Interspeech 2024
1-5 September 2024, Kos, Greece

YOLO-Stutter: End-to-end Region-Wise Speech Dysfluency Detection

Xuanru Zhou[1], *Anshul Kashyap*[2], *Steve Li*[3], *Ayati Sharma*[2], *Brittany Morin*[4], *David Baquirin*[4], *Jet Vonk*[4], *Zoe Ezzes*[4], *Zachary Miller*[4], *Maria Tempini*[4], *Jiachen Lian*[2†], *Gopala Anumanchipalli*[2†]

[1] Zhejiang University, China [2] University of California, Berkeley
[3] Harvard University [4] University of California, San Francisco
xuanruzhou15@gmail.com, {jiachenlian, gopala}@berkeley.edu

Abstract

Dysfluent speech detection is the bottleneck for disordered speech analysis and spoken language learning. Current state-of-the-art models are governed by rule-based systems [1, 2] which lack efficiency and robustness, and are sensitive to template design. In this paper, we propose *YOLO-Stutter*: a *first end-to-end* method that detects dysfluencies in a time-accurate manner. YOLO-Stutter takes *imperfect speech-text alignment* as input, followed by a spatial feature aggregator, and a temporal dependency extractor to perform region-wise boundary and class predictions. We also introduce two dysfluency corpus, *VCTK-Stutter* and *VCTK-TTS*, that simulate natural spoken dysfluencies including repetition, block, missing, replacement, and prolongation. Our end-to-end method achieves *state-of-the-art performance* with a *minimum number of trainable parameters* for on both simulated data and real aphasia speech . Code and datasets are open-sourced at `https://github.com/rorizzz/YOLO-Stutter`

Index Terms: dysfluency, end-to-end, simulation, clinical

1. Introduction

Speech dysfluencies are defined as any form of speech that contains sound repetition, deletion, insertion, replacement, prolongation, block, etc. [1–4]. Dysfluency modeling holds significant promise in disordered speech screening [5, 6], and has a huge impact on the language learning market [7]. Dysfluency modeling is a speech transcription problem, which seems to be well tackled by recent large-scale developments [8–11]. However, large ASR models struggle with dysfluent speech for two reasons: (1) they transcribe speech into words or phonemes that lack dysfluency-related information, and (2) their strong language models often interpret dysfluent speech as fluent, e.g., transcribing "P-P-Please call Stella" as "Please call Stella" or "Praise call Stella". Progress in dysfluency modeling has been limited due to the lack of a well-defined problem formulation and high-quality annotated dysfluency data.

UDM [1] first formally defines *dysfluency modeling* as the detection of all types of dysfluencies at both word and phoneme levels, with accurate time region prediction. UDM transcribes dysfluent speech into unconstrained forced alignment and adopts *2D-Alignment* for detection task. H-UDM [2] developed a recursive 2D alignment algorithm that further boosts performance. These methods represent the de-facto pipeline at the time.

In this work, we tackle dysfluency modeling from a totally different perspective. Still following the dysfluency modeling criterion [1], we develop an *end-to-end model* which directly predicts dysfluencies and time regions from dysfluent speech and reference text input without any handcraft templates. To create training data, we introduce *VCTK-TTS* (7X larger than VCTK [12]), a synthetic dysfluency dataset created using the VITS [13], including repetition, missing, block, replacement, and prolongation at both the phoneme & word levels. VCTK-TTS offers a more natural representation of speech dysfluencies compared to VCTK++ [1], and the creation process is automated. In addition, we extend VCTK++ by incorporating word-level dysfluency and obtain a new dataset named *VCTK-Stutter* (5X larger than VCTK), thus achieving word-phoneme detection. Our newly proposed datasets have the potential to set a standard benchmark for studies in this field. For the dysfluency detection task, we drew inspiration from the YOLO [14] and devised a region-wise prediction scheme that captures both spatial and temporal information. We developed *YOLO-Stutter*, which takes soft speech-text alignments [13] as input, followed by a spatial feature aggregator and a temporal dependency extractor to directly predict dysfluency types and time regions. We also collaborated with clinical partners and obtained data from 38 Aphasia subjects. Results on simulated speech, public corpora, and Aphasia speech indicate that YOLO-Stutter achieves state-of-the-art performance even with a minimum number of trainable parameters.

