Interspeech 2022
18-22 September 2022, Incheon, Korea

Speech2Slot: A Limited Generation Framework with Boundary Detection for Slot Filling from Speech

Pengwei Wang[1], *Yinpei Su*[1,2], *Xiaohuan Zhou*[1], *Xin Ye*[1], *Liangchen Wei*[1], *Ming Liu*[1]
Yuan You[1], *Feijun Jiang*[1]

[1]Alibaba Group,[2]Boston University
{hoverwang.wpw,shiyi.zxh,sunny.yx,liangchen.wlc,grady.lm,youyuan.yy,feijun.jiangfj}@alibaba-inc.com
syinpei@bu.edu

Abstract

Slot filling is an essential component of Spoken Language Understanding. In contrast to conventional pipeline approaches, which extract slots from the ASR output, end-to-end approaches directly get slots from speech within a classification or generation framework. However, classification relies on predefined categories, which is not scalable, and the generative model is decoding in an open-domain space, suffering from blurred boundaries of slots in speech. To address the shortcomings of these two formulations, we propose a new encoder-decoder framework for slot filling, named Speech2Slot, leveraging a limited generation method with boundary detection. We also released a large-scale Chinese spoken slot filling dataset named Voice Navigation Dataset in Chinese (VNDC). Experiments on VNDC show that our model is markedly superior to other approaches, outperforming the state-of-the-art slot filling approach with 6.65% accuracy improvement. We make our code[1] publicly available for researchers to replicate and build on our work.
Index Terms: Speech2Slot, slot filling, limited generation, spoken language understanding

1. Introduction

Spoken Language Understanding (SLU) has attracted much attention with the rapidly growing demand for voice assistants such as Alexa, Siri, and Google Home [1, 2, 3, 4, 5]. The conventional pipeline approaches typically consist of automatic speech recognition (ASR), which converts speech into the underlying text, and natural language understanding (NLU), which learns semantics from the converted text input [6, 7, 8]. The major problem of such approaches is that NLU suffers from the upstream ASR errors, which set an accuracy upper bound of the entire system.

Over the last few years, end-to-end SLU approaches that directly infer semantic meaning from speech have been proposed to eliminate the error propagation problem [9, 10, 11, 12, 13, 14, 15]. Several end-to-end approaches [12, 16] formulate the slot filling task as an intent classification problem by flatting slot labels into different intents. Other studies[15, 17, 18, 19, 20] formulate slot filling as a generation task, where the speech is used as the input to generate slot sequence using a sequence-to-sequence (seq-to-seq) framework.

However, there are several problems with these two formulations. (1) It is found that there are millions of slot values in most vertical domains, such as music, movie, and locations. The classification framework relies on predefined categories, which is not scalable. (2) For the seq-to-seq framework, the entire vocabulary will be searched at each decoding step, thus the produced slots are not well controlled for subsequent tasks. (3) The entire information of the input is helpful for decoding in general seq-to-seq frameworks, such as machine translation and end-to-end ASR. However, the speech contains much redundant information for slot filling. It is hard to distinguish the slot and redundant information from the speech in slot decoding. (4) A position where the slot sound is clear could be in the former, middle, or latter of the speech. Thus, the uni-direction decoding would search a wrong path once the sound is not clear at first.

To address these problems, we propose a new encoder-decoder framework called Speech2Slot. More specifically, we formulate slot filling as a limited generation task leveraging the trie tree to restrict the decoding space. Furthermore, we propose a Gated Attention module to distinguish the slot information from speech for detecting slot boundaries. Finally, to choose an appropriate position to start decoding, we design a tri-direction decoding method compared to uni-direction decoding. These modules form our Speech2Slot framework. It directly takes the phoneme posterior of speech as inputs and generates the slots under control.

To the best of our knowledge, almost all existing SLU datasets are English and minor in scale (such as SLURP, ATIS, and SNIPS). We mainly focus on the large-scale slot filling scene, which has no ready-made dataset. Thus, we release a large-scale Chinese spoken slot filling dataset named Voice Navigation Dataset in Chinese (VNDC). VNDC contains 820k TTS training samples, 10k TTS testing samples, 2k human testing samples, and 830,000 slots. Experiments on VNDC show that our approach outperforms the end-to-end slot filling approach with over 6.65% and 6.28% accuracy improvement on the human test set and TTS test set, respectively.

