Naturalization of Text: Inserting Disfluencies

Parth Vipul Shah, Ryan Luu, Bowman Brown, Ira Deshmukh, Alfianto Widodo
Department of Computer Science, University of Southern California,
941 Bloom Walk, Los Angeles, CA 90089 , USA

Abstract

In this work, we have successfully implemented methods to transform raw text documents of spoken dialogue/speech into its more human natural-sounding version by augmenting these documents with disfluencies. We present two such methods to naturalize text. The first method is a bigram approach that uses the frequency of the occurrences of desired bigrams in the training data as a basis to insert filler words or pauses in the given input sentences. The second method uses a transformer to learn the most probable insert locations and disfluencies. The performance of each model was then measured using two automated scoring systems based off similar sentence score and similar insertion score.

1 Introduction

A major factor that discriminates spontaneous speech from written text is the presence of paralinguistic features such as filled pauses (fillers), false starts, laughter, disfluencies and discourse markers that are beyond the framework of formal grammars. There are multiple ways of achieving natural speech, but the focus of our research are speech disfluencies.

A speech disfluency is any of various breaks, irregularities, or non-lexical vocables which occur within the flow of otherwise fluent speech. This can include filler grunts and noises such as ”uh” or ”um”. Speakers use ”uh” and ”um” to announce that they are initiating what they expect to be a minor or major delays in speaking. A speaker can use these announcements in turn to implicate that they are searching for a word, are deciding what to say next, want to keep the floor, or want to cede the floor. Understanding these cues and their impact on the context of a given conversation can be said to be considered a natural part of human dialogue.

We aim to transform raw text documents of spoken dialogue/speech into its most natural-sounding version by augmenting these documents with disfluencies. Using various NLP techniques such as bigrams and context analysis, we aim to create a model that can detect the most appropriate places in the document to insert these disfluencies and which form of it. This model would fit into any existing NLP pipeline. Upstream tasks could be question answering systems and downstream tasks could be Text-To-Speech synthesizers. Additionally, a requirement for this module would be fast transformation. This module must refrain from inserting additional latency, slowing down the entire pipeline. A successful implementation of the ideas examined in this proposal can aid in the efforts of naturalizing speech synthesis.

2 Related Work

Recent innovations in natural language processing have significantly improved the ability for artificial intelligence applications to both process and create structurally and logically correct text. While generated text may read naturally, there is a definitive gap when it comes to natural sounding artificial speech. Approaches to naturalizing any kind of generated language are few and far between, despite users deeming less artificial sounding voices as better (Kühne K, Fischer MH, Zhou Y., 2020) and more intelligible (Pisoni DB, Manous LM, Dedina MJ, 1987).

Contrary to this papers proposal to introduce disfluencies, current approaches to creating natural sounding speech attempts to generate speech waveforms directly instead of adding additional natural sounding elements to text that can be processed (Tan et. al., 2022) (Kong et. al., 2020) (Li et. al., 2021) Furthermore, many of the works involving disfluencies and natural human speech patterns is centered around removing disfluencies and naturalizations in order to better process input speech (Wang et. al., 2022) as challenges may arise when attempting to process disfluencies downstream. Other attempts to naturalize synthetic voices involve rephrasing speech commands (Einolghozati et. al., 2020) or generating the rhythms that are common in natural human speech (Kharitonov et. al., 2022) and (Lee et. al., 2021)

There is very little work that involves including disfluencies as an efficient means of naturalizing text. A recent example, LARD (Passali et. al., 2022) allows for the addition of “repetitions, replacements and restarts” to the data for simple disfluency generation, but is intended to modify sets of training data in order to allow for better disfluency detection (for later removal). (Wester et. al., 2015) takes a psychological approach and works to add disfluencies as a way to measure perceptions of synthetic voices. Our method is efficient and makes use of a psychological understanding of disfluency generation (Bakti et. al., 2009) in human speaking in order to create a dialogue script that can efficiently allow lower quality synthetic voices to sound more human. Current state-of-the-art Text-To-Speech synthesizers do not render disfluencies in the most natural way.

3 Method

3.1 Preprocessing

Since our end goal is to transform a given sentence into a more natural sounding one for potential use in existing NLP pipelines, we ended up basing our choice of disfluencies on parameters accepted by the SSML (Speech Synthesis Markup Language) of modern text to speech technologies. Towards that end, our research also identified the (Santa Barbara Corpus,), a corpus containing hundreds of audio transcriptions down to the level of intonation units, as the most relevant dataset suiting that purpose.

