Semantic Parsing of Disfluent Speech

Priyanka Sen
Amazon Alexa AI
Cambridge, UK
sepriyan@amazon.com

Isabel Groves
Amazon Alexa AI
Cambridge, UK
isabeg@amazon.com

Abstract

Speech disfluencies are prevalent in spontaneous speech. The rising popularity of voice assistants presents a growing need to handle naturally occurring disfluencies. Semantic parsing is a key component for understanding user utterances in voice assistants, yet most semantic parsing research to date focuses on written text. In this paper, we investigate semantic parsing of disfluent speech with the ATIS dataset. We find that a state-of-the-art semantic parser does not seamlessly handle disfluencies. We experiment with adding real and synthetic disfluencies at training time and find that adding synthetic disfluencies not only improves model performance by up to 39% but can also outperform adding real disfluencies in the ATIS dataset.

1 Introduction

Spoken language differs from written language. Unlike written texts, spontaneous speech frequently contains disfluencies such as filled pauses, repetitions, repairs, and false starts. These affect around 6% of words (Kasl and Mahl, 1965; Tree, 1995; Bortfeld et al., 2001) and occur in both human-human and human-computer interactions (Oviatt, 1995). While considerable research has been done on speech disfluencies in syntactic parsing (Johnson and Charniak, 2004; Wang et al., 2018; Jamshid Lou et al., 2019), semantic parsing of disfluent speech has received less attention.

Semantic parsing is the task of mapping written text to a representation of its meaning. In voice assistants, such as Amazon Alexa, Google Assistant, or Siri, an automated speech recognition (ASR) component is often followed by a semantic parsing task that identifies the intent and slots of the transcribed utterance. In voice-based contexts, speech disfluencies are to be expected, and so a voice assistant must accurately support disfluent speech to support more natural user interaction. Despite the growing popularity of voice assistants, semantic parsing research to date has focused on written language. Most popular semantic parsing datasets only include written forms that were generated from grammars (Coucke et al., 2018) or crowdsourced (Gupta et al., 2018) and do not contain speech phenomena such as disfluencies.

One approach to parsing disfluent speech is to detect and remove transcribed disfluencies in a post-processing step after ASR with a disfluency detector (Zayats et al., 2016; Jamshid Lou et al., 2018). Alternatively, some ASR models attempt to directly produce fluent transcriptions from disfluent inputs (Jamshid Lou and Johnson, 2020) but have lower performance. We investigate semantic parsing of disfluent utterances directly, without ASR post-processing, by training a single semantic parsing model that handles both fluent and disfluent utterances. Our approach has the advantage of being simple and easier to implement than approaches that require multiple models in a pipeline.

We investigate semantic parsing of disfluent speech using the ATIS semantic parsing dataset (Price, 1990), which was originally collected as spontaneous speech. ATIS is a small dataset but currently the only semantic parsing dataset that contains speech disfluencies. As such, our findings are limited by dataset size and results may differ given more data, however, we still find interesting preliminary results. First, we compare performance of a state-of-the-art semantic parser on disfluent and fluent examples in ATIS and identify a performance gap of 79%. To address this gap, we experiment with augmenting the training data with real disfluencies, synthetically-generated disfluencies, and a combination of both. We find that adding synthetic disfluencies adds up to 39% improvement in exact-match accuracy for disfluent utterances. We also find that real disfluencies degrade perfor-

Fluent	please find a flight from detroit michigan to st. petersburg arriving before 10 pm
Speech Transcription	please find a flight <from> <saint> <petersburg> <excuse> <me> from detroit michigan to saint petersburg arriving before ten p m
Disfluent	please find a flight from st. petersburg excuse me from detroit michigan to st. petersburg arriving before 10 pm

Table 1: An example of a disfluent utterance transcribed in ATIS

mance on fluent data, while synthetic disfluencies do not. This demonstrates that adding synthetic but targeted disfluent training data can be more effective than adding real but noisier data to improve model performance on disfluent utterances.

