INCREASE APPARENT PUBLIC SPEAKING FLUENCY BY SPEECH AUGMENTATION

Sagnik Das *Nisha Gandhi* *Tejas Naik* *Roy Shilkrot*

Human Interaction Lab, Stony Brook University
Department of Computer Science
Stony Brook, NY, USA

ABSTRACT

Fluent and confident speech is desirable to every speaker. But professional speech delivering requires a great deal of experience and practice. In this paper, we propose a speech stream manipulation system which can help non-professional speakers to produce fluent, professional-like speech content, in turn contributing towards better listener engagement and comprehension. We propose to achieve this task by manipulating the disfluencies in human speech, like the sounds *uh* and *um*, the filler words and awkward long silences. Given any unrehearsed speech we segment and silence the filled pauses and doctor the duration of imposed silence as well as other long pauses (*disfluent*) by a predictive model learned using professional speech dataset. Finally, we output a audio stream in which speaker sounds more fluent, confident and practiced compared to the original recorded speech. According to our quantitative evaluation, we significantly increase the fluency of speech by reducing rate of pauses and fillers.

Index Terms— Speech disfluency detection, Speech disfluency repair, Speech Processing, Assistive technologies in speech

1. INTRODUCTION

Professional speakers, who make a living from their speech, speak clearly and fluently with very few repetitions and revisions. This kind of error-free utterances is the result of many hours of deliberate practice. On the other hand, a regular unrehearsed speaker generally speaks with no real practice of articulation and delivery. Naturally, words of an unrehearsed speech contain unintentional *disfluencies* interrupting the flow of the speech. Speech disfluency generally comes in the form of long pauses, discourse markers[1], repeated words, phrases or sentences and fillers or filled pauses like *uh* and *um*. According to Tree [1] approximately 6% of the speech appears to be non-pause disfluency. Filled pauses or filler words are the most common disfluency in any unrehearsed, impromptu speech [2].

Considering the diverse factors affecting speaker fluency, our proposition is to doctor a speech to make it appear fluent by masking the factors contributing to disfluency. We hereby propose a system to detect, segment, and remove the most common disfluencies, namely filler words and long, unnatural pauses from a speech to aid speakers' apparent fluency. Our system takes raw speech as input and outputs a modified fluent version of it by automatically removing the filled pauses and adjusting "long" silences.

We interpret the occurrence of disfluencies in a speech as an acoustic event, and a segmentation approach is taken for the detection. A combined convolutional-recurrent neural network (CRNN) architecture is used to achieve the task, inspired by Cakir et al [3]. Further, a binary classification approach is taken to detect long pauses between words. After deleting the filler-words and adjusting the silences the fluent version of the speech is obtained. The performance of our system is evaluated on speeches of non-native speakers of English using fluency metrics proposed by [4]. We also propose an assistive user interface which can be used to help users' to visualize and comparatively analyze their speech. The essential contributions of this paper are: 1) *A disfluency detection mechanism that works directly on acoustic features without using any language features*; 2) *A silence modeling scheme directly conditioned on the previous speech segment*; 3) *A disfluency repair technique to help users improve a pre-delivered speech.*

2. RELATED WORKS

In recent years, there have been many works related to speech disfluencies, spanned across the domains of psychology, linguistics and natural language processing (NLP). The prime motivation for disfluency detection in NLP is to better interpret the speech-to-text transcripts for natural language understanding systems.

Charniak et al's initial work [5] focused on classifying the edit words (restarts, repairs) from the transcription text using a boosted classifier. More contemporary methods applied a noisy channel model to detect and correct speech disfluencies [6, 7, 8]. Later, Hidden Markov Model (HMM), Conditional Random Field (CRF), Integer Linear Program-

Examples: https://sagniklp.github.io/pub-speaker-aug/

[1]Words used for organization or connection: "So, ...", or "Well, ..."

ming (ILP) based [9, 10] methods were introduced to tackle the same goal. Recent methods define it as a joint task of parsing and detecting disfluency [11, 12]. Even with convincing results, these methods are limited to pre-defined feature templates (lexical, acoustic, and prosodic). With advances in deep learning, most recent methods rely on recurrent neural networks (RNN) [13, 14] using word embeddings and acoustic features instead of pre-defined feature templates.

