INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

End-to-End Spoken Language Understanding for Generalized Voice Assistants

Michael Saxon[1,2†], *Samridhi Choudhary*[1], *Joseph P. McKenna*[1‡], *Athanasios Mouchtaris*[1]

[1]Alexa AI, Amazon, USA
[2]University of California, Santa Barbara, USA
saxon@ucsb.edu, samridhc@amazon.com, mouchta@amazon.com

Abstract

End-to-end (E2E) spoken language understanding (SLU) systems predict utterance semantics directly from speech using a single model. Previous work in this area has focused on targeted tasks in fixed domains, where the output semantic structure is assumed a priori and the input speech is of limited complexity. In this work we present our approach to developing an E2E model for generalized SLU in commercial voice assistants (VAs). We propose a fully differentiable, transformer-based, hierarchical system that can be pretrained at both the ASR and NLU levels. This is then fine-tuned on both transcription and semantic classification losses to handle a diverse set of intent and argument combinations. This leads to an SLU system that achieves significant improvements over baselines on a complex internal generalized VA dataset with a 43% improvement in accuracy, while still meeting the 99% accuracy benchmark on the popular Fluent Speech Commands dataset. We further evaluate our model on a hard test set, exclusively containing slot arguments unseen in training, and demonstrate a nearly 20% improvement, showing the efficacy of our approach in truly demanding VA scenarios.

Index Terms: End-to-end, spoken language understanding, voice assistants, BERT, transformers, pretraining

1. Introduction

Spoken language understanding (SLU) systems produce interpretations of user utterances to enable interactive functions [1]. SLU is typically posed as a recognition task, where an utterance's semantic *interpretation* is populated with results from various sub-tasks, including utterance-level label identification tasks like domain and intent classification as well as sequence tagging tasks such as named entity recognition (NER) or slot filling. The conventional approach to SLU breaks the task into two discrete problems, each solved by a separately-trained module. First, an automatic speech recognition (ASR) module transcribes the utterance to text. This is then passed on to a natural language understanding (NLU) module that infers the utterance interpretation by predicting the domain, intent and slot values. Deep learning advances in both ASR [2–4] and NLU [5–7] have improved the performance of SLU systems, driving the commercial success of voice assistants (VAs) like Alexa and Google Home. However, a drawback of this modular design is that the components are trained independently, with separate objectives. Errors encountered in either model do not inform the other; in practice this means incorrect ASR transcriptions might be "correctly" interpreted by the NLU, thereby failing to provide the user's desired response. While work is ongoing in detecting [8], quantifying [9, 10], and rectifying [11, 12] these ASR driven NLU misclassifications, end-to-end (E2E) approaches are a promising way to address this issue.

Rather than containing discrete ASR and NLU modules, E2E SLU models are trained to infer the utterance semantics directly from the spoken signal [13–20]. These models are trained to maximize the SLU prediction accuracy where the predicted semantic targets vary from solely the intent [21, 22], to a full interpretation with domain, intents, and slots [13]. The majority of recent work on English SLU has targeted benchmark datasets such as ATIS [23], Snips [24], DSTC4 [25] and Fluent Speech Commands (FSC) [21], with FSC in particular gaining recent popularity. A similar collection of French spoken NER and slot filling datasets has been investigated [26]. Over the last year the state-of-the-art on FSC has progressed to over 99% test set accuracy for several E2E approaches [14–20]. However, there remains a gap between the E2E SLU capabilities demonstrated thus far and the requirements of a *generalized* VA [27]. In particular, existing benchmarks focus on tasks with limited **semantic complexity** and output **structural diversity**.

Different SLU use-cases have significantly different dataset requirements and feasible model architectures. For example, controlling a set of smart appliances may only require device names and limited commands like "on" and "off." Similarly, a flight reservation system can assume the user intends to book a flight [28]. In these settings, a restricted vocabulary and output structure is appropriate to ensure high performance. However, when interacting with generalized VAs like Alexa, users expect a system capable of understanding an unrestricted vocabulary, able to handle any song title or contact name. This leads to tasks with a long tail of rare utterances containing unique n-grams and specific slot values unseen during training, that are more *semantically complex* than the tasks tackled in aforementioned benchmark SLU datasets. Differences in semantic complexity across datasets can be assessed using n-gram entropy and utterance embedding MST complexity measures [27]. Furthermore, in generalized VA tasks the output label space is countably infinite, as any arbitrary sequence of words could be a valid slot output. Thus an assumption of a simple output structure is no longer valid, making the problem *structurally diverse*.

