# Improving Spoken Semantic Parsing using Synthetic Data from Large Generative Models

*Roshan Sharma[1*], Suyoun Kim[2], Daniel Lazar[3], Trang Le[2], Akshat Shrivastava[2], Kwanghoon An[2], Piyush Kansal[2], Leda Sari[2], Ozlem Kalinli[2], Mike Seltzer[2]*

[1]Carnegie Mellon University, USA [2]Meta, USA, [3]Coldrays, USA

suyounkim@meta.com

## Abstract

Spoken semantic parsing (SSP) involves generating machine-comprehensible parses from input speech. Training robust models for existing application domains represented in training data or extending to new domains requires corresponding triplets of speech-transcript-semantic parse data, which are expensive to obtain. In this paper, we address this challenge by examining methods that can use transcript-semantic parse data (unpaired text) without the corresponding speech. First, when unpaired text is drawn from existing textual corpora, Joint Audio Text (JAT) and Text-to-Speech (TTS) are compared as ways to generate speech representations for unpaired text. Experiments on the STOP dataset show that unpaired text from existing and new domains improves performance by 2% and 30% in absolute Exact Match (EM) respectively. Second, we consider the setting when unpaired text is not available in existing textual corpora. We propose prompting Large Language Models (LLMs) to generate unpaired text for existing and new domains. Experiments show that examples and words that cooccur with intents can be used to generate unpaired text with Llama 2.0. Using the generated text with JAT and TTS for spoken semantic parsing improves EM on STOP by 1.4% and 2.6% absolute for existing and new domains, respectively.

**Index Terms**: spoken language understanding, on-device, unpaired data,large language models, prompting

## 1. Introduction

Spoken Language Understanding (SLU) is essential for many real-world applications today, including conversational agents and virtual assistants. Spoken Semantic Parsing (SSP) is the SLU task that involves transforming a recording to a machine-comprehensible parse tree [1]. End-to-end models [2] operate directly on speech while cascade models [3] generate a semantic parse based on the speech transcript. Two-pass deliberation models [4] combine the best of both worlds, using first-pass transcripts and speech embeddings to perform spoken semantic parsing within a second pass. However, training such models with supervision requires matched triplets of speech, transcript, and semantic parse. Annotating these triplets is expensive, which limits the size of training data and, consequently, model performance.

The need for matched data can be alleviated by developing methods that can use unpaired text-only data. Text data (transcript-semantic parse) is more easily obtained than speech – either from existing textual corpora or by prompting Large Language Models (LLMs), and training models with a small amount of paired speech-text data and a large amount of unpaired text is useful. It is nontrivial to incorporate text-only data into end-to-end models because model output cannot be obtained without speech input. Prior work has explored the use of text data for speech recognition [5–7]. External language models trained in text can be used to interpolate token prediction probabilities [8], but require additional memory, making them unsuitable for on-device applications. Coordinated learning methods [9, 10] project speech and text to a shared embedding space for speech recognition, but such models require significant amounts of paired speech-text data to learn robust mappings. The final class of work generates speech representations for unpaired speech - Joint Audio Text (JAT) [11] uses mean speech embeddings from paired data to represent unpaired text. This is computationally inexpensive, but the speech embeddings do not contain information embedded in real speech. In contrast, synthetic speech from Text-to-Speech (TTS) models [5] produces informative speech representations but can be computationally expensive.

There are two cases where additional textual data may be acquired for semantic parsing, (a) to improve models on existing domains (ED) and (b) to support new domains (ND). In this paper, we compare JAT and TTS for SSP when unpaired text data is drawn from these two setups - ED and ND.

When unpaired text is not available from existing corpora, we propose to prompt Large Language Models (LLMs) [12–14] to generate textual data for SSP. LLMs have been used in prior work to generate synthetic data for text classification using approaches such as Self-Instruct [15], AttrPrompt [16], ZeroGen [17], and more recently use in-context learning with seed samples [18]. Semantic parsing requires sequence labeling, i.e., (a) it requires the correct identification of the number and identity of intent and slot tags, and (b) the correct placement of entity and slot tags to form the right parse tree, all while not inserting unrelated or unseen intent and slot tags. Therefore, generating useful and diverse data for semantic parsing is more complex than other classification tasks.

