fillers, the top five words that appear before the fillers are “and”, “but”, “the”, “it’s” and “that”. When listening to the audio samples, we found that the miss-detected fillers are mostly “uh” which are generally short and occur at the utterance medially [30]. They mostly happened during the linking words pronunciation, for example, “*... and [uh] probably...*”. Finally, we also find a decent amount of misaligned fillers, longer than the 200ms threshold, causing both false positive and missed errors.

5. CONCLUSION

In this work, we presented a transcription free filler word detection system, which employs a S4 backbone and a neural semi-CRF layer. The S4 block acts as an efficient sequence encoder for the input features and semi-CRF layer directly model filler events and output intervals without any post-processing or threshold calibration. Future work can concentrate on closing the gap from the transcription-based system and extending this framework to the detection of general speech disfluencies without transcriptions.

6. REFERENCES

[1] Joanne Arciuli, David Mallard, and Gina Villar, ""um, i can tell you're lying": Linguistic markers of deception versus truth-telling in speech," *Applied Psycholinguistics*, vol. 31, no. 3, pp. 397–411, 2010.

[2] Gina Villar, Joanne Arciuli, and David Mallard, "Use of "um" in the deceptive speech of a convicted murderer," *Applied Psycholinguistics*, vol. 33, no. 1, pp. 83–95, 2012.

[3] Jiahong Yuan, Yuchen Bian, Xingyu Cai, Jiaji Huang, Zheng Ye, and Kenneth Church, "Disfluencies and fine-tuning pre-trained language models for detection of alzheimer's disease.," in *Interspeech*, 2020, vol. 2020, pp. 2162–6.

[4] Sharon Goldwater, Dan Jurafsky, and Christopher D Manning, "Which words are hard to recognize? prosodic, lexical, and disfluency factors that increase asr error rates," in *Proceedings of ACL*, 2008, pp. 380–388.

[5] Koharu Horii, Meiko Fukuda, Kengo Ohta, Ryota Nishimura, Atsunori Ogawa, and Norihide Kitaoka, "End-to-end spontaneous speech recognition using disfluency labeling," *Proc. Interspeech 2022*, pp. 4108–4112, 2022.

[6] Nguyen Bach and Fei Huang, "Noisy bilstm-based models for disfluency detection.," in *INTERSPEECH*, 2019, pp. 4230–4234.

[7] Shaolei Wang, Wangxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang, "Multi-task self-supervised learning for disfluency detection," in *AAAI*, 2020, vol. 34, pp. 9193–9200.

[8] Paria Jamshid Lou, Peter Anderson, and Mark Johnson, "Disfluency detection using auto-correlational neural networks," in *EMNLP*, 2018, pp. 4610–4619.

[9] Hirofumi Inaguma, Masato Mimura, Koji Inoue, Kazuyoshi Yoshii, and Tatsuya Kawahara, "An end-to-end approach to joint social signal detection and automatic speech recognition," in *2018 ICASSP*. IEEE, 2018, pp. 6214–6218.

[10] Johann C. Rocholl, Vicky Zayats, Daniel D. Walker, Noah B. Murad, Aaron Schneider, and Daniel J. Liebling, "Disfluency Detection with Unlabeled Data and Small BERT Models," in *Proc. Interspeech 2021*, 2021, pp. 766–770.

[11] Kartik Audhkhasi, Kundan Kandhway, Om D Deshmukh, and Ashish Verma, "Formant-based technique for automatic filled-pause detection in spontaneous spoken english," in *2009 ICASSP*. IEEE, 2009, pp. 4857–4860.

[12] Hugues Salamin, Anna Polychroniou, and Alessandro Vinciarelli, "Automatic detection of laughter and fillers in spontaneous mobile phone conversations," in *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2013, pp. 4282–4287.

[13] Björn Schuller, Florian Eyben, and Gerhard Rigoll, "Static and dynamic modelling for the recognition of non-verbal vocalisations in conversational speech," in *International Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems*. Springer, 2008, pp. 99–110.

[14] Lakshmish Kaushik, Abhijeet Sangwan, and John HL Hansen, "Laughter and filler detection in naturalistic audio," 2015.

[15] Rahul Gupta, Kartik Audhkhasi, Sungbok Lee, and Shrikanth S Narayanan, "Paralinguistic event detection from speech using probabilistic time-series smoothing and masking.," in *Interspeech*, 2013, pp. 173–177.

[16] Ge Zhu, Juan-Pablo Caceres, and Justin Salamon, "Filler Word Detection and Classification: A Dataset and Benchmark," in *Proc. Interspeech 2022*, 2022, pp. 3769–3773.

[17] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, "Metrics for polyphonic sound event detection," *Applied Sciences*, vol. 6, no. 6, pp. 162, 2016.

[18] Albert Gu, Karan Goel, and Christopher Ré, "Efficiently modeling long sequences with structured state spaces," *arXiv preprint arXiv:2111.00396*, 2021.

[19] Yujia Yan, Frank Cwitkowitz, and Zhiyao Duan, "Skipping the frame-level: Event-based piano transcription with neural semi-crfs," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20583–20595, 2021.

[20] Albert Gu, Isys Johnson, Karan Goel, Khaled Saab, Tri Dao, Atri Rudra, and Christopher Ré, "Combining recurrent, convolutional, and continuous-time models with linear state space layers," *Advances in neural information processing systems*, vol. 34, pp. 572–585, 2021.

[21] Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré, "It's raw! audio generation with state-space models," *arXiv preprint arXiv:2202.09729*, 2022.

[22] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu, "On layer normalization in the transformer architecture," in *ICML*, 2020, pp. 10524–10533.

[23] Sunita Sarawagi and William W Cohen, "Semi-markov conditional random fields for information extraction," *Advances in neural information processing systems*, vol. 17, 2004.

[24] James Ferguson, Greg Durrett, and Dan Klein, "Disfluency detection with a semi-markov model and prosodic features," in *NAACL*, 2015, pp. 257–262.

[25] Liang Lu, Lingpeng Kong, Chris Dyer, Noah A Smith, and Steve Renals, "Segmental recurrent neural networks for end-to-end speech recognition," *arXiv preprint arXiv:1603.00223*, 2016.

[26] Yefei Chen, Heinrich Dinkel, Mengyue Wu, and Kai Yu, "Voice activity detection in the wild via weakly supervised sound event detection.," in *INTERSPEECH*, 2020, pp. 3665–3669.

[27] Elizabeth E Shriberg, "Phonetic consequences of speech disfluency," Tech. Rep., SRI INTERNATIONAL MENLO PARK CA, 1999.

[28] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, "wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019.

[29] Jimmy TH Smith, Andrew Warrington, and Scott W Linderman, "Simplified state space layers for sequence modeling," *arXiv preprint arXiv:2208.04933*, 2022.

[30] Herbert H Clark and Jean E Fox Tree, "Using uh and um in spontaneous speaking," *Cognition*, vol. 84, no. 1, pp. 73–111, 2002.