The contributions of our paper are as follows:

- we introduce a limited generation method leveraging trie for spoken slot filling.
- We propose a Gated Attention module to detect the slot boundaries from the speech.
- We design a tri-direction decoding method to decode slots.
- We release a large-scale Chinese dataset for spoken slot filling.

[1]https://github.com/eehover/speech2slot

10.21437/Interspeech.2022-11347

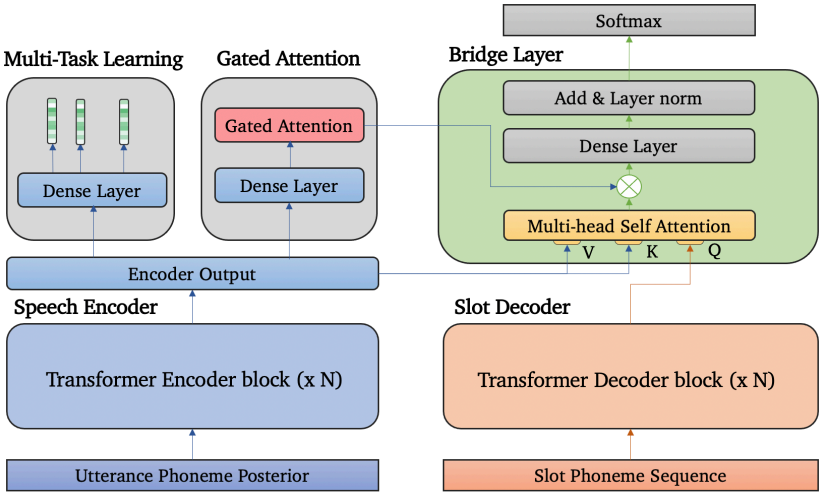

Figure 1: *Speech2Slot Model*

2. Related Work

There is plenty of work on conventional pipeline Spoken Language Understanding (SLU) [6, 7, 8, 11, 21, 22, 23, 24, 25]. In the slot filling (SF) part of SLU, most studies focus on extracting slots from the text [26, 27, 28, 29, 30]. The ASR error easily propagates to the SF module. In the past few years, end-to-end SLU models are proposed to address the error propagate problem [10, 12, 13, 14, 31, 32]. The end-to-end SLU models leverage a unified neural network to replace the ASR and NLU model, where raw waveforms or the acoustic features are directly used as the inputs to infer the NLU result.

Most of these studies aim to get the sentence-level representation of the input speech, which can only be used in domain or intent classification. To apply the end-to-end modeling approach in SF, several studies [12, 16] formulate the SF as a classification problem. In [16], the slot labels with combinations of slot values are flatted as different intents. The classification framework can produce discrete results, but it relies on predefined categories, which is not scalable. It is hard to use this method on the large-scale and long-tail slot filling scene. In the meanwhile, a new slot can not be predicted using a trained model.

Other studies [15, 17, 18, 19, 20] regard SF as a generation task leveraging an encoder-decoder architecture. In [19, 20], the slots are directly decoded in the encoder-decoder network by adding special symbols to denote the start and end of the slot. In [15], the intent, slot types, and slot values are decoded in sequence. However, the generation framework can produce arbitrary results, which is not in existing slot database. The generated slots cannot be used directly in subsequent modules, such as the music slot or video slot. Additionally, the entire information of the speech is used for decoding, and the slot boundary is not well detected in previous studies. Thus, to address these challenges, we propose the Speech2Slot, leveraging a limited-generation method to keep the generation results in the slot database and integrating a gated attention module to detect the slot boundary.