Prior work [19] has proposed the use of template-based masked training of BART to produce additional variants for masked words; however, this limits the potential lexical diversity of the generated data, and requires a significant amount of labeled data, which may not be available for the ND setting. Since LLMs can learn in context and generalize better in few-shot settings, they need fewer exemplars to generate diverse and high-quality synthetic data for semantic parsing. This paper addresses the task of generating synthetic text data for semantic parsing by using different prompting approaches with Llama 2.

For the ED setup, it is sufficient to generate transcripts (similar utterances ) since semantic parses can be obtained from transcripts using pre-trained semantic parsers. We describe two prompting methods: (a) intent-word-based prompting (IWP),

*Work done while author was at Meta, author is at Google now

where the LLM produces transcripts corresponding to a particular intent class and containing words that co-occur with the intent, and (b) exemplar-based prompting (EP), where it generates transcripts that are similar to provided examples. We generate pseudo-labels for the generated utterances using a pre-trained RoBERTa [20] model and train SSP models using JAT. We find that EP is simpler but IWP generates the desired intent more often. Using data from both methods improves the Exact Match (EM) on STOP data by 1.4 points absolute.

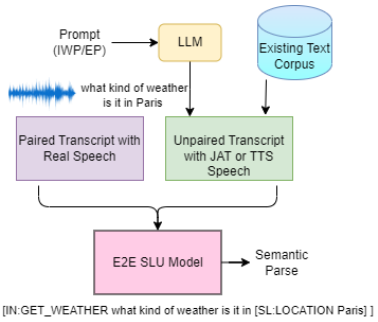


Figure 1: *This paper: We describe ways to unpaired text to train deliberation models, where unpaired data can be obtained from LLMs or existing textual corpora. We use JAT or TTS to obtain speech representations of unpaired data*

For the ND setup, pre-trained models for pseudo-labeling are unavailable for the new domain(s), and hence LLMs are used to generate the seqlogical form (containing the transcript with intent and slot tags annotated) of semantic parses directly. The transcript is then inferred from the seqlogical form of the semantic parse. Exemplar-based prompting (EP) is used with 3 real examples for every possible intent-slot combination to generate large-scale data. We find that the generated data improves EM by 2.3 points absolute over a baseline that uses only 3 examples per combination.

## 2. Deliberation Model for SLU

Deliberation-based SLU models [4, 21] are two-pass models that predict an ASR transcript in the first pass. Using the first pass transcript and audio, it then generates the semantic parse in the second pass. In contrast to cascade models that utilize separately trained Automatic Speech Recognition (ASR) and SLU components, a deliberation model optimizes both ASR and SLU components jointly. To achieve on-device streaming functionality, the first pass ASR component is implemented using the Recurrent Neural Network Transducer (RNNT) [22–24].

To maintain transcription accuracy, the ASR component of our deliberation model is trained independently and kept frozen. Our deliberation-based SLU model comprises two primary modules: (1) Fusion, and (2) Decoder. The fusion module combines intermediate audio and text embeddings from the first pass RNNT encoder and predictor respectively. Using Multi-Head Attention [25], the fusion module generates a combined representation that is used by the transformer-based decoder module to predict the target semantic parse sequence.

## 3. Speech Representations

### 3.1. Joint Audio-Text Training (JAT)

Joint Audio-Text training (JAT) [11] is a recent approach for leveraging unpaired text-only data to improve ASR [10, 11, 26, 27]. Unlike shallow fusion that considers token distributions from an external neural network language model (NNLM), JAT does not require additional model parameters or latency, making it suitable for on-device streaming ASR. The core idea behind JAT is that speech representations for unpaired text can be generated by simply using average speech embeddings computed over available paired speech/text data. In this paper, we use the JAT approach to train our Spoken Language Understanding (SLU) models to enable training with both ”speech-text-semantic parse” and ”text-semantic parse” datasets.