Since every corpus transcribes its data differently, we had the challenge of having to both create a specific mapping from our desired disfluencies to the specific annotations provided by the Santa Barbara corpus while also having to remove extra unnecessary annotations that would have contaminated our desired sentence. This task was performed primarily through the usage of regular expressions and formed the basis of our training dataset.

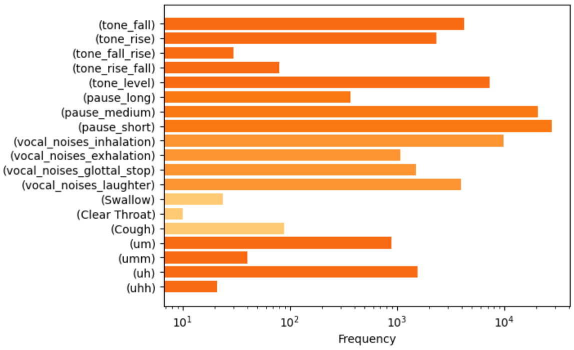

Figure 1: List of most frequently occuring tags identified within the Santa Barabara corpus

3.2 Model Design

We trained two types of models to insert disfluencies into text data. Our first model uses bigram probabilities to predict and insert the disfluencies. Our second model used a transformer that was fine-tuned on our training data to predict and insert the disfluencies. Despite using a more advanced architecture, the transformer-based model performed worse than the bigram model during disfluency selection, and was subsequently discarded from evaluation. The results produced during execution and reported in Section 5.1 are all products of the bigram model architecture.

Bigram Model The first model uses bigrams to augment input sentences with the most probable insert locations and disfluencies. During the model training process, the training corpus text is split into bigrams of two elements. For each bigram that contains a disfluency as either the first element or the second element, the model saves the probabilities of the bigram (word-disfluency or disfluency-word) pairing as well as the probabilities of the parts of speech of the word before and the word after the disfluency. To augment a sentence with disfluencies, The model takes in an input sentence and a ‘modifier’ parameter that is used to scale the number of disfluencies added to the input sentence. The ‘modifier’ parameter allows the model to successfully augment an input sentence with disfluencies even when the training data includes significantly more or significantly fewer than expected disfluencies but does require tuning for each training corpus. During sentence augmentation, the model selects the most likely locations for disfluencies based on the learned part-of-speech associations. For each selected location, the bigram probabilities are used to select the most likely disfluency, based on the word before and the word after the selected disfluency location. The model performs filtering to ensure that only one disfluency is allowed between words to keep the augmentation more natural sounding.

Transformer Model The second model uses a transformer to learn the most probable insert locations and disfluencies. Use of a transformer architecture was motivated by the expectation that additional context from the entire sentence, rather than just the tokens before and after the disfluency, would improve results. The transformer model was based on the Huggingface GPT-2 'small' language model. During the training process the training corpus was tokenized using the Huggingface GPT2TokenizerFast class. Custom tokens were added to cover the start and end elements of each sentence, as well as the individual disfluencies in the training corpus. The model was trained with 100 epochs with a learning rate of 0.005 and a batch size of 16 using an AdamW optimizer. To augment a sentence with disfluencies, the model takes in an input sentence and attempts to predict the next token in an auto-regressive manner. At each auto-regressive step, the model took in the abbreviated input and produced output token probabilities for the next token. The model then probabilistically selected the next token. If the selected token was a disfluency, then the disfluency was added to the abbreviated sentence. Otherwise, no disfluency was added. Finally, the next word from the raw sentence was added to the abbreviated sentence. This mechanism ensures that the output sentence matched the input sentence except for the inserted disfluencies.

4 Experimental Setup and Evaluation

The subject of this study is to produce text that sounds natural. Therefore, the ideal method to measure the model's performance would be to conduct a survey asking the participants to evaluate whether the output sentences sound more natural than its unannotated counterpart. However, the project's scope limitation does not allow for this. Human evaluation is time consuming and expensive. We decided to design and employ two automated scoring systems - similar sentence scoring and similar insertion scoring.

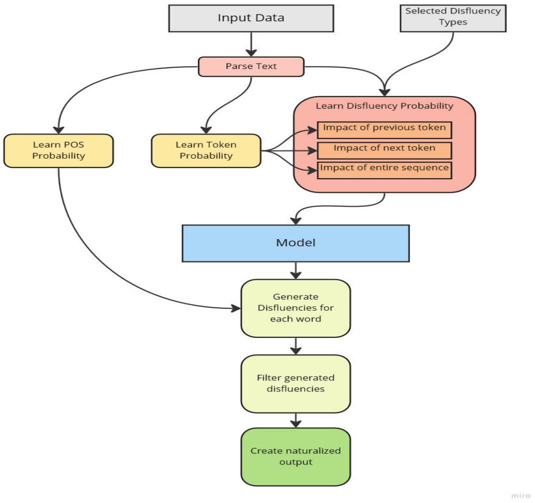
Input Data
Selected Disfluency Types
Parse Text
Learn Disfluency Probability
Impact of previous token
Impact of next token
Impact of entire sequence
Learn POS Probability
Learn Token Probability
Model
Generate Disfluencies for each word
Filter generated disfluencies
Create naturalized output
miro

Figure 2: An overview of our disfluency-generation architecture.