3. Speech2Slot

The entire framework consists of two major components: Acoustic Model (AM) and Speech2Slot. The AM, which generates the phoneme posterior distribution, is trained with a CTC loss [33] over the ground-truth phoneme sequence. The Speech2Slot takes the phoneme posterior as input. Such implementation makes Speech2Slot independent from the AM implementation conditioned on the phoneme posterior. Therefore, Speech2Slot is less sensitive to the choice of the AM implementation as long as it provides proper phoneme posterior with comparable accuracy. In this paper, we use an open-source code[2] to train the AM.

3.1. Encoder-Decoder Framework

As shown in Figure 1, Speech2Slot is based on the transformer encoder-decoder architecture. The input of the encoder is the phoneme posterior. We integrated two submodules in the encoder: multi-task learning and gated attention. To minimize the overfitting effect, we add a masked language model target in the encoder to perform multi-task learning. We followed [32] to randomly mask 10%-15% of the frames from the phoneme posteriors. At the encoder hidden vectors of the masked frames, we add a dense layer to predict the masked frame. The objective function is the cross-entropy between the original phoneme posterior frame and the predicted ones. Additionally, since there is much redundant information in the speech, in order to detect the slot boundary, we also integrated gated attention in the encoder, detailed in 3.2.

The input of the decoder is the slot phoneme sequence, which is obtained according to a word-to-phoneme map table. To restrict the generated slots, we limit the decoding process during inference leveraging a trie structure, introduced in 3.3. Additionally, to choose the appropriate position to start decoding, we design a tri-direction decoding method [34] to further improve

[2]https://github.com/swimmingCreative/DeepSpeechRecognition-master

model performance, detailed in 3.4.

3.2. Gated Attention

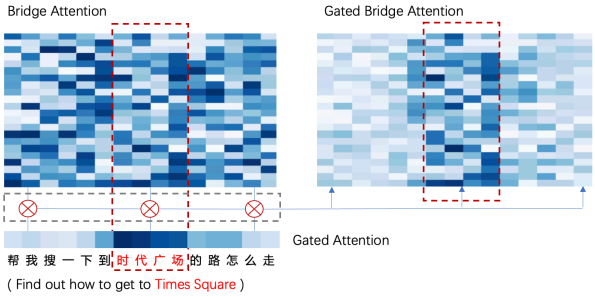

Figure 2: *Gated Attention*

In general encoder-decoder tasks, almost the entire input sequence is necessary, e.g. machine translation. However, in the spoken slot filling task, there are many noises (silent frames and nonsense) or templates ("*I want to...*") in the input speech, which are not helpful for the slot filling. Inspired by object detection in computer vision, we prefer to detect the slot boundaries to boost performance. Therefore, we introduced a Gated Attention module and integrated it into the encoder-decoder attention, as shown in Figure 2.

The encoder output is passed through a fully connected layer consists of a linear transformation with a sigmoid activation function to compute gated attention:

$$Gated\ Attention(GA) = Sigmoid(W_g E_{out}) \quad (1)$$

Where $W_g \in \mathbb{R}^{1\times d}$ is the projection matrix and $E_{out} \in \mathbb{R}^{d\times L}$ is the encoder output. L is the length of input sequence and d is the hidden size. The gated attention $GA \in \mathbb{R}^{1\times L}$ is a vector with the dimension L, indicating which position might be a slot. The GA will be integrated into the encoder-decoder attention (denoted as bridge attention). The bridge attention is standard encoder-decoder attention, which is calculated as:

$$Bridge\ Attention(BA) = \frac{QK^T}{\sqrt{d}}V \quad (2)$$

The BA is used to compute the gated bridged attention by element-wise product with GA through broadcasting:

$$Gated\ Bridge\ Attention(GBA) = GA * BA \quad (3)$$

The gated bridge attention $GBA \in R^{d\times L}$ will be regarded as the new encoder-decoder attention.

3.3. Limited Generation

Our goal is that the output slots must be in our database. For example, a song slot is predicted to belong to the music domain. Thus, we introduce the trie structure to limit the decoding space. We first use the slots database to build a trie tree, shown in Figure 3. Then, it will only consider words in the trie as candidates when the model generates the next token. Particularly, the trie is only used in the inference phase. Therefore, we could build different tries for diverse situations based on the same pre-trained model. Also, we can update the tries without retraining the model when new slots appear.