2. Naturalistic Dysfluency Simulation

End-to-end modeling of speech dysfluency requires high-quality annotations. The prevalent approach is through simulation. Currently, existing simulated datasets [1, 15, 16] either do not incorporate time steps and broad coverage of dysfluencies or are not naturalistic. To develop a comprehensive and naturalistic simulated dysfluency corpus, we have taken two steps:

- We extended VCTK++ [1] by introducing rule-based word-level dysfluencies in *acoustic space*, creating a new dataset named **VCTK-stutter** [12]).
- We designed *TTS rules* and injected phoneme and word dysfluencies in *text space*. VITS [13] was used to generate naturalistic dysfluent speech. This dataset is named **VCTK-TTS**.

2.1. VCTK-Stutter

VCTK++ [1] was created in the acoustic space by editing audio based on forced alignment. However, it only focuses on phoneme-level dysfluencies. We extended VCTK++ by incorporating word-level dysfluencies (named *VCTK-Stutter*). Similar to obtaining phoneme-level alignments through MFA [17], in VCTK++ we employ WhisperX [18] to get word-level alignments, enabling us to establish precise insertion points for dysfluencies. Utilizing the word alignments generated by WhisperX, we annotate specific regions where artificial dysfluencies are to be introduced with the type of word-level dysfluency and the start and end times. We enable each word to have an equal

10.21437/Interspeech.2024-1855

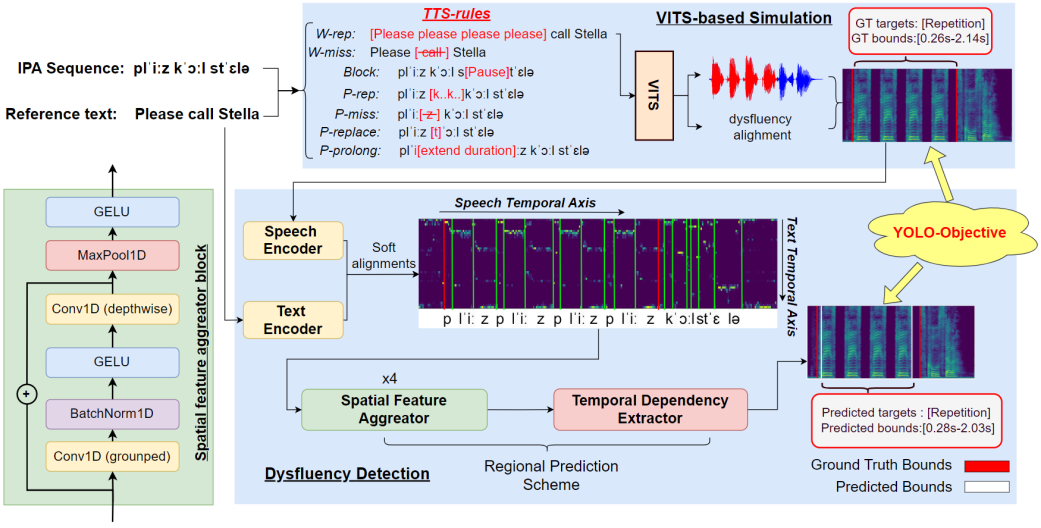

Figure 1: *YOLO-Stutter method's workflow: starting from reference text and its IPA sequence, by applying TTS-rules and VITS model, we generate the dysfluent speech along with its dysfluent alignment. Utilizing pretrained VITS speech and text encoders, we produce a soft-alignment matrix from the given reference text and mel spectrogram of dysfluent speech. Subsequently, the matrix is then processed through spatial feature aggregator and temporal dependency extractor, leading to predicted targets and bounds. The architecture of our spatial feature aggregator block is illustrated in the left green block. Here are examples of our simulated speech* `https://bit.ly/3PkKE8W`

probability of experiencing a dysfluency for consistency. The word-level dysfluency types we incorporate include:

- **Word-repetition**: This involves the repetition of entire words or phrases multiple times, with the count of repetitions varying randomly from 1 to 4, appended by a sample of silence of length 70% of the original word after each repetition.
- **Word-missing**: This pertains to the omission of certain words. The randomly chosen missing word is replaced with a silence of equivalent duration

2.2. VCTK-TTS

Traditional rule-based simulation methods [1, 15, 16] operate in acoustic space, and the generated samples are not naturalistic. We developed a new pipeline that simulates in text space. To achieve this, we first convert a sentence into an IPA phoneme sequence. Then, we develop TTS rules for phoneme editing to simulate dysfluency. These rules are applied to the entire VCTK dataset [13], allowing the voice of generated speech to vary from the 109 speakers included in the VCTK, thus enhancing the scalability of the dataset. We call this **VITS-based Simulation**. The entire pipeline is detailed in Sec.2.2.1, and TTS rules are discussed in Sec.2.2.2. Statistics are listed in Table. 1. Overall VCTK-Stutter/VCTK-TTS are 5X/7X larger than VCTK.

Table 1: *Types of Dysfluency Data in VCTK-Stutter & VCTK-TTS*

Dysfluency	# Samples VCTK-Stutter	Percentage VCTK-Stutter	# Samples VCTK-TTS	Percentage VCTK-TTS
Prolongation	43738	19.94	41084	13.37
Block	43959	20.040	49365	16.06
Replacement	0	0	46320	15.07
Repetition (Phoneme)	43738	19.94	40940	13.32
Repetition (Word)	43959	20.04	41800	13.60
Missing (Phoneme)	0	0	46320	15.07
Missing (Word)	43959	20.04	41559	13.52
Total Hours of Audio	304.66		426.93	

2.2.1. *Method pipeline*

VCTK-TTS pipelines can be divided into following steps: **(i) Dysfluency injection:** We first convert ground truth reference text of VCTK text [12] into IPA sequences via the VITS phonemizer [13], then add different types of dysfluencies at the phoneme level according to the *TTS rules*. **(ii) VITS inference:** We take dysfluency-injected IPA sequences as inputs, conduct the VITS inference procedure and obtain the dysfluent speech. **(iii) Annotation:** We retrieve phoneme alignments from VITS duration model, annotate the type of dysfluency on the dysfluent region.

2.2.2. *TTS rules*

We inject dysfluency in the text space following these rules:

- **Repetition** (phoneme&word): The first phoneme or syllable of a randomly picked word was repeated 2-4 times, with pauselengths varying between 0.5 to 2.0 seconds.
- **Missing** (phoneme&word): We simulated two phonological processes that characterize disordered speech [19] - weak syllable deletion (deletion of a random unstressed syllable based on stress markers[1]) and final consonant deletion.
- **Block**: A duration of silence between 0.5-2.0 seconds was inserted after a randomly chosen word in between the sentence.
- **Replacement** (phoneme): We simulated fronting, stopping, gliding, deaffrication - processes that characterize disordered speech [20] - by replacing a random phoneme with one that would mimic the phonological processes mentioned above.
- **Prolongation** (phoneme): The duration of a randomly selected phoneme in the utterance was extended by a factor randomly chosen between 10 to 15 times its original length, as determined by the duration model.

[1]https://github.com/timmahrt/pysle

2.3. Evaluation

To evaluate the rationality and naturalness of two datasets we constructed, we collected Mean Opinion Score (MOS, 1-5) ratings from 10 people. The final results are as displayed in Table. 2. VCTK-TTS was perceived to be far more natural than VCTK-Stutter (MOS of 4.13 compared to 2.22). We employ VCTK-Stutter as a baseline corpus.