### 3.2. Speech Synthesis with Voicebox

Voicebox[28] is a state-of-the-art non-autoregressive speech generation model based on Flow Matching [29]. We generate representations for unpaired text by extracting speech features from synthesized speech. Synthetic speech can be obtained by using Voicebox in TTS mode, i.e. where audio is generated by conditioning on input text. Different from [28], the Voicebox model we use represents input text as graphemes rather than phonemes. To generate audio, we first sample unit durations for each grapheme in the input text using a flow-matching-based duration model and then upsample the grapheme sequence using the unit duration information. This information is used as conditioning to generate the spectrogram using the audio model. Finally, we used a HiFi-GAN [30] vocoder to convert the spectrograms into time-domain signals.

## 4. Generating Textual Data with LLama 2.0

LLama 2.0 [14] is a public open-source large language model trained on large volumes of publicly available data and code with context as large as 4096. In this paper, we use the 13B parameter chat model.

### 4.1. Generating Textual Data for Existing Domains

In the ED setup, we propose to use LLMs to generate transcripts. Corresponding semantic parses are obtained using a pseudo-labeling textual semantic parse model trained on existing paired data. The semantic parse model here takes transcripts as inputs and produces pseudo-label semantic parses as output. Transcripts can be generated using one of two prompting strategies, i.e., intent-word-based or exemplar-based.

#### *4.1.1. Intent Word-based prompting (IWP)*

The goal of IWP is to generate transcripts that may be classified under a certain intent, optionally containing ”intent words”. Intent words are the words from semantic parses that occur most frequently with given intents after removing stop-words. The 40 words that co-occur most frequently with every intent in the STOP data are used as intent words. 40 examples are generated for every intent and intent-word combination. Though IWP produces good synthetic data, it is limited by the fact that words that co-occur less frequently with the intent are less related to the intent. Such examples produced with less relevant intent words may not be classified under the desired intent class. This also limits the amount of synthetic data that can be generated since the LLM cannot generate many unique examples using a small number of intent-intent word combinations.

#### *4.1.2. Exemplar-based Prompting (EP)*

Since LLMs are strong in-context learners [31], an alternative approach is to prompt LLMs to generate transcripts based on

examples. For every intent-slot combination, we provide up to 4 random example transcripts and ask the model to generate 60 more transcripts that are similar but have diverse sentence structures. Though the resulting transcripts may not always correspond to the intent classes from which the examples are drawn, this method enables us to generate larger volumes of data without duplication.

#### *4.1.3. Semantic Parse generation and Quality Assessment*

Transcripts generated by LLMs are first normalized – written text is converted to spoken form, punctuation except apostrophes are removed and text is transformed into lower case. Semantic parse pseudo-labels are obtained from these normalized transcripts using a strong RoBERTa-based semantic parser trained on STOP (EM=86.8). To assess data quality, we compare the intent in the obtained pseudo-labels to the intent in the prompt for IWP or the intent of the provided examples for EP. Intent Match Accuracy (IMA) is defined as the percentage of times the intent of the pseudo-label matches the desired intent of the prompt.

### 4.2. Transcript-Semantic Parse for New Domains

For new domains, paired data and pre-trained models are not available, and therefore, we would need to directly generate pairs of transcript and semantic parse. One way to do this is to generate pairs of semantic parse and corresponding transcript using LLMs directly, however, maintaining consistency across generated parses and transcripts is challenging for current LLMs. Another alternative is to generate only the seqlogical form of the semantic parse from the LLM and infer the transcript from the parse. The seqlogical form of the parse, unlike the decoupled form, comprises all the words in the transcript along with slot and intent tags. Therefore, the transcript can be obtained from the seqlogical parse merely by removing slot and intent tags.

#### *4.2.1. Exemplar-based Prompting*

We assume that (a) the intents and slots that must be recognized for the new domain are known, (b) the slots that may occur with every intent, i.e., the intent-slot combinations are known, and (c) some manually annotated examples for every intent-slot combination are known. Using this information, LLMs can be prompted to produce new seqlogical parses for a given intent-slot combinations. The prompt first describes the steps to generate a valid seqlogical parse and then presents up to 3 examples of seqlogical parses with the desired intent-slot combinations.