Designing an E2E system for semantically complex and structurally diverse SLU use-cases is the focus of this work. We present a transformer-based E2E SLU architecture using a multi-stage topology [13] and demonstrate its effectiveness in handling structurally diverse outputs, while achieving the 99% accuracy benchmark for FSC. We use an de-identified, representative slice of real-world, commercial VA traffic to test if our model is capable of handling *complex* datasets. Furthermore, we demonstrate how to leverage large-scale pretrained language models (BERT) and acoustic pretraining for increased robustness. We perform a supplementary analysis across multiple choices of differentiable interfaces for our multistage E2E setup. Finally, we show the performance of our proposed model

[†]Work completed during author's Amazon internship.
[‡]Work completed at Amazon, currently at Google Cloud AI.

http://dx.doi.org/10.21437/Interspeech.2021-1826

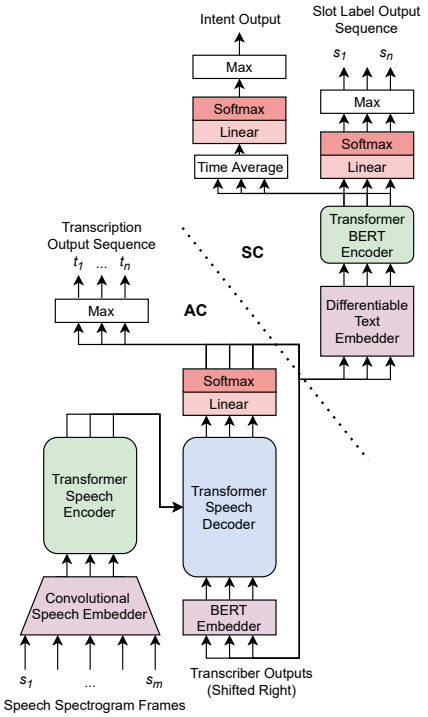

Figure 1: *A diagram depicting the full E2E SLU model and the soft Acoustic (AC) and Semantic (SC) component boundary.*

on "hard" data partitions which exclusively contain slot arguments that are absent from the training data, demonstrating more robust performance in demanding general VA settings.

2. Model Architecture

We adopted the multistage E2E topology from [13], that resembles an end-to-end trainable variation of the traditional modularized SLU architecture. Due to this resemblance, we find it helpful to think of our model, shown in Figure 1, as being composed of two components: an "acoustic component" (AC) and a "semantic component" (SC). The AC takes in speech spectrograms and outputs a sequence of wordpiece tokens. The SC ingests the AC's output posterior sequence and produces an utterance-level intent class and a sequence of wordpiece-level slot labels. These two components are connected by a modified embedder that is differentiable by operating on wordpiece posteriors. Thus gradients flow from SC to AC, enabling end-to-end training for the entire setup on a single training objective. The differentiable interface idea is similar to [29] except we employ it to build the SC around a pretrained neural language model.

This architecture gives us the flexibility of still being able to produce a transcription, from which the slot values can be extracted via a slot tagger. However, unlike the modular SLU, we propagate gradients from semantic loss all the way to the acoustic input layer. Moreover, we can selectively pretrain components with different datasets and various objectives across the speech and text modalities. For example, the AC can be pretrained using non-SLU, speech-only datasets, that are often available in large quantities. Similarly, since the SC operates on wordpiece-level data, it can be designed to use a pretrained language model, in this case BERT [30] as a text encoder, where we attach task-specific heads to create an appropriate SC. Therefore, we are able to incorporate appropriate inductive biases in the model, by capturing both the acoustic (via AC pretraining) and linguistic information (via SC pretraining) that is difficult to learn from relatively small E2E SLU datasets.