2 ATIS

The Airline Travel Information System (ATIS) corpus (Price, 1990) is a widely used dataset in semantic parsing and spoken language understanding. ATIS contains human-computer dialogs in American English where speakers asked a computer system about hypothetical flight planning scenarios. In our experiments, we use the ATIS data splits from Tur et al. (2010)'s *All Train* setup with 4,978 training examples (of which 500 are for development) and 893 test examples selected from the wider corpus as those that are context independent. We use the target meaning representation format with slots and intents introduced by Gupta et al. (2018), with the same modification as Rongali et al. (2020) of custom end-brackets.

	Real	Synthetic
Contains Filled Pause(s)	43.1%	73.0%
Contains Repair(s)	68.7%	66.5%
Fluent avg token length	13.6	11.1
Disfluent avg token length	15.4	13.5

Table 2: Description of ATIS disfluencies

Unlike most semantic parsing datasets, the ATIS corpus contains recorded speech transcribed in detail, which has been used to research characteristics of disfluencies (Shriberg, 1996; Savova and Bachenko, 2003). However, most semantic parsing work on ATIS uses a cleaned version of the utterances where disfluencies are removed (Price, 1990). We use the original ATIS transcriptions to restore disfluencies to the dataset. We reintroduce deleted words in repairs, partial words longer than 1 character, and filled pauses ('mm', 'uh', 'ah', 'um'). Utterances identified as disfluent are manually verified. An example is shown in Table 1 and dataset statistics about the disfluencies are in Table 2. We map the disfluent utterances to the target meaning representation of their fluent counterpart. This results in 314 disfluent utterances in the train set, 36 in development, and 56 in the test set, accounting for 6.9% of the dataset.

3 Model Architecture

Hyperparameter	Value
Batch Size	1400
Learning Rate	0.05
Epochs	100
Max Seq Length	50
Beam Width	4

Table 3: Hyperparameters used when training semantic parsing models on the ATIS dataset

Our semantic parsing model architecture is a reimplementation of the sequence-to-sequence model with pointer generator network (Vinyals et al., 2015) proposed by Rongali et al. (2020), which achieved state-of-the-art performance on three public datasets including ATIS. We use a pre-trained RoBERTa language model (Liu et al., 2019) as the encoder and a transformer based on Vaswani et al. (2017) as the decoder. The encoder converts a sequence of words into a sequence of embeddings. Then at each time step, the decoder outputs either a symbol from the output vocabulary or uses a pointer generator network to generate pointers to tokens in the source sequence. A final softmax layer provides a probability distribution over all actions, and beam search maximizes the output sequence probability. We train on a single Nvidia Tesla v100 16GB GPU with the hyperparameters shown in Table 3.

4 Experiments

We run four experiments to evaluate how our semantic parsing model handles speech disfluencies. Each model is evaluated against three test sets: the *Original* ATIS test set of 893 fluent utterances, the *Disfluent* subset of 56 disfluent test set utterances identified in §2, and the *Fluent Subset* containing the fluent versions of these 56 utterances. The Fluent and Disfluent subsets are small, however we train multiple models and report results as the average exact-match accuracy over 5 runs with standard deviation. We also calculate slot-match accuracy for one model per setting on the disfluent subset, which measures whether the slots in the representation are correct. Finally, we provide a breakdown of performance based on disfluency type for a selection of models in Table 5.

4.1 Experiment 1: Are models trained on fluent data robust to disfluencies?

First, we investigate how a semantic parsing model trained on fluent data performs on disfluent utterances. We train a model with the original ATIS dataset and evaluate on all test sets. The results are in the first row of Table 4. These results show that our ATIS model is not robust to disfluencies, scoring 79% lower on the disfluent subset compared to the fluent subset (0.02 vs. 0.81) even though these are variations of the same utterances. In an error analysis, we find that most of the errors on the disfluent subset come from incorrect slots rather than intents, and so we also report the slot accuracy on the disfluent subset (0.29). Common errors include repairs within slots where they should be deleted (e.g. *[SL:TolocCityName um washington]*), or mislabeling filled pauses as airport or state codes.