All the techniques above, make an assumption of having an automatic speech recognizer (ASR) in the pipeline and work at the transcript level. Also, these systems have never been paired with an acoustic level repair scheme with a goal of exploring the use-cases from the perspective of the listener and the speaker.

In our work, we address these motivations by devising a disfluency detection relying solely on acoustic features, combined with a repair method to synthesize temporally fluent speech segments.

3. PROPOSED METHOD

3.1. Disfluency Detection

Our work focuses on building a system that can be used not only as a disfluency detection system but also provide a way to understand users' disfluency better. The primary motivations of this work are the following-

- Detect disfluencies without relying on transcript.
- Consider long pauses as disfluency. On acoustic level, this is a big factor of the speakers' fluency.
- Repairing disfluent segments to help users create better speech.

The types of disfluencies we considered in this work, are the use of filler words, and intermittent long pauses.

3.1.1. Dataset

The dataset used for filler word segmentation is obtained from Switchboard [2] and Automanner[3] [15] transcriptions. To label disfluent silences we use combination of a silence probability model [16] and a disfluency detection model [17]. For each word pair utterance the silence probability model gives a probability of a silence (P_{sil}) occurring between them. A word pair with low P_{sil} but a significant amount of silence is labeled as disfluent. If a word pair doesn't exist the model vocabulary, we resort to the following approach. General disfluencies accompany longer silences, therefore any silence within a disfluent segment (detected by [17]) is labeled as an unnatural pause. Additionally, the word pairs surrounded with silences more than 0.7 seconds are also labeled similarly. This choice is experimental and can be considered safe because it's considerably higher than the suggested quantitative measure of micro-pauses (*fluent*), 0.2s [18]. On the other hand, additional fluent pairs are collected from TIMIT [19].

[2]https://www.isip.piconepress.com/projects/switchboard/

[3]https://www.cs.rochester.edu/hci/currentprojects.php?proj=automanner

3.1.2. Features

In this step, frame level acoustic features (log mel band energy or mel frequency cepstral coefficients (MFCCs)) are obtained at each timestep t resulting a feature vector $\boldsymbol{m}_t \in \mathcal{R}^C$. Here, C is the number of features (in frequency dimension) at frame t. The task of segmenting the filler words is formulated as binary classification of each frame to its correct class k (Eq. 1).

$$\arg\max_k \; P(y_t^{(k)} \mid \boldsymbol{m}_t, \boldsymbol{\theta}) \tag{1}$$

Where, $k = \{1, 2\}$ and $\boldsymbol{\theta}$ are the parameters of the classifier. In the training data, the target class $y_t^{(k)} = 1$ if frame t belongs to class k (determined using the onset/offset timeline of k associated with a sound segment) , otherwise zero.

3.1.3. CRNN for filler word segmentation

We propose a Convolutional Recurrent Neural Network (CRNN) for filler word segmentation. A similar architecture is previously used for sound event detection (SED) [3] and speech-recognition [20] task. The architecture is a combination of convolutional and recurrent layers, followed by feed-forward layers.

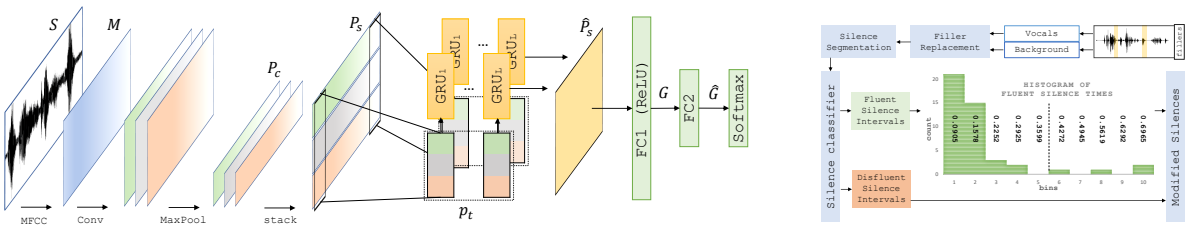
The sequence of extracted features $\boldsymbol{M} \in \mathcal{R}^{C \times T}$ is fed to the CNN layers. Then Max-pooling is applied over the frequency dimension. Output of max-pooling is a tensor $\mathcal{P}_c \in \mathcal{R}^{F \times M' \times T}$. Where, F is the number of filters of the final convolution layer, M' is the truncated frequency dimension after the max-pooling operation.