Acoustic Component (AC) — The AC is made up of a convolutional neural network (CNN)-based time-reducing embedder, a transformer encoder, and a transformer decoder. The input to the embedder consists of $256d$ log spectrograms with a 20 ms frame length and 10 ms frame spacing. These frames are embedded using three $1d$ convolutional layers, with an output size 240, kernel size 4, stride 2, and ReLU activations. After embedding, a sequence of encodings corresponding to 240 ms of input audio, with a 120 ms spacing are produced. This architecture is inspired from the time-reducing convolutional speech encoders employed in [16]. A sequence of wordpieces is then autoregressively transcribed from the encodings using a 12-layer, 12-head transformer encoder-decoder with hidden size 240, trained with teacher forcing during both pretraining and fine-tuning [31].

Semantic Component (SC) — The semantic component is made up of four parts—a differentiable embedder, a pretrained BERT encoder, an utterance-level dense intent decoder, and a wordpiece-level dense slot sequence decoder. The differentiable embedder performs the same function as the typical BERT embedder lookup table, but can take in uncertain posterior inputs from the AC during training, enabling end-to-end gradient flow. The pretrained BERT encoder is a standard 12 layer $768d$ transformer encoder, that takes in the sequence of embeddings from the differentiable embedder and outputs a sequence of encodings of equal length. The intent decoder is a single linear layer of size $768 \times N_{IC}$ (num. intent classes) that takes the time-averaged encoded sequence to generate a single intent class estimate. The slot label sequence decoder is a single linear layer of size $3072 \times N_{SL}$ (num. slot labels). The input to this decoder is formulated by concatenating the top 4 BERT encoder layer outputs at each step [30], while the output is a slot label estimate. The final sequence of (slot label, slot value) pairs is constructed by concatenating subsequent wordpiece tokens tagged with a slot label other than null.

Differentiable Embedders — In a non E2E system, an argmax over the vocabulary length dimension could be performed on the AC output, after which the BERT lookup table would embed the transcribed word-pieces. However, this approach interrupts gradient flow, thereby rendering E2E training impossible. We experimented with three different approaches to generate differentiable BERT input encodings from the AC output posteriors. As some approaches to doing this require producing a very large internal posterior or producing large matrix multiplications (vocab size x vocab size), we analyze their impacts on both accuracy and inference speed.

TopK: In this approach, the posterior sequence of the embedder is sorted along the vocabulary dimension to produce a sequence of tokens of decreasing likelihood. This is followed by generating a mixture of the top-k token embeddings using the embedding lookup table and the softmax values of the top-k tokens. We used $k = 20$.

MatMul: Here, we store a *vocab size* $\times$ *embedding size* matrix containing the input embedding for every token in the vocabulary. With this we can easily generate a confidence-weighted mixture of all possible embeddings by multiplying this matrix by the output softmax of the embedder.

Gumbel: Instead of taking an argmax over the vocabulary, we instead use the Gumbel-softmax trick [32] to select a single word whose embedding is then passed on to the SC at each step. Gumbel-softmax helps approximate a smooth distribution for back propagation, allowing gradient flow.

3. Methodology

We follow a two step training approach: (1) pretrain the AC and SC layers on appropriate datasets and optimization objectives to help encode acoustic and linguistic semantic information, then (2) fine-tune the entire model end-to-end on a task-specific VA dataset. Details for our training and evaluation methodologies, datasets, and baselines are provided below.

3.1. Pretraining

In the pretraining stage, the AC is trained for the ASR transcription task on 460 hours of clean LibriSpeech data [33]. Rather than using typical ASR-style subwords or full words as targets, the transcriptions are converted into BERT-style wordpiece sequences using the HuggingFace `bert-base-uncased` tokenizer [34]. This helps us prime the AC layers to return tokens in the format that is expected at input to the BERT encoder in the SC. We use the Adam optimizer to minimize the sequential ASR cross entropy loss $\mathcal{L}_{ASR}$.