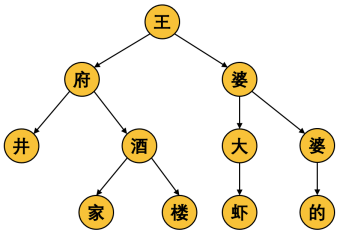

Figure 3: *Trie Structure*

3.4. Tri-direction Decoding

We observe that since the generative models produce tokens based on the previous output, the tokens generated earlier have a profound effect on the latter. Thus, the starting position of decoding is significant. Decoding from a position where the sound is clear will be beneficial. However, the sound could be apparent in the former, middle, or latter position. Thus, we design a tri-direction decoding method: forward, backward, and middle to both sides, followed [34, 35]. Here, a token is formulated as a phoneme.

We implement backward decoding by reversing the sequence as the optimization objective. For middle decoding, we need first to determine where to start decoding. Hence we designed a module to predict the middle word:

$$O_{mid\_word} = Sigmoid(W_m(E_{out}GA^T)) \quad (4)$$

The dot product $E_{out}GA^T$ is regarded as a weighted average operation to get a sentence-level representation, and $W_m \in R^{d_{vocab}\times d}$ is a transformation matrix to project the output to vocab dimension. We choose those tokens of the input sequence which are in the corresponding slots as true labels. This module will only be used in the inference phase. During training, we will choose the middle word of the slot as the start position of middle decoding.

We use beam search during inference with a beam size of 10 for forward and backward decoding. For middle decoding, we first decode one side tokens, and then use the forward or backward module to decode the other side. Finally, we add the beam scores of the same results decoding in different directions and choose the highest score as the predicted slot. The key to effective middle decoding is to choose the right start words. The model could avoid starting from the pattern words precisely because the slot boundary could be effectively detected by gated attention.

3.5. Dataset

This section describes the preparation of the Voice Navigation Dataset in Chinese (VNDC). First, we collect more than 830,000 place names in China, such as *The Palace Museum*, *Great Wall of Badaling*, and *Jishuitan Hospital*. To generate the navigation queries in training data, we also collect more than 100 query patterns (such as navigation to [*location*], the route to [*location*], search [*location*]). We fill out the query patterns with places to generate all the queries.

Next, we use an open-source TTS tool[3] to generate the speech files. The TTS tool provides several variant articulation types, such as men, women, background music, and underwater, which increase the data diversity significantly. In this way, we generate 820k speech-slot pairs for training and 10k speech-slot pairs for testing. To build real-person testing data, we also invite five volunteers to record 2,000 audio clips. To valid the generalization ability, the volunteers can freely describe the place without using the exiting 100 query patterns. We have released the entire training and testing data[4].

3.6. Experimental Settings

Acoustic Model. We train the acoustic model with four open-source ASR datasets, THCHS-30, Aishell, Free ST Chinese Mandarin Corpus, and Primewords Chinese Corpus Set[5].

Speech2Slot. We use a transformer architecture with 4 layers. Each block uses 6 self-attention heads and hidden size 384. We separately set the max input length of the encoder and decoder to be 40 and 10. All the model updates use a batch size of 256 and a Adam optimizer with learning rate 1.5e-4.

3.7. Experimental Methods

Generation-SF(Gen-SF) [18]. To verify the performance of the end-to-end slot filling model, we implement an end-to-end SF model with the generative framework, which is denoted as Generation-SF. Since the generation process is not limited within this model, the output is not necessarily in our database. For a fair comparison, we use the Levenshtein distance to match the closest slot in our database, denoted as Generation-SF-search.

Convention-SF(Con-SF) [30]. We implement a conventional pipeline SLU approach by cascading the ASR model with an SF model. The ASR model is implemented by the same open-source code, which is used to generated AM as mentioned above. The SF is implemented by a transformer stacked by a CRF. We denote this method as Convention-SF. Since there are errors in ASR results, we also use the Levenshtein distance to match the closest slot in our database, denoted as Convention-SF-search.