Table 2: *MOS for VCTK-Stutter & VCTK-TTS Samples*

Dysfluency Type	VCTK-Stutter MOS	VCTK-TTS MOS
Block (word)	2.80 ± 0.63	3.40 ± 0.52
Missing (phoneme)	N/A	4.70 ± 0.48
Missing (word)	2.90 ± 0.32	4.80 ± 0.63
Prolong (phoneme)	N/A	3.80 ± 0.92
Repetition (phoneme)	1.40 ± 0.70	4.60 ± 0.84
Repetition (word)	2.80 ± 0.79	3.70 ± 1.16
Replacement (phoneme)	N/A	3.90 ± 0.99
Overall	2.22 ± 0.84	4.13 ± 0.56

3. End-to-End Dysfluency Detection

Dysfluency modeling is text-dependent [1]. We adopt the *soft speech-text alignment* from VITS [13] as input, and adopt a YOLO-like objective for the detection. We take 1D extension of 2D object detection from [14], which utilizes a region-wise prediction scheme designed to aggregate local spatial information, to develop our detector. The entire paradigm is shown in Fig. 1 and the corresponding modules are detailed in the following.

3.1. Soft speech-text alignments

VITS [13] takes in speech and text data to train alignment, producing a $|c_{text}| \times |z|$ monotonic attention matrix A that represents how each input phoneme expands to be time-aligned with target speech, where c_{text} represents the tokenized text dimension and z the temporal speech dimension. We use the soft alignments A and apply a softmax operation across the text dimension, computing the maximum attention value for each time step. We use pre-trained text and speech encoders from [13].

3.2. Spatial feature aggregator

To preserve local spatial features for region-wise predictions, we use learnable spatial feature aggregator blocks. Unlike spectrograms, soft alignment data cannot be effectively processed using traditional conformer [21] methods with pointwise and depthwise convolutions, as collapsing information across either the text or speech dimension would severely corrupt the relevant signal. Our initial experiments confirm that this design fails to learn dysfluencies. To address this, we employ a series of depthwise and grouped convolution operations, enabling iterative downsampling while preserving regional spatial features.

3.3. Temporal dependency extractor

We found that using only spatial feature aggregator modules is insufficient for accurately detecting dysfluencies due to their reliance on local spatial information. To effectively capture the sequential nature of dysfluent speech, we employ a second-stage Transformer encoder [22] that treats each speech region as an element in a sequence, using the text dimension as the embedding dimension. This Transformer encoder consists of 8 layers with 8 attention heads per layer, enabling the extraction of longer-term temporal information crucial for dysfluency detection.

3.4. YOLO objective

Following YOLO [14], we use a weighted loss-based multi-task training pipeline with three separate loss values: dysfluency presence confidence loss (binary cross-entropy), start and end bound loss (mean squared error), and dysfluency type categorical loss (cross-entropy). Confidence loss is computed across all regions, categorical loss in regions with dysfluency, and bound loss on the region "responsible" for the dysfluency. Bound values are normalized between 0-1 using fixed padded lengths as max bound values. Loss is shown below:

$$\mathbb{L} = \lambda_{\text{bound}} \frac{1}{S} \sum_{i=0}^{S} \mathbb{1}_{\text{obj}} [(b_{\text{start}} - \hat{b}_{\text{start}})^2 + (b_{\text{end}} - \hat{b}_{\text{end}})^2]$$
$$-\lambda_{\text{conf}} \frac{1}{S} \sum_{i=0}^{S} \hat{y}_i \log(p(y_i)) + (1 - \hat{y}_i) \cdot \log(1 - p(y_i))$$
$$-\lambda_{\text{class}} \frac{1}{S} \sum_{i=0}^{S} \sum_{j=0}^{n} c_n \log(p(\hat{c}_n))$$

where $\mathbb{1}_{obj}$ denotes the presence of a dysfluency in a specific region, y_i & $\hat{y}_i$ denote the predicted & target confidence values, and c & $\hat{c}$ denote the predictions & targets for n classes. Loss is computed and averaged across S regions within a single sample. We scale the bound loss term by a factor of 5 and the classification term by a factor of 0.5.