4.1 Similar Sentence Score

Similar sentence score is calculated for every output sentence. The sBERT (Reimers et. al., 2019)(Reimers et. al.) embeddings are used on the entire corpus following which a sentence most similar to the output sentence is retrieved by selecting the pair with maximum cosine similarity.

Test sentence: It's just too bad.
Most similar corpus sentence: It's too long.
Test sentence with insertions:
(pause_medium) It's just (vocal_noises) too long.
Corpus sentence with insertions:
(pause_short) It's too (pause_short) bad.

The sentences are POS-tagged using a pre-trained tagger in Python's NLTK library and split into bigrams. If a pair of bigrams have matching disfluencies in the same position and the other token in the bigram have matching POS tags, we count that as a match and the score is incremented by 1. This will give us the sentence score of that particular sentence. The sum of all sentences' scores normalized over the number of sentences gives us the overall score.

We implemented two modes of evaluation for similar sentence scoring; hard-matching and soft-matching. In hard-matching, the disfluencies match only if the two tags are an exact match. For example, (pause_short) would only match with (pause_short) and not (pause_long). However, in soft-matching, the disfluencies match

Test Bigram	Corpus Bigram	Score
(pause, J') , (it's, NN)	**(pause, JJ)**, (it's, NN)	1
(it's, NN), (just, RB)	(it's, NN), (too, RB)	0
(just, RB), (inhale, VB)	(too, RB), (pause, RB)	0
(inhale, VB), (too, RB)	(pause, RB), (long, RB)	0
(too, RB), (bad, JJ)	-	0
Sentence Score		1

Table 1: An example detailing the similar sentence scoring evaluation technique.

as long as the they are both in the same class. For example, (pause_short) would match with (pause_long) since they belong to the *'Pause'* class of disfluencies.

4.2 Similar Insertion Score

To calculate similar insertion score, the evaluation script scans through the sentence for insertions. For each insertion, the corpus is searched for similar insertions and compares the POS tags of the token(s) that surround the insertions. For each token where the position of that token relative to the insertion matches and their POS tag also matches, we add 1 point to the total score. The total score is then normalized over the number of sentences, giving us the overall score.

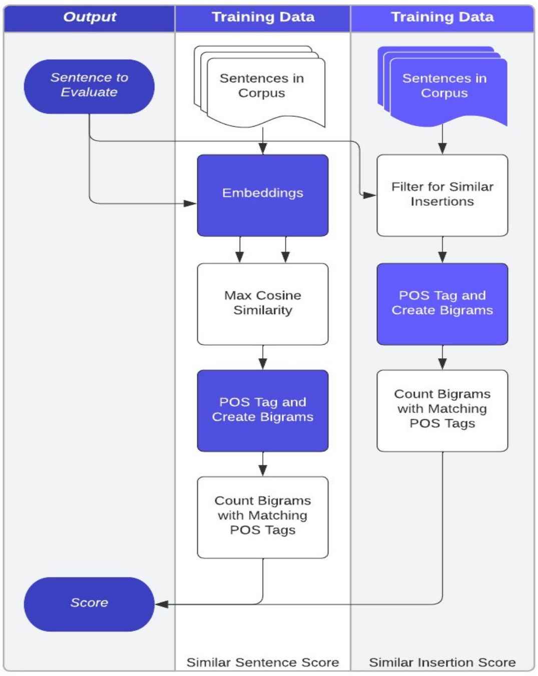

Figure 3: Evaluation and scoring pipeline

Scoring method	Scores
Similar sentence score (hard-matching)	0.156
Similar sentence score (soft-matching)	0.215
Similar insertion score (dataset)	0.250
Similar insertion score (gold)	0.406

Table 2: Evaluation scores of the two method on the output of sentences from the dataset and the gold standard.