We built the SC around the pretrained `bert-base-uncased` model distributed by HuggingFace [34]. We perform no task-specific text-level pretraining beyond the cloze (masked LM) task and next sentence prediction learning that is inherent to using a pretrained BERT module [30]. The final output linear layers (intent and slot decoders) are randomly initialized at the beginning of the end-to-end training phase.

3.2. End-to-end training

After pretraining, the AC and SC layers are composed such that the AC output posteriors are fed directly into the input of the SC, with the differentiable embedder acting as the embedding lookup component. This setup is trained on a three-term sum of categorical cross entropy loss for the ASR output sequence $\mathcal{L}_{ASR}$, the slot labels $\mathcal{L}_{Slot}$, and a single utterance-level intent $\mathcal{L}_{Intent}$. $\mathcal{L}_{Slot}$ is a sequence-level target where each token in the ASR output sequence is assigned either a null output or a slot label. This three-term loss (Eq. (1)) is minimized using Adam.

$$\mathcal{L}_{E2E} = \mathcal{L}_{Intent} + \mathcal{L}_{Slot} + \mathcal{L}_{ASR} \tag{1}$$

3.3. Model evaluation

We use greedy ASR decoding to produce the output sequence of wordpieces from the AC. The inputs to SC are the output posteriors rather than the discrete word choices themselves. We perform a grid search over learning rates $\in [10^{-5}, 0.01]$, dropout $\in (0, 1]$, and hidden layer sizes $\in \{120, 240, 400, 512\}$, as well as experiment with slanted triangular learning rate schedules and hierarchical unfreezing strategies as described in [35], to get the best performing model. All models were trained and evaluated on EC2 instances with Tesla V100 GPUs. In order to analyze the final SLU performance, we use three metrics:

1. **Intent Classification Error Rate (ICER)** - Ratio of the number of incorrect intent predictions to the total number of utterances.
2. **Slot Error Rate (SER)** - Ratio of incorrect slot predictions to the total number of labeled slots in the dataset.
3. **Interpretation Error Rate (IRER)** - Ratio of the number of incorrect interpretations to the total number of utterances. An incorrect interpretation is the one where either the intent or the slots are wrong. This "exact match" error rate is the strictest of our evaluation metrics.

3.4. Data

We use two E2E SLU datasets for our experiments - (1) the publicly available Fluent Speech Commands (FSC) and (2) an internal SLU dataset. Additionally, we create a "hard test set" to assess model performance in the most demanding scenarios in generalized VA. We use the average n-gram entropy and Minimum Spanning Tree (MST) complexity score as described in [27] to quantify their levels of semantic complexity.

Fluent Speech Commands — FSC [21] is an SLU dataset containing 30,043 utterances with a vocabulary of 124 words and 248 unique utterances over 31 intents in home appliance and smart speaker control. The SLU task on this dataset is just the intent classification task. It has an average n-gram entropy of 6.9 bits and an average MST complexity score of 0.2 [27].

Internal SLU Dataset — In order to analyze the effectiveness of our proposed architecture on a generalized voice assistant (VA) setting, we collect a random, de-identified slice of internal data from a commercial VA system. The data is processed so that users are not identifiable. The resulting dataset contains about 150 hours of audio, with over 100 different slot labels, dozens of intent classes and no vocabulary restrictions. It has an average entropy of 11.6 bits and an average MST complexity of 0.52 [27]. Both complexity metrics, alongside the less structurally constrained output label space, demonstrate that this task is more complex than FSC.

Hard Subset of Internal Traffic Data — In generalized VA, accuracy on semantic outliers is desirable. To assess this dynamic we produce a **hard test set** of 18k utterances from our internal dataset. This is done by selecting utterances that exclusively contain at least one *minimum-frequency bigram*, a pair of subsequent words that is not present in our training or validation sets. This test set helps us simulate how a system will perform on unforseen utterances that tend to arise in production VA.

3.5. Baselines

We design our baselines using a multitask E2E topology, defined by Haghani et al. [13]. Our ability to use proven E2E models vetted on public SLU tasks such as FSC, as baselines, is hampered by the fact that they are typically designed with non-generalized VA use-cases in mind. In particular, the hard subset classification task is impossible for the models designed according to the direct or joint topologies from [13] to perform without significant modification. Specifically, they lack the ability to select arbitrary words from the transcription vocabulary as slot values. Most high-performing models for FSC follow the direct or joint topology [14, 16, 20]. Instead, the multitask topology [13] provides a good contrast to our proposed multistage model; both maintain the necessary capability of identifying slots by labeling a sequence of wordpieces.