Classification-SF(Cla-SF) [13]. We also implement an end-to-end SF model with a classification framework. However, due to the fact that the number of categories is too much and even exceeds the number of training samples, the model is hard to converge. We do not put it in the results table.

4. Results

4.1. Main Results

We compare different approaches on the VNDC. The results of the experiment are shown in Table 1. Because the training data of ASR or AM comes from general domains and the decoding slots in the VNDC is extremely large scale, the result of Convention-SF is poor. The Levenshtein distance matching (Convention/Generation-SF-search) can increase the performance. As the results show, our proposed Speech2Slot model significantly outperforms the conventional slot filling approaches by a large margin on both TTS and human test sets, proving that our method could resolve the ASR error propagation. Furthermore, the Speech2Slot also outperforms the end-to-end generation method with 6.26% and 6.65% accuracy improvement separately, which means our method is also more effective than previous end-to-end models.

Table 1: *Slot filling accuracy comparison*

Model	TTS Test	Human Test
Con-SF	14.12%	8.40%
Con-SF-search	49.48%	48.80%
Gen-SF	40.16%	11.70%
Gen-SF-search	88.40%	72.15%
Speech2Slot	**94.68%**	**78.80%**

Table 2: *Ablation Studies*

Model	TTS	Human
Speech2Slot Baseline	84.20%	58.40%
+ Gated Attention	84.28%	60.10%
+ Bi-dec	93.89%	74.05%
+ Tri-dec	94.21%	75.30%
+ Gated Attention + Bi-dec	94.39%	77.35%
+ Gated Attention + Tri-dec	**94.68%**	**78.80%**

4.2. Ablation Studies

We perform ablation experiments that quantify the individual contribution of each of the design choices for Speech2Slot. The results are shown in Table 2. The Speech2Slot with gated attention and tri-decoding outperformed the baseline with 10.48% and 20.4% respective on two test sets. The results verify that the gated attention could detect the slot boundary. Also, it proves the tri-direction decoding could tackle different situations such as the sound is clear either in the former, middle, or latter position.

5. Conclusion

In this paper, we propose Speech2Slot, a practical way for spoken slot filling with large-scale slot values. We formulate the slot filling as a limited generation task, where the generated slots will be limited in the slots database. We also designed a gated attention module to detect the slot boundary and introduce tri-direction decoding for better performance. In addition, we release a large-scale Chinese speech-to-slot dataset. The experiment results show that our proposed Speech2Slot significantly outperforms the pipeline SLU approach and the end-to-end slot filling approach. Current work mainly focuses on the large-scale slots, which have appeared in the slot database. In future work, we will make the Speech2Slot have the capability to generate new appearing slots. Furthermore, the multi-slots filling is also an important and practical research direction.

[3]https://www.xfyun.cn/

[4]http://www.speech2slot.com/dataset

[5]http://www.openslr.org/

6. References

[1] G. Tur, L. Deng, D. Hakkani-Tur, and X. He, "Towards deeper understanding: Deep convex networks for semantic utterance classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012.

[2] P. Xu and R. Sarikaya, "Contextual domain classification in spoken language understanding systems using recurrent neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014.

[3] K. Yao, G. Zweig, M.-Y. Hwang, Y. Shi, and D. Yu, "Recurrent neural networks for language understanding," in *Interspeech*, 2014.

[4] A. Siddhant, A. Goyal, and A. Metallinou, "Unsupervised transfer learning for spoken language understanding in intelligent agents," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2019.

[5] L. Zhao and Z. Feng, "Improving slot filling in spoken language understanding with joint pointer and attention," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[6] A. Coucke, A. Caulier, A. Ball, C. L. Dec, and T. Bluche, "Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces," in *arXiv:1805.10190v3*, 2018.

[7] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may i help you?" in *Speech Communication*, 1997.

[8] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkanitur, and X. He, "Using recurrent neural networks for slot filling in spoken language understanding," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2015.

[9] Y.-P. Chen, R. Price, and S. Bangalore, "Spoken language understanding without speech recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[10] 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)*, 2018.