#### *4.2.2. Post-processing*

The generated seqlogical parses are checked for invalid placement of brackets, and Out of Vocabulary (OOV) intents and slots. OOV intents were fixed by re-prompting the model to replace OOV intents with correct intents and replace any intents other than the first. Any OOV slots are removed while retaining corresponding slot words.

## 5. Experimental Setup

### 5.1. STOP Data, Model and Metrics

**Data**: STOP [1] [32] is a public dataset with 100 hours of real speech for spoken semantic parsing. STOP has data for 8 domains - alarm, event, messaging, music, navigation, reminder, timer, and weather and has 28 unique intents and 82 slot types.
**Metrics**: Exact Match (EM) is used to evaluate all our models. We report EM (No Err) and EM w/ Err, which are the Exact Match accuracies averaged over utterances with no ASR error and averaged over utterances with any ASR error respectively.
**Model Configuration**: For the ASR module, we use RNNT with 3 layers of conformer in the encoder, 1 layer of LSTM in the predictor, and 1 linear layer in the joiner. For the deliberation model, we use attention in the Fusion module, 2 transformer encoder layers in the Pooling module, and a transformer decoder layer with a pointer-generator in the Decoder module [21]. Models are optimized with Adam [33], having a peak learning rate of 8e-3.
**Voicebox TTS Model**: We use a Voicebox model trained on approximately 14k hours of manually transcribed data that comprises a diverse range of speakers, accents, topics, and acoustic conditions. The audio model has 12 transformer layers [25] containing 16 attention heads, convolutional positional embeddings [34] and ALiBi self-attention bias [35]. Graphemes are embedded into 80-d features and concatenated with the 80-d log-mel features. The duration model has 8 transformer layers with 8 heads, and graphemes are embedded into 40-d features. Hyperparameters are similar to the setup described in [28].
**Computational Cost** : Our experiments were performed on a single node with 8 V100-32 GB GPUs on the cluster. Each run took approximately 18 hours for model training. For LLama2 inference, we used 4 x V100-32 GB or 2 x A100-40GB with model parallelism and fp32 precision. For Voicebox inference, we used 1X V100-32 GB GPUs over 40 parallel processes to speed up speech synthesis.

### 5.2. Setup: Textual Data from Text Corpora

For experiments where we assume textual data is available, we split the STOP datasets into two parts. We perform two experiments – one using the first and second splits as paired and unpaired data respectively and the other using the second and first splits as paired and unpaired data respectively. The average performance across these 2 experiments is reported in each case. In the ED setup, equal amounts of data from every domain are present in the two splits. For the ND setup, STOP is split by domain, where one split contains all training data from 4 domains(messaging, reminder, time, and weather), while the other split contains training data from the other 4 domains (alarm, event, music, and navigation). Both splits are designed to ensure that they have a nearly equal number of utterances.

### 5.3. Setup: Textual Data from LLMs

When unpaired data is not available, we use Llama 2.0 to generate examples for the ED and ND setups. For the ED setup, LLama 2.0 is used to generate utterances. We then use a pre-trained 12-layer RoBERTa model trained on STOP to generate pseudo-labels for the generated utterances. We augment STOP with the generated LLama 2.0 transcript-semantic parse. JAT is used to represent LLama 2 text.

For the ND setup, LLama 2.0 generated data is not suitable as a real test set since it does not have matching real speech. Therefore, we choose to partition the existing STOP data into 7 seen domains and 1 new domain - weather. We use exemplar-based prompting to generate transcript-semantic parse pairs for weather. For this, real examples of transcript-semantic parse from STOP are used. We use TTS to generate equivalent speech representations for the generated data. We compare the perfor-

[1] STOP was used per its LICENSE terms

mance on the weather domain for models trained on (a) 7 domains of STOP, (b) 7 domains of STOP with examples for the weather (with TTS for examples and real speech for 7 domains), (c) 7 domains of STOP with examples and Llama 2.0 generated data, and (d) the topline that uses 7 domains of STOP with real data and TTS.