We analyze three baseline multitask models, that differ only in the sequential encoder and decoder used, in particular (1) unidirectional LSTM, (2) bidirectional LSTM, and (3) transformer. All baseline models use a CNN-based speech spectrogram embedder identical to the one presented in Section 2. This is followed by the speech sequence encoder using one of the three aforementioned encoder types. Finally, these encodings are decoded with task-specific heads, that consist of a dense layer for utterance level intent classification and word-level dense layer for sequential slot decoding. The final structured output for IRER evaluation contains the slot values and slot labels along with the intent label for the entire utterance. Our baselines allow us to evaluate both the efficacy of a multistage setup and of using a transformer based encoder-decoder with BERT.

Table 1: *Results from Internal Traffic Dataset, for both the regular and hard test sets. Relative improvement in absolute Intent Classification Error Rate (ICER), Slot Error Rate (SER) and Interpretation Error Rate (IRER) are reported as positive deltas over the Multitask LSTM Baseline (lowest performance).*

		Regular Test Set			Hard Test Set		
Model	*Num. Params.*	*ICER*	*SER*	*IRER*	*ICER*	*SER*	*IRER*
Multitask LSTM Baseline	45.9M	—	—	—	—	—	—
Multitask BiLSTM Baseline	67.3M	+0.5	+0.5	–1.1	+1.8	+0.2	+0.5
Multitask Transformer Baseline	101M	+4.0	–2.7	–0.1	+4.4	–3.5	+0.2
Ours (No Pretraining)	115M	+9.1	+37.1	+40.5	+11.4	+14.2	+18.1
Ours (BERT, AC Pretraining)	115M	**+9.3**	**+37.3**	**+42.8**	**+12.9**	**+15.0**	**+18.9**

4. Results

We present internal dataset results in Table 1. All metrics are reported as relative improvements in percent over the simplest baseline model (the unidirectional "multitask LSTM baseline"). We also report the results for our architecture with randomly-initialized AC and SC at the start of fine-tuning (No Pretraining). In this condition the model is only trained on our internal dataset, from scratch. As we can see our pretrained model, with both LibriSpeech AC pretraining and BERT language model pretraining, achieves the best performance with a 9.3% improvement in ICER, 37.3% in SER and a huge 42.8% in IRER, on the "regular" test set. For this table we use the best-performing Gumbel interface (subsection 4.1).

The hard test set results are especially noteworthy. While baselines struggle to correctly identify slot values at all, our model improves the hard test set IRER by $\approx 19\%$. Many of these slot arguments are never seen in the training data, and are only correctly classified because our model is able to successfully identify which wordpieces in the output sequence should correspond to a slot value. This gain in generalization performance is a strength of our approach and demonstrates the efficacy of this architecture for complex use-cases.

Our model achieved a 0.6% IRER (99.4% accuracy) on the FSC dataset. While our model is designed for a structurally diverse and semantically complex SLU use-case, it nevertheless meets the benchmark of beating 99% accuracy on FSC, previously demonstrated in [14, 17–20], and is therefore comparable to the state-of-the-art in intent-only classification performance on FSC.

4.1. Embedder analysis

We evaluate all three proposed differentiable emedders, by reporting the inference speed and accuracy for models containing each. We timed the speed of inference on a single, 32-utterance minibatch on a single GPU. We also report the ICER, IRER and the hard test set IRER (h-IRER) for each interface.

Table 2: *Comparing the speed and performance of the three differentiable interfaces on the "regular" test set.*

Interface	*Speed*	*ICER*	*IRER*	*h-IRER*
MatMul	—	—	—	—
Top20	+10 ms	+5.1	+2.7	+1.5
Gumbel	**–16 ms**	**+7.2**	**+4.3**	**+2.4**

As seen in Table 2, the Gumbel-softmax outperforms the other interfaces both in inference speed and error rates. The improved error rates suggest that certainty in the selection of words being passed to the SC improves performance.