4.2 Experiment 2: Can we improve performance on disfluencies with real disfluent data?

Next we experiment with adding real disfluent examples to improve model performance. We add the 314 disfluent examples identified in §2 into the train set and 36 into the dev set and fully retrain the model. As shown in the second row of Table 4, adding real disfluent examples improves model performance on the disfluent subset by 32% exact-match accuracy and 17% slot accuracy. This is promising, especially since only 7% of the train set was disfluent. However, this improvement comes at a cost: average performance drops by 6% on the original test set and by 4% on the fluent subset. We see errors where meaningful tokens are erroneously dropped from slots, such as omitting *"closest"* when parsing *"which airport is closest to ontario california"*, or *"city"* from *"kansas city"*.

4.3 Experiment 3: Can we improve performance on disfluencies with synthetic data?

Adding disfluent examples improved performance on the disfluent subset, but real disfluencies can be sparse and costly to collect with target labels, requiring expert annotators familiar with the meaning representation. Additionally, real examples can be complex with multiple disfluencies (e.g. *"beginning on april thirtieth and returning no beginning on april twenty fifth and beginning on may sixth returning on may sixth"*. The drop in performance on fluent examples observed in §4.2 could suggest that adding real disfluencies caused the model to overfit to noisier signals, thus leading to worse performance. We hypothesize that adding simpler and more targeted synthetic disfluencies may improve model performance. Previous studies have also illustrated the benefits of augmenting training data with generated synthetic data for disfluency detection and removal (Bach and Huang, 2019; Dong et al., 2019; Wang et al., 2019). And so our third experiment evaluates whether synthetically-generated disfluencies can improve model performance.

We generate synthetic disfluencies for each fluent utterance in the ATIS train and dev set. While real disfluencies can be affected by several factors, such as sentence length (Shriberg, 1994), word frequency (Hartsuiker and Notebaert, 2009; De Jong, 2016), and cognitive load (Oviatt, 1995), we try to generate disfluencies with a simple method. This lets us generate disfluencies quickly, and also makes our generation process easy to replicate and transfer to new domains and languages. We leave research on the generation of more complex and linguistically-guided disfluencies for future work.

For each fluent utterance *u* we randomly generate one of 3 disfluent variations:

1. **Filled Pause:** Add 1 random filled pause per 10 words inserted after a random token in *u*. Example: *"um show me flights from denver"*
2. **False start repair:** Select a random second utterance *u2*. Take the first *n* tokens of *u2* (where *n* is a random number between 1 and

	Dataset sizes		Exact match accuracy			Slot accuracy
	Train	Dev	Original	Fluent	Disfluent	Disfluent
ATIS	4478	500	0.85 ± 0.01	$\mathbf{0.81 \pm 0.01}$	0.02 ± 0.02	0.29
+ real	4792	536	0.79 ± 0.03	0.77 ± 0.04	0.34 ± 0.02	0.46
+ 1% synth	4523	505	0.85 ± 0.01	0.80 ± 0.02	0.20 ± 0.05	0.47
+ 5% synth	4702	525	0.85 ± 0.01	0.80 ± 0.02	0.31 ± 0.03	0.50
+ 10% synth	4926	550	0.85 ± 0.01	0.80 ± 0.01	0.34 ± 0.02	0.52
+ 25% synth	5598	625	0.85 ± 0.01	0.78 ± 0.01	0.37 ± 0.03	0.60
+ 50% synth	6717	750	0.85 ± 0.01	0.79 ± 0.02	0.39 ± 0.02	0.61
+ 100% synth	8956	1000	$\mathbf{0.86 \pm 0.01}$	0.79 ± 0.02	0.40 ± 0.04	**0.64**
100% synth	4478	500	0.79 ± 0.01	0.73 ± 0.01	$\mathbf{0.41 \pm 0.01}$	**0.64**
+ real + 1% synth	4837	541	0.78 ± 0.01	0.78 ± 0.02	0.33 ± 0.02	0.46
+ real + 5% synth	5016	561	0.79 ± 0.01	0.78 ± 0.02	0.33 ± 0.02	0.47
+ real + 10% synth	5240	586	0.79 ± 0.00	0.76 ± 0.03	0.32 ± 0.00	0.47
+ real + 25% synth	5912	661	0.78 ± 0.02	0.77 ± 0.03	0.33 ± 0.03	0.47
+ real + 50% synth	7031	786	0.79 ± 0.01	0.76 ± 0.03	0.30 ± 0.02	0.46
+ real + 100% synth	9270	1036	0.76 ± 0.03	0.73 ± 0.05	0.33 ± 0.02	0.54