# 6. Experiments

Table 1: *Comparing JAT and TTS as speech representations for unpaired text from ED and ND. Number of paired and unpaired utterances, and Exact Match (EM) is reported*

| | Model | #Pair/#Unpair | EM | EM(No Err) | EM w/ Err |
|---|---|---|---|---|---|
| *ED* | Baseline | 60.4k / 0 | 64.25 | 80.51 | 24.37 |
| | w/ JAT | 60.4k / 60.4k | 66.92 | 83.90 | 25.25 |
| | w/ TTS | 60.4 / 60.4k | **67.05** | **83.88** | **25.80** |
| *ND* | Baseline | 60.7k / 0 | 33.28 | 41.32 | 13.54 |
| | w/ JAT | 60.7k / 60.1k | 57.74 | 73.34 | 19.50 |
| | w/ TTS | 60.7k / 60.1k | **63.95** | **80.70** | **22.88** |
| | Topline | 120.9k / 0 | 67.67 | 84.52 | 26.34 |

Table 2: *Impact of Paired-Unpaired Data Ratio on JAT Performance under the Existing Domain Setting*

| Pair (%) | Unpair (%) | EM-No Err | EM-ASR Error | EM (overall) |
|---|---|---|---|---|
| 0 | 100 | 85.48 | 21.22 | 66.87 |
| 30 | 70 | 84.27 | 24.67 | 67.01 |
| 50 | 50 | 84.15 | 25.5 | 67.17 |
| 70 | 30 | 84.24 | 25.43 | 67.2 |
| 100 | 0 | 84.52 | 26.34 | 67.67 |

### 6.1. When textual data is available

Table 1 compares the performance of different models for the ED and ND settings where unpaired text is drawn from existing domains and new domains respectively. Across both ED and ND setups, we find that the use of unpaired text improves EM scores.

For the ED setup, we find that JAT and TTS achieve similar Exact Match scores. Since JAT is comparable in performance to TTS and relatively inexpensive compared to complex TTS models like Voicebox, JAT is optimal for the ED setup. TTS model training depends on the specific model, but in our case, Voicebox training takes 3 days on 8 GPUs, and inference to produce synthetic speech takes 3 hours on 40 parallel GPU inference jobs. In comparison, JAT data preparation involves using mean speech embeddings, which takes 1 hour on 40 CPUs for the STOP training, evaluation, and test data. Therefore, JAT indeed takes little time in comparison to TTS.

Further, the difference between JAT and TTS appears to be primarily on utterances with ASR errors, since synthetic speech representations can be used to reduce the impact of ASR errors on semantic parsing. For the ND setup, we find that though JAT outperforms the baseline, TTS outperforms JAT. This is because new domains may have different entities and domain-specific terms that may be harder to recognize, and TTS provides valid speech representations that can be used to improve predictions based on the first-pass ASR.

Table 3: *Assessing the impact of augmenting the training data with LLama 2.0 generated utterances and RoBERTa pseudo-labels.EM is Exact Match Accuracy*

| Model | #Utts | IMA | EM | EM(No Err) | EM w/ Err |
|---|---|---|---|---|---|
| STOP Baseline | 160k | - | 67.37 | 84.52 | 26.34 |
| + IWP-JAT | 230k | **68.87** | 68.12 | 84.96 | 26.82 |
| + EP-JAT | 218k | 64.24 | 68.21 | 85.01 | **27.04** |
| + (IWP+EP)-JAT | 298k | 67.87 | **68.75** | **85.82** | 26.86 |

### 6.2. LLama 2.0 Generated Data: ED Setup

Table 3 compares various prompting strategies for generating utterances in the same domain using Llama 2.0. We find that combining LLama-generated data with existing STOP data can improve performance across test examples with and without ASR errors. On further analysis, we find that significant improvements are observed across domains with relatively poor performance in the STOP baseline. Between IWP and EP, we find that EP is slightly better. Since EP is not constrained to generate utterances that may be classified under a given intent, the Intent Match Accuracy (IMA) is lower than that of IWP. Combining the data generated from both these strategies further improves performance over the STOP baseline.