4. Experiments

4.1. Datasets

(1) VCTK [12] includes 109 native English speakers with accented speech. It is used in both *VCTK-Stutter* and *VCTK-TTS*. **(2) LibriStutter [15]** contains artificially stuttered speech and stutter classification labels for 5 stutter types. It was generated using 20 hours of audio selected from [23]. **(3) Aphasia Speech** is collected from our clinical collaborators, our dysfluent data comprises 38 participants diagnosed with Primary Progressive Aphasia (PPA), larger than the data used in [1, 2] which only has 3 speakers. People were asked to read grandfather passage, leading about 1 hour of speech. **(4) UCLASS [24]** contains recordings from 128 children and adults who stutter. Only 25 files have been annotated and did not annotate for the block class, we only used those files and did not use the block class for subsequent datasets. **(5) SEP-28K** is curated by [25], contains 28,177 clips extracted from publicly available podcasts. We removed files where the annotations were labelled as "unsure".

4.2. Training

The detector is trained with a randomized 90/10 train/test split on both VCTK-Stutter and VCTK-TTS, with a batch size of 64. We utilize the Adam optimization method with beta values (0.9, 0.999) and learning rate of 3e-4, opting not to use dropout or weight-decay in our training process. For VCTK-Stutter, the model is trained for 20 epochs and total of 39 hours on a RTX A6000. Meanwhile, for VCTK-TTS, the model is trained for 30 epochs and total of 70 hours on the RTX A6000.

4.3. Evaluation Metrics

(1) Accuracy is the accuracy of correct predictions of stutter types in regions that contain a dysfluency. The confidence prediction accuracy is defined as the number of correct predictions of dysfluent and fluent regions over all the regions. **(2) Bound loss** is the mean squared loss between the predicted bounds and the actual bounds of the dysfluent regions, normalized to the

Table 3: *Accuracy (Acc) and Bound loss (BL) of the five dysfluency types trained on the VCTK-Stutter and VCTK-TTS.*

Methods	Trainable parameters	Dataset	Rep		Block		Miss		Replace		Prolong	
			Acc.%	BL	Acc.%	BL	Acc.%	BL	Acc.%	BL	Acc.%	BL
H-UDM [2]	92M	VCTK-Stutter Testset	82.08	30ms	97.12	27ms	20	24ms	-	-	-	-
YOLO-Stutter(VCTK-Stutter)	33M	VCTK-Stutter Testset	**99.16**	**26ms**	99.29	25ms	**80.00**	18ms	-	-	**91.84**	35ms
YOLO-Stutter(VCTK-TTS)	33M	VCTK-Stutter Testset	83.11	27ms	**100**	**22ms**	40.00	**17ms**	-	-	90.34	**34ms**
H-UDM [2]	92M	VCTK-TTS Testset	74.66	68ms	88.44	85ms	15	100ms	-	-	-	-
YOLO-Stutter(VCTK-Stutter)	33M	VCTK-TTS Testset	78.31	66ms	92.44	**43ms**	43.33	42ms	-	-	88.17	42ms
YOLO-Stutter(VCTK-TTS)	33M	VCTK-TTS Testset	**98.78**	**27ms**	**98.71**	78ms	**70.00**	**8ms**	**73.33**	**10ms**	**93.74**	**32ms**