5 Results and Discussion

5.1 Results

The output sentences with the inserted disfluencies are evaluated using the two scores. We see, as expected, the soft matching score is greater than the hard-matching score for the similar sentence method. Additionally, to benchmark our evaluation method, we provided the similar insertion method with sentences from manually transcribed interviews. This is our gold standard. Since the context of these sentences are not the same as the dataset, we only calculate the similar insertion score, skipping the similar sentence score. This is detailed in Table 2. As evident from the numerical proximity of the two scores, we conclude that our evaluation method is sound. However, we realise that our bigram model performs marginally worse in its disfluency insertions when compared to insertions in real conversations.

5.2 Discussion

One of the setbacks that we faced in this study is finding suitable datasets that will work well for our purposes. We found that the Santa Barbara Corpus was one of the only datasets of spoken-text with disfluency annotations. Since the model was trained on just the Santa Barbara corpus, there are some concerns for limitations with the model's ability to generalize to real-world conversation data. In the future, we would like to use the context behind the spoken sentence as part of the model's features. We insert disfluencies into our speech for a reason and deducing this reason from the sentence's purpose and context can improve the insertions' accuracy significantly.

6 Division of Labor

There was equal contribution from all authors for the proposal, poster and report. Ryan Luu and Ira Deshmukh worked on the dataset pre-processing.

Bowman Brown worked on the model. Alfianto Widodo and Parth Vipul Shah worked on the evaluation metrics. There was also equal contribution towards the theorizing, research and other logistics of this project.

References

[Bakti et. al. 2009] Maria Bakti 2009. *Speech Disfluencies in Simultaneous Interpretation*

[Einolghozati et. al. 2020] Arash Einolghozati and Anchit Gupta and Keith A. Diedrick and S. Gupta 2020. *Sound Natural: Content Rephrasing in Dialog Systems*

[Kharitonov et. al. 2022] Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu-Anh Nguyen, Morgane Rivière, Abdelrahman Mohamed, Emmanuel Dupoux, Wei-Ning Hsu 2022. *Text-Free Prosody-Aware Generative Spoken Language Modeling*

[Kühne K, Fischer MH, Zhou Y. 2020] The Human Takes It All: Humanlike Synthesized Voices Are Perceived as Less Eerie and More Likable. Evidence From a Subjective Ratings Study. Front Neurorobot. 2020 Dec 16;14:593732. doi: 10.3389/fnbot.2020.593732. PMID: 33390923; PMCID: PMC7772241.]

[Kong et. al. 2020] Jungil Kong, Jaehyeon Kim, Jaekyoung Bae 2020. *HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech SynthesisQuality*

[Li et. al. 2021] Yinghao Aaron Li, Ali Zare, Nima Mesgarani 2021. *StarGANv2-VC: A Diverse, Unsupervised, Non-parallel Framework for Natural-Sounding Voice Conversion*

[Lee et. al. 2021] Jaeryoung Lee 2021. *Generating Robotic Speech Prosody for Human Robot Interaction: A Preliminary Study*

[Passali et. al. 2022] T. Passali, T. Mavropoulos, G. Tsoumakas, G. Meditskos, S. Vrochidis 2022. *LARD: Large-scale Artificial Disfluency Generation*

[Pisoni DB, Manous LM, Dedina MJ 1987] Comprehension of natural and synthetic speech: effects of predictability on the verification of sentences controlled for intelligibility. Comput Speech Lang. 1987 Sep;2(3-4):303-320. doi: 10.1016/0885-2308(87)90014-3. PMID: 23226919; PMCID: PMC3515065.]

[Reimers et. al. 2019] Reimers, Nils and Gurevych, Iryna 2019. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*

[Santa Barbara Corpus] Du Bois, John W., Wallace L. Chafe, Charles Meyer, Sandra A. Thompson, Robert Englebretson, and Nii Martey. 2000-2005. Santa Barbara corpus of spoken American English, Parts 1-4. Philadelphia: Linguistic Data Consortium.

www.linguistics.ucsb.edu/research/santa-barbara-corpus

[Tan et. al. 2022] Xu Tan, Jiawei Chen, Haohe Liu, Jian Cong, Chen Zhang, Yanqing Liu, Xi Wang, Yichong Leng, Yuanhao Yi, Lei He, Frank Soong, Tao Qin, Sheng Zhao, Tie-Yan Liu 2022. *NaturalSpeech: End-to-End Text to Speech Synthesis with Human-Level Quality*

[Wang et. al. 2022] Wang, Shaolei and Wang, Zhongyuan and Che, Wanxiang and Zhao, Sendong and Liu, Ting 2022. *Combining Self-supervised Learning and Active Learning for Disfluency Detection*

[Wester et. al. 2015] Mirjam Wester, Matthew Aylett, Marcus Tomalin, Rasmus Dall 2015. *Artificial Personality and Disfluency*