### 6.3. LLama 2.0 Generated Data: ND Setup

Table 4: *Using TTS to generate speech for LLama 2.0 text when unpaired text is in an unseen new domain*

| Model | #Utts(Weather) | Weather EM | Overall EM |
|---|---|---|---|
| STOP 7 dom. | 0 | 0 | 54.61 |
| + 3 real example-TTS | 360 | 48.18 | 61.80 |
| + Exemplar LLama2-TTS | 2,910 | **50.82** | **62.29** |
| Topline: STOP Weather-TTS | 2,910 | 63.80 | 66.33 |

Table 4 compares the performance of baseline models that have no data for weather or 360 examples for weather with models that use LLama 2.0 generated data. Llama 2 generated text can improve performance by over 2 points absolute EM but lags behind the performance of a topline that uses data from STOP.

# 7. Conclusion

We address the high cost of manually labeling speech-transcript-semantic parse data for spoken semantic parsing by enabling models to use text-only data. JAT is preferred for unpaired text in existing domains for its efficiency and gain of 2.5 % EM over a paired data baseline while remaining within 0.1 % EM of the more computationally expensive TTS. For unpaired text in new domains, TTS outperforms JAT by 6 % absolute EM overall, with a gain of 30.6 % EM over a paired baseline. When text data cannot be obtained from existing text corpora, we propose to prompt LLMs to generate transcript-semantic parse pairs. We show that using different prompting strategies, we can generate unpaired text data in relatively large volumes. Using JAT and TTS, we can leverage this LLM-generated data to further improve SSP by 1.4 % EM and 2.6 % EM absolute for existing and new domains.

# 8. References


[1] S. Wang, A. Shrivastava, and S. Livshits, *Treepiece: Faster semantic parsing via tree tokenization*, 2023.

[2] S. Arora, H. Futami, S.-L. Wu, J. Huynh, Y. Peng, Y. Kashiwagi, E. Tsunoo, B. Yan, and S. Watanabe, "A study on the integration of pipeline and e2e slu systems for spoken semantic parsing toward stop quality challenge," in *Proc. ICASSP*, 2023, pp. 1–2.

[3] H. Futami, J. Huynh, S. Arora, S.-L. Wu, Y. Kashiwagi, Y. Peng, B. Yan, E. Tsunoo, and S. Watanabe, "The pipeline system of asr and nlu with mlm-based data augmentation toward stop low-resource challenge," in *Proc. ICASSP*, 2023, pp. 1–2.

[4] D. Le, A. Shrivastava, P. Tomasello, S. Kim, A. Livshits, O. Kalinli, and M. L. Seltzer, "Deliberation model for on-device spoken language understanding," *Interspeech*, 2022.

[5] G. Wang, A. Rosenberg, Z. Chen, Y. Zhang, B. Ramabhadran, Y. Wu, and P. Moreno, "Improving speech recognition using consistent predictions on synthesized speech," in *Proc. ICASSP*, 2020, pp. 7029–7033.

[6] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A comparison of techniques for language model integration in encoder-decoder speech recognition," in *2018 IEEE spoken language technology workshop (SLT)*, 2018, pp. 369–375.

[7] T. Hori, R. Astudillo, T. Hayashi, Y. Zhang, S. Watanabe, and J. Le Roux, "Cycle-consistency training for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 6271–6275.

[8] Z. Meng, Y. Gaur, N. Kanda, J. Li, X. Chen, Y. Wu, and Y. Gong, "Internal Language Model Adaptation with Text-Only Data for End-to-End Speech Recognition," in *Proc. Interspeech*, 2022, pp. 2608–2612.

[9] Z. Chen, Y. Zhang, A. Rosenberg, B. Ramabhadran, P. J. Moreno, A. Bapna, and H. Zen, "MAESTRO: Matched Speech Text Representations through Modality Matching," in *Proc. Interspeech*, 2022, pp. 4093–4097.