Table 4: *Type-specific accuracy (ACC) and time F1-score*

Methods	Dataset	Accuracy (%, ↑)			Time F1 (↑)
		Rep	*Prolong*	*Block*	
Kourkounakis et al. [15]	UCLASS	84.46	94.89	-	0
Jouaiti et al. [26]	UCLASS	89.60	99.40	-	0
Lian et al. [2]	UCLASS	75.18	-	50.09	0.700
YOLO-Stutter (VCTK-Stutter)	UCLASS	76.82	78.49	54.13	0.799
YOLO-Stutter (VCTK-TTS)	UCLASS	92.00	91.43	56.00	**0.893**
Kourkounakis et al. [15]	LibriStutter	82.24	92.14	-	0
Lian et al. [2]	LibriStutter	85.00	-	-	0.660
YOLO-Stutter (VCTK-Stutter)	LibriStutter	88.26	65.32	-	0.684
YOLO-Stutter (VCTK-TTS)	LibriStutter	89.71	67.74	-	**0.697**
Jouaiti et al. [26]	SEP-28K	78.70	93.00	-	0
Lian et al. [2]	SEP-28K	70.99	-	66.44	0.699
YOLO-Stutter (VCTK-Stutter)	SEP-28K	69.79	63.72	71.70	0.747
YOLO-Stutter (VCTK-TTS)	SEP-28K	82.01	89.19	68.09	**0.813**

Table 5: *Dysfluency evaluation on Aphasia speech.*

Methods	Ave. Acc. (%, ↑)	Best Acc.(%, ↑)	Ave. BL (ms, ↓)
H-UDM [2]	41.8	70.22	52ms
YOLO-Stutter(VCTK-Stutter)	45.71	73.08(Rep)	46ms
YOLO-Stutter(VCTK-TTS)	54.19	**92.54 (Block)**	**21ms**

1024-length padded spectrogram and converted to a time scale using a known 20ms sampling frequency. **(3) Time F1 [1]** measures accuracy in bounds prediction, calculating based on the intersection between predicted and actual bounds. Once there is an overlap, the sample is considered as a True Positive sample.

4.4. Validation

To assess the performance of trained detector, we conduct evaluations on the VCTK++ and VCTK-TTS testsets, as well as on the PPA data. The evaluation results, including type-specific detection accuracy and bound loss metrics, are presented in Table 3 for the training datasets and in Table 5 for the PPA data. To compare the performance with previous works, we also validated the model on UCLASS, Libristutter and SEP-28K, calculating type-specific accuracy as well as Time F1.

4.5. Results and Discussions

We conducted inference on our proposed VCTK-Stutter and VCTK-TTS test sets, using H-UDM as baseline. As indicated in Table 3, both YOLO-Stutter (VCTK-Stutter) and YOLO-Stutter (VCTK-TTS) surpassed H-UDM across all metrics. Notably, the results on VCTK-TTS from YOLO-Stutter (VCTK-TTS) were particularly promising. VCTK-TTS closely mimics human speech, affirming the potential of YOLO-Stutter (VCTK-TTS) to deliver substantial performance on actual human speech. Furthermore, we presented our findings on UCLASS, LibriStutter, and SEP-28K, as shown in Table 4. Given that the test sets from the original benchmarks are private, direct accuracy comparisons may not be entirely fair. Focusing more on time-aware detection, we reported the Time F1 score for each dataset, where all baselines, except for H-UDM, scored 0. Both of our proposed methods consistently outperformed H-UDM. Lastly, we evaluated our methods on actual PPA speech, with results detailed in Table 5. Both models consistently exceeded H-UDM's performance, with VCTK-TTS achieving additional gains owing to its higher naturalness, as reflected in Table 2. However, the average accuracy remains low, highlighting the challenge of perfectly capturing the real distribution of dysfluency.

4.6. Ablation experiments

To investigate the impact of the proportions of different dysfluency types quantities on training results, we selected three different proportions except for average on VCTK-TTS, as follows: P = [Rep, Block, Miss, Replace, Prolong], P_1=[0.9 : 1 : 1 : 1 : 1], P_2= [1 : 1 : 1.2 : 1: 1], P_3=[1 : 1 : 1.2 : 1.2 : 1]. Table 6 shows validated type-specific accuracy on VCTK-TTS inference testsets, from which we can see that although the proportions were adjusted, the type accuracy for repetition and block has remained relatively high. The accuracy for missing increases with its sample proportion, but does not improve as the proportions of other samples (repetition) decrease. Increasing missing and replace proportions improves missing type accuracy but lowers replacement accuracy.