To learn the features over time axis, F feature maps are then stacked along the frequency axis- $\mathcal{P}_s \in \mathcal{R}^{(F \times M') \times T}$. This is fed to the RNN as a sequence of frames $\boldsymbol{p}_t$ which outputs a hidden vector $\hat{\boldsymbol{p}}_t$. The ith recurrent layer output is given as in Eq. 2. Where, $\mathcal{F}$ is a function learned by the each RNN unit. In this work, we use GRUs as presented in [21].

$$\hat{\boldsymbol{p}}_t^i = \mathcal{F}(\hat{\boldsymbol{p}}_t^{i-1}, \hat{\boldsymbol{p}}_{t-1}^i) \tag{2}$$

Final RNN outputs $\hat{\boldsymbol{p}}_t^f$ are then fed to a fully-connected (FC) layer with ReLU activation and $\mathcal{G} \in \mathcal{R}^{FC_1 \times T}$ is obtained where, FC_1 is the number of neurons of the layer. Finally, another layer with softmax activation is applied to get the class probabilities. Although classification is on frame level, the longer context is preserved by the recurrent structure of the network. The CRNN training objective is to minimize the cross-entropy loss with l_2 regularization (Eq. 3)

$$L(\boldsymbol{\theta}) = -\sum_{0:t} \sum_k \log P(y_t^{(k)}) + \lambda ||\boldsymbol{\theta}|| \tag{3}$$



(a) (b)

Fig. 1. (a) Block diagram of the filler-word segmentation; (b) Silence modification pipeline: The dashed line on the histogram shows the median time of the fluent silences

3.1.4. Disfluent silence Classification

The problem is formulated as a binary classification task, given a silent segment Z, the task is to decide whether it's a disfluent or a non-disfluent silence. Classifying a silence only makes sense when it's combined with adjacent utterances. Because an occurrence of silence is apparently driven by the utterance and also heavily influenced by disfluencies. Thus, it's not always evident that all pauses higher than a significant threshold is disfluent.

We train a binary classifier to achieve this task. Given a silent segment Z, it's first padded with the one-word utterances on the left and right ($\hat{Z}$). Then, the MFCC features are extracted and we take the mean over the frequency bands to create the feature vector $\boldsymbol{z}_i \in \mathcal{R}^T$. T is the number of frames in $\boldsymbol{z}_i$. Segments are of variable length thus $\boldsymbol{z}_i$ padded with trailing zeros prior the classification. In test time instead of the previous and next word boundaries, a fixed length time window is used. In our experiments, we found that $0.8 - 1.0$ secs. give satisfying results.

3.2. Disfluency Repair

To remove the fillers, simplest case is to silence the predicted segments. However, it's often helpful (such as when ambient noise is present) to use a decomposition mechanism [22] to separate the background noise and vocals first. Then, use the background segments as replacement of the fillers. On the modified track the silences are then segmented (Z) and finally, the classification is done. Now, the silent segment lengths are modified to make the speech fluent (Fig. 1b). The goal is to reduce the amount of long, unnatural pauses that hurt the fluency of the speech. It is also required to keep the pace of the speech intact. Too much reduction of silences makes the speech unnatural and broken. We take the fluent silence times (i.e., suggested by our silence classifier) and obtain a histogram. We experimentally found that taking the median of the histogram bins as the optimal amount of silence works quite well. In this way, the distribution of the silence along the speech progression confines to a measure and speaker sounds more fluent in the modified speech.

Features	CNN	RNN	FC
log mel	conv1 [32,(8,8)], conv2 [64,(4,4)] maxpool1 [8,4], maxpool2[4,2] dr=0.25	l=3 d=128	d=100 dr=0.5

Table 1. Final CNN, RNN and FC layer paramerters from our best model

4. RESULTS & ANALYSIS

4.1. Experimental Settings

4.1.1. Datasets

The experiments are performed on Switchboard [23], Automanner [15] and our dataset of public speaker recording. To train the CRNN we use the segments from the Switchboard. The CRNN test results are reported on held-out data from Switchboard-I. Silence classification results are reported on TIMIT [19], Switchboard, and Automanner held-out dataset. All the fluency metrics are evaluated on our dataset, containing recordings of 20 non-native speakers of English. The speakers were asked to talk on a specific topic for 50-60 seconds.