5. Discussion

Our approach meets the 99% test accuracy benchmark on FSC. However, this benchmark task is simple, with low semantic complexity [27]. The key benefit to our approach is its ability to perform inference on a structurally diverse set of semantically complex utterances. Our multistage model, containing a differentiable interface with ASR- and NLU-level fine-tuning on task-specific data, is able to not only beat baselines on production-like structurally diverse traffic but also generalize to a hard test set uniquely composed of previously unseen slot arguments, achieving a 19% improvement over the very poor IRER achieved by the baselines.

We note that pretraining both the AC and SC modules only produced modest improvements over random initialization. This might be because the quantity of data provided during the fine-tuning stage is sufficiently large for achieving a good fit, providing a good sample of relevant transcriptions and interpretations. Alternatively, the pretrained representations might be too general or in the wrong domains; the audiobook speech in LibriSpeech and the massive corpora of internet text used to train BERT span diverse topics. This pretraining data may be of limited applicability to our setting when sufficient in-domain data is available.

For scenarios where generalized VA is necessary, but less training data is available, our proposed architecture would enable using a maximum amount of semantic pretraining for each modality of the model (speech and text). Apart from pretraining, following a multistage approach is also one of the core reasons that our model performs so well, especially for the hard test set. By accepting transcription loss during fine-tuning, the model is constantly corrected on recognizing the lexical content of user utterances. By forming the semantic decision from these supervised transcriptions, the SC is able to directly benefit from the improved AC accuracy in a way that multistage models (such as our baselines) and direct models [13] can not.

6. Conclusion

We have demonstrated the performance of a multistage transformer-based E2E SLU model that is capable of handling the output structural diversity necessary for deployment in a generalized VA setting. We have shown that this approach significantly outperforms various multitask baselines on the hardest slot classification examples characteristic of semantically complex datasets. Furthermore, we demonstrated that these gains in functionality do not come at a cost of performance on simpler SLU benchmarks. We hope for future work further exploring E2E SLU in structurally diverse, semantically complex general VA settings, especially in low-data scenarios.

7. References

[1] G. Tur and R. De Mori, *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons, 2011.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[3] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 6645–6649.

[4] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *2016 IEEE ICASSP*. IEEE, 2016, pp. 4945–4949.

[5] P. Xu and R. Sarikaya, "Contextual domain classification in spoken language understanding systems using recurrent neural network," in *2014 IEEE ICASSP*. IEEE, 2014, pp. 136–140.

[6] S. Ravuri and A. Stolcke, "Recurrent neural network and LSTM models for lexical utterance classification," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[7] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of deep belief networks for natural language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 778–784, 2014.

[8] Y.-C. Tam, Y. Lei, J. Zheng, and W. Wang, "ASR error detection using recurrent neural network language model and complementary asr," in *2014 IEEE ICASSP*. IEEE, 2014, pp. 2312–2316.

[9] R. Voleti, J. M. Liss, and V. Berisha, "Investigating the effects of word substitution errors on sentence embeddings," in *2019 IEEE ICASSP*. IEEE, 2019, pp. 7315–7319.

[10] M. Moore, M. Saxon, H. Venkateswara, V. Berisha, and S. Panchanathan, "Say what? a dataset for exploring the error patterns that two ASR engines make." in *INTERSPEECH*, 2019, pp. 2528–2532.

[11] A. Raghuvanshi, V. Ramakrishnan, V. Embar, L. Carroll, and K. Raghunathan, "Entity resolution for noisy ASR transcripts," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, 2019, pp. 61–66.

[12] H. Wang, S. Dong, Y. Liu, J. Logan, A. K. Agrawal, and Y. Liu, "ASR error correction with augmented transformer for entity retrieval," *Proc. Interspeech 2020*, pp. 1550–1554, 2020.

[13] P. Haghani, A. Narayanan, M. Bacchiani, G. Chuang, N. Gaur, P. Moreno, R. Prabhavalkar, Z. Qu, and A. Waters, "From audio to semantics: Approaches to end-to-end spoken language understanding," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 720–726.