Table 4: Results reported as averages over 5 runs $\pm$ the standard deviation. *ATIS* refers to fluent examples. *Original* is the original test set of 893 utterances. *Fluent* and *Disfluent* are the 56 fluent/disfluent counterparts from transcriptions. *Slot Accuracy* is the accuracy of predicted slots.

	FPs	Repairs
# of examples	18	34
ATIS	0.06	0.06
+ real	0.72	0.12
+ 50% synth	0.72	0.18
+ 100% synth	**0.78**	**0.24**
100% synth	0.78	0.26
+ real + 50% synth	0.72	0.15
+ real + 100% synth	0.78	0.21

Table 5: A breakdown of performance on the disfluent subset comparing Filled Pauses (FPs) vs. Repairs for select models. Examples containing both filled pauses and repairs are excluded.

5, no larger than 60% of the tokens in *u2*) and prepend them to *u*, along with a random interregnum (e.g. 'sorry'), filled pause, or nothing. Example: *"what's the earliest flight mm show me flights from denver to philadelphia"*

3 **Repeat repair:** Split *u* after a random character, ensuring the final token of the first part *p1* is at least 2 characters long. We repeat up to 2 of final full tokens in *p1* then append the remainder *p2* with the complete token if split. Example: *"show me flights from denver to phila denver to philadelphia"*

Statistics about the synthetic disfluencies are in Table 2. We incrementally add between 1% to 100% of the disfluent examples into the original train and dev sets and fully retrain the model. We also train a model on only synthetic disfluencies.

The results in Table 4 show that adding 10% of the synthetic disfluencies has comparable performance to adding the same number of real disfluencies on the disfluent subset and has less degradation on the fluent test sets. A model trained entirely on synthetic disfluencies only achieves 41% exact-match accuracy and 64% slot accuracy on the disfluent subset, indicating there is still work to be done in creating more robust models and better synthetic disfluencies. The improvements of the synthetic models on the disfluent subset come at little to no cost on the fluent test sets. This lack of degradation supports our hypothesis that real disfluencies may add more noise at training time than synthetic disfluencies.

In Table 5, we see that the performance boost from synthetic examples comes mostly from better handling of repairs. For disfluent utterances containing filled pauses, real disfluencies alone can get high exact-match accuracy (72%), while the addition of synthetic data increases this up to 78%.

For models trained with either real or synthetic disfluencies, performance for repairs is worse than for filled pauses, with 12% accuracy with real disfluencies and up to 26% accuracy with synthetic disfluencies. This suggests that repairs are generally harder for our semantic parser to handle. Still, synthetic disfluencies outperform real disfluencies on repairs by up to 13%. For example, the partial word "*phoe*" in "*flight between dallas and phoe phoenix*" is correctly dropped by synthetic models, whereas models trained on fluent or real disfluencies mistake "*phoe*" for an airport code.

4.4 Experiment 4: Combining real and synthetic disfluent data

In our final experiment, we investigate whether real disfluencies contribute alongside synthetic examples. We combine the fluent ATIS examples with the real ATIS disfluencies and then incrementally add between 1% to 100% of the synthetic examples. The results are in Table 4. These results show an initial exact-match accuracy advantage on the disfluent subset from using real and synthetic examples. Models with real disfluencies and 1% of the synthetic disfluencies score higher than 1% of the synthetic disfluencies alone (0.33 vs. 0.20), but synthetic disfluencies alone outperform after 10%. Performance on the original test set and fluent subset remain low. These results reveal that even if a dataset of real disfluencies is available, it can still be better to add only simpler synthetic disfluencies to training data to avoid adding noise to the model.