4.1.2. Parameter settings

We experimented with different configurations of the CNN and RNN parameters and different features.

Types of features: The *mfcc* ($40 \times t$) and *log mel* ($128 \times t$) features are used for filler segmentation. For the silence classification *mfcc* features are used. All features are extracted in 30ms frames with 15ms overlap.

RNN & CRNN parameters: In experiments with the CNN and CRNN, convolutional (*conv*) layers are used with combination of max pooling and average pooling. At each layer, ReLU activation is used. The RNN we use consists Gated Recurrent Units (GRU) with no intermediate dropouts (dr). The network is trained in an end-to-end fashion using AdaGrad algorithm for 200 epochs. The learning rate was set to 0.001. The regularization constant (λ) was set to 0.01. The best model parameters are given in table 1.

Metrics →	*SR* ↑	*AR* ↑	PTR ↑	MLR ↑	MLP ↓	FPM ↓
Original	191.456	198.155	66.717	0.420	0.789	4.739
Ours	206.465	208.437	**77.151**	0.495	**0.422**	1.813
Ours+ASR	**206.710**	**208.770**	76.974	**0.504**	0.438	**1.608**

Table 2. The fluency metrics before and after processing the speeches. ↑ means higher and ↓ denotes lower is better.

Silence classification parameters: Max length of the sequences were set to 128. Final parameters of our best classifier (XGBoost) are- depth=3, lr=0.1 and estimators=100.

Features	*Precision*	*Recall*	F_1
mfcc	0.9482	0.9610	0.9534
log mel	**0.9495**	**0.9629**	**0.9550**

Table 3. Performance of the CRNN with different features.

Method	*Precision*	*Recall*	F_1
ASR	**0.9774**	**0.9792**	**0.9775**
CRNN	0.9495	0.9629	0.9550

Table 4. Performance of filler word segmentation compared to an automatic speech recognizer.

4.1.3. Evaluation Metrics

To evaluate the filler word segmentation we use the F_1 score calculated at frame level (30ms) taking fillers as the positive class. The silence classification is evaluated using the F_1 score w.r.t. the disfluent silence class.

To evaluate the quality of the augmented speech from our system, we use six metrics defined in [4]. These are- Speech rate (SR), Articulation rate (AR), Phonation-time ratio (PTR), Mean length of runs (MLR), Mean length of pauses (MLP) and Filled pauses per min. (FPM).

4.2. Filler Word Segmentation

The filler word segmentation evaluation results are given in table 3 and 4. In Table 3 we report the comparative performance of the CRNN using different features. To understand more about the credibility of the CRNN, in table 4 we show the results compared to an automatic speech recognizer available with Kaldi (ASpIRE Chain Model[4]). Considering the simplicity of our network, it performs pretty close to the ASR in terms of F_1 score. All results are evaluated on a subset of Switchboard-I dataset.

Method →	**SVM**	**LogReg**	**XGBoost**
F_1	0.9055	0.9200	**0.9207**

Table 5. Silence classification performance on TIMIT, SwitchBoard and Automanner

[4]https://github.com/kaldi-asr/kaldi/tree/master/egs/aspire

The only drawback that we have observed while comparing our method and ASR is that, sometimes our classifier detects segments that sounds similar with 'uh' or 'um'.

4.3. Disfluent Silence Classification

For this task we experimented with SVM, Logistic Regression (LogReg) and XGBoost. The results are summarized in table 5. We used 10-fold cross validation to report our results.

4.4. Disfluency Repair

To compare the fluency of the synthesized and the original speech, mean of each metric (Section 4.1.3) across all the samples are reported in table 2. *Our method+ASR* gives slightly better result in most cases. But, the metrics are almost same proving the credibility of our disfluency detection method. Apart from the numbers, for qualitative understanding, some processed samples are available **here**.

5. FUTURE WORK

To the extent of the types of disfluencies produced in a speech, this work is a small step towards a bigger goal. Along with the pitfalls of our method following could be the future directions of this work- 1) *Devising techniques to segment other kinds of common disfluencies (repetition, discourse markers, corrections) and speech impairments (stuttering*); 2) *Creating a dynamic and online repair scheme, by generating necessary (disfluent) portions of speech, instead of replacing.*

6. CONCLUSION

In this work, we interpret disfluency detection from speakers' perspective thus going beyond just helping intelligent systems. We introduce an additional component of repairing the disfluencies. Consequently, we tried to work solely on the acoustic domain, diminishing a need for a complex system like an ASR, before performing disfluency detection. With the results of our detection and repair scheme, we show improved fluency in speakers' dialogues, given a less-fluent speech. To the best of our knowledge, this is the first work related to disfluency repair for the sake of users' and can be further extended to assist users with speech impairments and other general disfluencies.