Table 6: *Type-specific accuracy of different proportions*

Proportions	Rep	Block	Miss	Replace	Prolong
Average	98.78%	98.71%	70.00%	73.33%	93.74%
P_1	96.66%	99.97%	78.33%	66.67%	87.72%
P_2	96.67%	99.97%	60.00%	46.67%	93.33%
P_3	96.67%	99.97%	83.29%	54.32%	90.00%

5. Conclusion and Limitations

We introduce *YOLO-Stutter*, the first end-to-end dysfluency detection paradigm, achieving state-of-the-art performance with fewer trainable parameters compared to rule-based systems. We contribute two simulated dysfluent corpora, *VCTK-Stutter* and *VCTK-TTS*, with high-quality annotations covering sound repetition, replacement, insertion, deletion, and block. However, limitations persist. First, average accuracy on aphasia speech inference remains limited, indicating a significant gap between simulated and real dysfluency. Second, VCTK-TTS still produces robotic noise and discontinuous sounds, warranting exploration of adversarial methods to address these issues. Third, it is worth to explore simulation in articulatory space [27–31]. Lastly, dysfluency is a clinical problem and is speaker dependent. Disentangled analysis and synthesis [32–36] should be leveraged to tackle dysfluency in speaker-free space.

6. Acknowledgement

Thanks for support from UC Noyce Initiative, Society of Hellman Fellows, NIH/NIDCD and the Schwab Innovation fund.

7. References

[1] J. Lian, C. Feng, N. Farooqi, S. Li, A. Kashyap, C. J. Cho, P. Wu, R. Netzorg, T. Li, and G. K. Anumanchipalli, "Unconstrained dysfluency modeling for dysfluent speech transcription and detection," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8.

[2] J. Lian and G. Anumanchipalli, "Towards hierarchical spoken language dysfluency modeling," *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, 2024.

[3] J. Pálfy and J. Pospíchal, "Pattern search in dysfluent speech," in *2012 IEEE International Workshop on Machine Learning for Signal Processing*. IEEE, 2012, pp. 1–6.

[4] T. Kouzelis, G. Paraskevopoulos, A. Katsamanis, and V. Katsouros, "Weakly-supervised forced alignment of disfluent speech using phoneme-level modeling," *Interspeech*, 2023.

[5] M. C. Brady, H. Kelly, J. Godwin, P. Enderby, and P. Campbell, "Speech and language therapy for aphasia following stroke," *Cochrane database of systematic reviews*, no. 6, 2016.

[6] NIDCD, "Nidcd," https://www.nidcd.nih.gov/health/statistics/quick-statistics-voice-speech-language, 2016.

[7] VCL, "Vcl," https://vclenglish.com/54-8-billion-by-2025-

[8] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*. PMLR, 2023, pp. 28 492–28 518.

[9] J. Lian, A. Baevski, W.-N. Hsu, and M. Auli, "Av-data2vec: Self-supervised learning of audio-visual speech representations with contextualized target representations," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023.

[10] Y. Zhang, W. Han, J. Qin, Y. Wang, A. Bapna, Z. Chen, N. Chen, B. Li, V. Axelrod, G. Wang *et al.*, "Google usm: Scaling automatic speech recognition beyond 100 languages," *arXiv preprint arXiv:2303.01037*, 2023.

[11] V. Pratap, A. Tjandra, B. Shi, P. Tomasello, A. Babu, S. Kundu, A. Elkahky, Z. Ni, A. Vyas, M. Fazel-Zarandi *et al.*, "Scaling speech technology to 1,000+ languages," *arXiv preprint arXiv:2305.13516*, 2023.