5 Conclusion

In this work, we evaluate semantic parsing of disfluent utterances in the ATIS dataset. We highlight the importance of considering spoken language phenomena when building semantic parsers by showing that a state-of-the-art semantic parsing model trained on fluent data is not robust to naturally occurring speech disfluencies. Although the ATIS dataset is small, we present preliminary results showing that adding real or synthetic disfluencies at training time can significantly improve performance by up to 39%. While adding real disfluencies comes with degradation in performance on fluent utterances, adding synthetic disfluencies improves performance on disfluent utterances with almost no degradation on fluent utterances and requires no additional costs for annotations or labeling. These findings show that adding simpler, targeted synthetic disfluencies can be a practical and effective way to improve a semantic parser's performance on disfluent utterances. With the release of more and larger spoken language semantic parsing datasets in the future, we hope to replicate and strengthen our findings in future work.

References

Nguyen Bach and Fei Huang. 2019. Noisy BiLSTM-Based Models for Disfluency Detection. In *Proc. Interspeech 2019*, pages 4230–4234.

Heather Bortfeld, Silvia D Leon, Jonathan E Bloom, Michael F Schober, and Susan E Brennan. 2001. Disfluency rates in conversation: Effects of age, relationship, topic, role, and gender. *Language and Speech*, 44(2):123–147.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: An embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Nivja H De Jong. 2016. Predicting pauses in L1 and L2 speech: The effects of utterance boundaries and word frequency. *International Review of Applied Linguistics in Language Teaching*, 54(2):113–132.

Qianqian Dong, Feng Wang, Zhen Yang, Wei Chen, Shuang Xu, and Bo Xu. 2019. Adapting translation models for transcript disfluency detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6351–6358.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Robert J Hartsuiker and Lies Notebaert. 2009. Lexical access problems lead to disfluencies in speech. *Experimental Psychology*.

Paria Jamshid Lou, Peter Anderson, and Mark Johnson. 2018. Disfluency detection using auto-correlational neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4610–4619, Brussels, Belgium. Association for Computational Linguistics.

Paria Jamshid Lou and Mark Johnson. 2020. End-to-end speech recognition and disfluency removal. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2051–2061, Online. Association for Computational Linguistics.

Paria Jamshid Lou, Yufei Wang, and Mark Johnson. 2019. Neural constituency parsing of speech transcripts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2756–2765, Minneapolis, Minnesota. Association for Computational Linguistics.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy-channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 33–39, Barcelona, Spain.

Stanislav V Kasl and George F Mahl. 1965. Relationship of disturbances and hesitations in spontaneous speech to anxiety. *Journal of Personality and Social Psychology*, 1(5):425.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Sharon Oviatt. 1995. Predicting spoken disfluencies during human-computer interaction. *Computer Speech and Language*, 9(1):19–36.

Patti Price. 1990. Evaluation of spoken language systems: The ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! A sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, pages 2962–2968.

Guergana Savova and Joan Bachenko. 2003. Designing for errors: Similarities and differences of disfluency rates and prosodic characteristics across domains. In *Eighth European Conference on Speech Communication and Technology*.

Elizabeth Shriberg. 1996. Disfluencies in Switchboard. In *Proc. of International Conference on Spoken Language Processing, Addendum*, pages 11–14.

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis, University of California, Berkeley.

Jean E Fox Tree. 1995. The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech. *Journal of Memory and Language*, 34(6):709–738.

Gokhan Tur, Dilek Hakkani-Tür, and Larry Heck. 2010. What is left to be understood in ATIS? In *2010 IEEE Spoken Language Technology Workshop*, pages 19–24. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3529–3538, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Shaolei Wang, Wanxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang. 2019. Multi-task self-supervised learning for disfluency detection. *arXiv preprint arXiv:1908.05378*.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional LSTM. *Interspeech 2016*, pages 2523–2527.