[10] T. N. Sainath, R. Prabhavalkar, A. Bapna, Y. Zhang, Z. Huo, Z. Chen, B. Li, W. Wang, and T. Strohman, "Joist: A joint speech and text streaming model for asr," in *Proc. SLT*, 2023, pp. 52–59.

[11] S. Kim, K. Li, L. Kabela, R. Huang, J. Zhu, O. Kalinli, and D. Le, "Joint audio/text training for transformer rescorer of streaming speech recognition," *EMNLP*, 2022.

[12] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.*, "Training language models to follow instructions with human feedback," *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022.

[13] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[14] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[15] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi, "Self-instruct: Aligning language models with self-generated instructions," in *Proc. ACL*, 2023, pp. 13 484–13 508.

[16] Y. Yu, Y. Zhuang, J. Zhang, Y. Meng, A. Ratner, R. Krishna, J. Shen, and C. Zhang, "Large language model as attributed training data generator: A tale of diversity and bias," in *Thirty-Seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[17] J. Ye, J. Gao, Q. Li, H. Xu, J. Feng, Z. Wu, T. Yu, and L. Kong, "ZeroGen: Efficient zero-shot learning via dataset generation," in *Proc. EMNLP*, 2022, pp. 11 653–11 669.

[18] Y. Yu, Y. Zhuang, R. Zhang, Y. Meng, J. Shen, and C. Zhang, "ReGen: Zero-shot text classification via training data generation with progressive dense retrieval," in *Proc. ACL*, 2023, pp. 11 782–11 805.

[19] K. Tran and M. Tan, "Generating synthetic data for task-oriented semantic parsing with hierarchical representations," in *Proceedings of the Fourth Workshop on Structured Prediction for NLP*, 2020, pp. 17–21.

[20] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Ro{bert}a: A robustly optimized {bert} pretraining approach*, 2020.

[21] S. Kim, A. Shrivastava, D. Le, J. Lin, O. Kalinli, and M. L. Seltzer, "Modality confidence aware training for robust end-to-end spoken language understanding," *Interspeech*, 2023.

[22] A. Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[23] S. Kim, Y. Shangguan, J. Mahadeokar, A. Bruguier, C. Fuegen, M. L. Seltzer, and D. Le, "Improved neural language model fusion for streaming recurrent neural network transducer," in *Proc. ICASSP*, 2021, pp. 7333–7337.

[24] C. Liu, F. Zhang, D. Le, S. Kim, Y. Saraf, and G. Zweig, "Improving RNN Transducer Based ASR with Auxiliary Tasks," in *Proc. SLT*, 2021.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[26] T. N. Sainath, R. Pang, R. J. Weiss, Y. He, C.-c. Chiu, and T. Strohman, "An attention-based joint acoustic and text on-device end-to-end model," in *Proc. ICASSP*, 2020, pp. 7039–7043.

[27] P. Wang, T. N. Sainath, and R. J. Weiss, "Multitask training with text data for end-to-end speech recognition," *arXiv preprint arXiv:2010.14318*, 2020.

[28] M. Le, A. Vyas, B. Shi, B. Karrer, L. Sari, R. Moritz, M. Williamson, V. Manohar, Y. Adi, J. Mahadeokar, *et al.*, "Voicebox: Text-guided multilingual universal speech generation at scale," *arXiv preprint arXiv:2306.15687*, 2023.

[29] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[30] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 022–17 033, 2020.

[31] J. Wei *et al.*, "Emergent abilities of large language models," *Transactions on Machine Learning Research*, 2022, Survey Certification.

[32] P. Tomasello, A. Shrivastava, D. Lazar, P.-C. Hsu, D. Le, A. Sagar, A. Elkahky, J. Copet, W.-N. Hsu, Y. Adi, *et al.*, "Stop: A dataset for spoken task oriented semantic parsing," in *Proc. SLT*, 2023, pp. 991–998.

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, Y. Bengio and Y. LeCun, Eds., 2015.

[34] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[35] O. Press, N. A. Smith, and M. Lewis, "Train short, test long: Attention with linear biases enables input length extrapolation," *arXiv preprint arXiv:2108.12409*, 2021.