[11] N. Tomashenko, A. Caubrière, Y. Estève, A. Laurent, and E. Morin, "Recent advances in end-toend spoken language understanding," in *International Conference on Statistical Language and Speech Processing*, 2019.

[12] D. Serdyuk, Y. Wang, C. Fuegen, A. Kumar, B. Liu, and Y. Bengio, "Towards end-to-end spoken language understanding," in *Interspeech*, 2019.

[13] L. Lugosch, M. Ravanelli1, P. Ignoto, V. S. Tomar, and Y. Bengio, "Speech model pre-training for end-to-end spoken language understanding," in *arXiv:1904.03670v1*, 2019.

[14] R. Price, "End-to-end spoken language understanding without matched language speech model pretraining data," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[15] Y. Qian, X. Bian, Y. Shi, N. Kanda, L. Shen, Z. Xiao, and M. Zeng, "Speech-language pre-training for end-to-end spoken language understanding," in *arXiv:2102.06283v1*, 2021.

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

[17] Y.-S. Chuang, C.-L. Liu, H.-Y. Lee, and L. shan Lee, "Speechbert: An audio-and-text jointly learned language model for end-to-end spoken question answering," in *Interspeech*, 2020.

[18] V. Pelloin, N. Camelin, A. Laurent, R. De Mori, A. Caubrière, Y. Estève, and S. Meignier, "End2end acoustic to semantic transduction," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[19] S. Ghannay, A. Caubriere, Y. Esteve, A. Laurent, and E. Morin, "End-to-end named entity extraction from speech," in *arXiv:1805.12045*, 2018.

[20] H. Yadav, S. Ghosh, Y. Yu, and R. R. Shah, "End-to-end named entity recognition from english speech," in *Interspeech*, 2020.

[21] R. De Mori, F. Bechet, D. Hakkani-Tur, M. McTear, G. Riccardi, and G. Tur, "Spoken language understanding," *IEEE Signal Processing Magazine*, vol. 25, no. 3, pp. 50–58, 2008.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the NAACL-HLT*, 2019.

[23] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," in *Lingvisticae Investigationes*, 2007.

[24] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *arXiv:1802.05365*, 2018.

[25] A. Baevski, S. Edunov, Y. Liu, L. Zettlemoyer, and M. Auli, "Cloze-driven pretraining of self-attention networks," in *arXiv:1903.07785*, 2019.

[26] C. Parada, M. Dredze, and F. Jelinek, "OOV sensitive named-entity recognition in speech," in *Interspeech*, 2011.

[27] Y. Jiang, C. Hu, T. Xiao, C. Zhang, and J. Zhu, "Improved differentiable architecture search for language modeling and named entity recognition," in *Proceedings of the 2019 Conference on EMNLP-IJCNLP*, 2019.

[28] I. Cohn, I. Laish, G. Beryozkin, G. Li, I. Shafran, I. Szpektor, T. Hartman, A. Hassidim, and Y. Matias, "Audio de-identification: A new entity recognition task," in *arXiv:1903.07037*, 2019.

[29] J. Strakova, M. Straka, and J. Hajic, "Neural architectures for nested ner through linearization," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[30] H. Yan, B. Deng, X. Li, and X. Qiu, "Tener: Adapting transformer encoder for named entity recognition," in *arXiv:1911.04474v3*, 2019.

[31] Y. Huang, H. Kuo, S. Thomas, Z. Kons, K. Audhkhasi, B. Kingsbury, R. Hoory, and M. Picheny, "Leveraging unpaired text data for training end-to-end speech-to-intent systems," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[32] P. Wang, L. Wei, Y. Cao, J. Xie, and Z. Nie, "Large-scale unsupervised pre-training for end-to-end spoken languageunderstanding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[33] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006.

[34] S. Mehri and L. Sigal, "Middle-out decoding," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems (ICML)*, 2018.

[35] K. Al-Sabahi, Z. Zuping, and Y. Kang, "Bidirectional attentional encoder-decoder model and bidirectional beam search for abstractive summarization," in *arXiv*, 2018.