[12] J. Yamagishi, C. Veaux, K. MacDonald *et al.*, "Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92)," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2019.

[13] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5530–5540.

[14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

[15] T. Kourkounakis, A. Hajavi, and A. Etemad, "Fluentnet: End-to-end detection of stuttered speech disfluencies with deep learning," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2986–2999, 2021.

[16] J. Harvill, M. Hasegawa-Johnson, and C. Yoo, "Frame-level stutter detection," in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2022, 2022, pp. 2843–2847.

[17] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi." in *Interspeech*, vol. 2017, 2017, pp. 498–502.

[18] M. Bain, J. Huh, T. Han, and A. Zisserman, "Whisperx: Time-accurate speech transcription of long-form audio," *Interspeech*, 2023.

[19] J. A. Bauman-Wangler, *Articulation and phonology in speech sound disorders a clinical focus*. Pearson Education, Inc., 2020.

[20] J. E. Bernthal, N. W. Bankson, and P. Flipsen, *Articulation and phonological disorders: Speech sound disorders in children*. Pearson, 2017.

[21] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," 2020.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[24] P. Howell, S. Davis, and J. Bartrip, "The uclass archive of stuttered speech," *Journal of Speech Language and Hearing Research*, vol. 52, p. 556, 2009. [Online]. Available: https://api.semanticscholar.org/CorpusID:141419086

[25] C. Lea, V. Mitra, A. Joshi, S. Kajarekar, and J. P. Bigham, "Sep-28k: A dataset for stuttering event detection from podcasts with people who stutter," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6798–6802.

[26] M. Jouaiti and K. Dautenhahn, "Dysfluency classification in stuttered speech using deep learning for real-time applications," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 6482–6486.

[27] J. Lian, A. W. Black, Y. Lu, L. Goldstein, S. Watanabe, and G. K. Anumanchipalli, "Articulatory representation learning via joint factor analysis and neural matrix factorization," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[28] J. Lian, A. W. Black, L. Goldstein, and G. K. Anumanchipalli, "Deep Neural Convolutive Matrix Factorization for Articulatory Representation Decomposition," in *Proc. Interspeech 2022*, 2022, pp. 4686–4690.

[29] P. Wu, L.-W. Chen, C. J. Cho, S. Watanabe, L. Goldstein, A. W. Black, and G. K. Anumanchipalli, "Speaker-independent acoustic-to-articulatory speech inversion," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[30] P. Wu, T. Li, Y. Lu, Y. Zhang, J. Lian, A. W. Black, L. Goldstein, S. Watanabe, and G. K. Anumanchipalli, "Deep Speech Synthesis from MRI-Based Articulatory Representations," in *Proc. INTERSPEECH 2023*, 2023, pp. 5132–5136.

[31] C. J. Cho, A. Mohamed, A. W. Black, and G. K. Anumanchipalli, "Self-supervised models of speech infer universal articulatory kinematics," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 061–12 065.

[32] J. Lian, C. Zhang, G. K. Anumanchipalli, and D. Yu, "Unsupervised tts acoustic modeling for tts with conditional disentangled sequential vae," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 2548–2557, 2023.

[33] K. Qian, Y. Zhang, H. Gao, J. Ni, C.-I. Lai, D. Cox, M. Hasegawa-Johnson, and S. Chang, "Contentvec: An improved self-supervised speech representation by disentangling speakers," in *ICML*, 2022.

[34] J. Lian, C. Zhang, and D. Yu, "Robust disentangled variational speech representation learning for zero-shot voice conversion," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022.

[35] J. Lian, C. Zhang, G. K. Anumanchipalli, and D. Yu, "Towards Improved Zero-shot Voice Conversion with Conditional DSVAE," in *Proc. Interspeech 2022*, 2022.

[36] H.-S. Choi, J. Yang, J. Lee, and H. Kim, "Nansy++: Unified voice synthesis with neural analysis and synthesis," *ICLR*, 2022.