7. ACKNOWLEDGEMENTS

We would like to thank the participating speakers for their speeches and anonymous reviewers for their valuable comments. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp and P6000 GPU used for this research.

8. REFERENCES

[1] Jean E Fox Tree, "The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech," *Journal of memory and language*, vol. 34, no. 6, pp. 709–738, 1995.

[2] Kathryn Womack, Wilson McCoy, Cecilia Ovesdotter Alm, Cara Calvelli, Jeff B Pelz, Pengcheng Shi, and Anne Haake, "Disfluencies as extra-propositional indicators of cognitive processing," in *Proceedings of the workshop on extra-propositional aspects of meaning in computational linguistics*. Association for Computational Linguistics, 2012, pp. 1–9.

[3] Emre Cakır, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *arXiv preprint arXiv:1702.06286*, 2017.

[4] Judit Kormos and Mariann Dénes, "Exploring measures and perceptions of fluency in the speech of second language learners," *System*, vol. 32, no. 2, pp. 145–164, 2004.

[5] Eugene Charniak and Mark Johnson, "Edit detection and parsing for transcribed speech," in *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*. Association for Computational Linguistics, 2001, pp. 1–9.

[6] Matthias Honal and Tanja Schultz, "Correction of disfluencies in spontaneous speech using a noisy-channel approach," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[7] Mark Johnson and Eugene Charniak, "A tag-based noisy-channel model of speech repairs," in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, 2004.

[8] Simon Zwarts, Mark Johnson, and Robert Dale, "Detecting speech repairs incrementally using a noisy channel approach," in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 1371–1378.

[9] Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Dustin Hillard, Mari Ostendorf, and Mary Harper, "Enriching speech recognition with automatic detection of sentence boundaries and disfluencies," *IEEE Transactions on audio, speech, and language processing*, vol. 14, no. 5, pp. 1526–1540, 2006.

[10] Kallirroi Georgila, "Using integer linear programming for detecting speech disfluencies," in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. Association for Computational Linguistics, 2009, pp. 109–112.

[11] Mohammad Sadegh Rasooli and Joel Tetreault, "Joint parsing and disfluency detection in linear time," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 124–129.

[12] Matthew Honnibal and Mark Johnson, "Joint incremental disfluency detection and dependency parsing," *Transactions of the Association of Computational Linguistics*, vol. 2, no. 1, pp. 131–142, 2014.

[13] Julian Hough and David Schlangen, "Joint, incremental disfluency detection and utterance segmentation from speech," in *Proceedings of the Annual Meeting of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[14] Shaolei Wang, Wanxiang Che, Yue Zhang, Meishan Zhang, and Ting Liu, "Transition-based disfluency detection using lstms," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 2785–2794.

[15] M Iftekhar Tanveer, Ru Zhao, Kezhen Chen, Zoe Tiet, and Mohammed Ehsan Hoque, "Automanner: An automated interface for making public speakers aware of their mannerisms," in *Proceedings of the 21st International Conference on Intelligent User Interfaces*. ACM, 2016, pp. 385–396.

[16] Guoguo Chen, Hainan Xu, Minhua Wu, Daniel Povey, and Sanjeev Khudanpur, "Pronunciation and silence probability modeling for asr," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[17] Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi, "Disfluency detection using a bidirectional lstm," *arXiv preprint arXiv:1604.03209*, 2016.

[18] Heidi Riggenbach, "Toward an understanding of fluency: A microanalysis of nonnative speaker conversations," *Discourse processes*, vol. 14, no. 4, pp. 423–441, 1991.

[19] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon technical report n*, vol. 93, 1993.

[20] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4580–4584.

[21] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[22] Zafar Rafii and Bryan Pardo, "Music/voice separation using the similarity matrix.," in *ISMIR*, 2012, pp. 583–588.

[23] John J Godfrey, Edward C Holliman, and Jane McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*. IEEE, 1992, vol. 1, pp. 517–520.