[14] L. Lugosch, B. H. Meyer, D. Nowrouzezahrai, and M. Ravanelli, "Using speech synthesis to train end-to-end spoken language understanding models," in *2020 IEEE ICASSP*. IEEE, 2020, pp. 8499–8503.

[15] E. Palogiannidi, I. Gkinis, G. Mastrapas, P. Mizera, and T. Stafylakis, "End-to-end architectures for ASR-free spoken language understanding," in *2020 IEEE ICASSP*. IEEE, 2020, pp. 7974–7978.

[16] M. Radfar, A. Mouchtaris, and S. Kunzmann, "End-to-End Neural Transformer Based Spoken Language Understanding," in *Proc. Interspeech 2020*. ISCA, 2020, pp. 866–870.

[17] Y. Tian and P. J. Gorinski, "Improving end-to-end speech-to-intent classification with Reptile," *Proc. Interspeech 2020*, pp. 891–895, 2020.

[18] Y.-A. Chung, C. Zhu, and M. Zeng, "Semi-supervised speech-language joint pre-training for spoken language understanding," *arXiv preprint arXiv:2010.02295*, 2020.

[19] S. Kim, G. Kim, S. Shin, and S. Lee, "Two-stage textual knowledge distillation to speech encoder for spoken language understanding," *arXiv preprint arXiv:2010.13105*, 2020.

[20] M. Kim, G. Kim, S.-W. Lee, and J.-W. Ha, "ST-BERT: Cross-modal language model pre-training for end-to-end spoken language understanding," *arXiv preprint arXiv:2010.12283*, 2020.

[21] L. Lugosch, M. Ravanelli, P. Ignoto, V. S. Tomar, and Y. Bengio, "Speech model pre-training for end-to-end spoken language understanding," *Proc. Interspeech 2019*, pp. 814–818, 2019.

[22] Y.-P. Chen, R. Price, and S. Bangalore, "Spoken language understanding without speech recognition," in *2018 IEEE ICASSP*. IEEE, 2018, pp. 6189–6193.

[23] P. Price, "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*, 1990.

[24] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril *et al.*, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," *arXiv preprint arXiv:1805.10190*, 2018.

[25] X. Yang, Y.-N. Chen, D. Hakkani-Tür, P. Crook, X. Li, J. Gao, and L. Deng, "End-to-end joint learning of natural language understanding and dialogue manager," in *2017 IEEE ICASSP*. IEEE, 2017, pp. 5690–5694.

[26] N. Tomashenko, A. Caubrière, Y. Estève, A. Laurent, and E. Morin, "Recent advances in end-to-end spoken language understanding," in *International Conference on Statistical Language and Speech Processing*. Springer, 2019, pp. 44–55.

[27] J. P. McKenna, S. Choudhary, M. Saxon, G. P. Strimel, and A. Mouchtaris, "Semantic Complexity in End-to-End Spoken Language Understanding," in *Proc. Interspeech 2020*. ISCA, 2020, pp. 4273–4277.

[28] H.-K. J. Kuo, Z. Tüske, S. Thomas, Y. Huang, K. Audhkhasi, B. Kingsbury, G. Kurata, Z. Kons, R. Hoory, and L. Lastras, "End-to-End Spoken Language Understanding Without Full Transcripts," in *Proc. Interspeech 2020*. ISCA, 2020, pp. 906–910.

[29] M. Rao, A. Raju, P. Dheram, B. Bui, and A. Rastrow, "Speech to semantics: Improve ASR and NLU jointly via all-neural interfaces," *Proc. Interspeech 2020*, Oct 2020.

[30] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[31] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE ICASSP*. IEEE, 2018, pp. 4774–4778.

[32] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *2015 IEEE ICASSP*. IEEE, 2015, pp. 5206–5210.

[34] T. Wolf, J. Chaumond, L. Debut, V. Sanh, C. Delangue, A. Moi, P. Cistac, M. Funtowicz, J. Davison, S. Shleifer *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.

[35] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 328–339.