TOWARDS REDUCING THE NEED FOR SPEECH TRAINING DATA TO BUILD SPOKEN LANGUAGE UNDERSTANDING SYSTEMS

Samuel Thomas, Hong-Kwang J. Kuo, Brian Kingsbury, George Saon

IBM Research AI

ABSTRACT

The lack of speech data annotated with labels required for spoken language understanding (SLU) is often a major hurdle in building end-to-end (E2E) systems that can directly process speech inputs. In contrast, large amounts of text data with suitable labels are usually available. In this paper, we propose a novel text representation and training methodology that allows E2E SLU systems to be effectively constructed using these text resources. With very limited amounts of additional speech, we show that these models can be further improved to perform at levels close to similar systems built on the full speech datasets. The efficacy of our proposed approach is demonstrated on both intent and entity tasks using three different SLU datasets. With text-only training, the proposed system achieves up to 90% of the performance possible with full speech training. With just an additional 10% of speech data, these models significantly improve further to 97% of full performance.

Index Terms— Spoken language understanding, end-to-end models, RNN Transducers.

1. INTRODUCTION

A major hurdle in building end-to-end spoken language understanding system that can process speech inputs directly [1–15] is the limited amount of available speech training data with SLU labels. To circumvent this issue, past approaches have synthesized speech using text-to-speech (TTS) systems or shared network layers with text based classifiers [8]. With the TTS approach, while additional processing resources have to be assembled to process and synthesize the text, with the shared classifier approach, existing models often have to be reconfigured and retrained to accommodate changes in network architecture and inputs. It would hence be useful to have a single E2E model that can process both speech and text modalities, such that SLU models can be effectively trained and adapted on both speech and text data.

In this work, we build on our previous work [16] that constructed E2E SLU systems using RNN Transducer models. RNN-T models typically consist of three different sub-networks: an encoder network, a prediction network, and a joint network [17]. The encoder or transcription network produces acoustic embeddings, while the prediction network resembles a language model in that it is conditioned on previous non-blank symbols produced by the model. The joint network combines the two embedding outputs to produce a posterior distribution over the output symbols. This architecture elegantly replaces a conventional ASR system composed of separate acoustic model, language model, pronunciation lexicon, and decoder components, using a single end-to-end trained, streamable, all-neural model that has been widely adopted for speech recognition [18–22]. Given their popularity, and the fact that RNN-T models can naturally handle more abstract output symbols such as ones marking speaker turns [21], in our previous work we explored the extension of these models to SLU

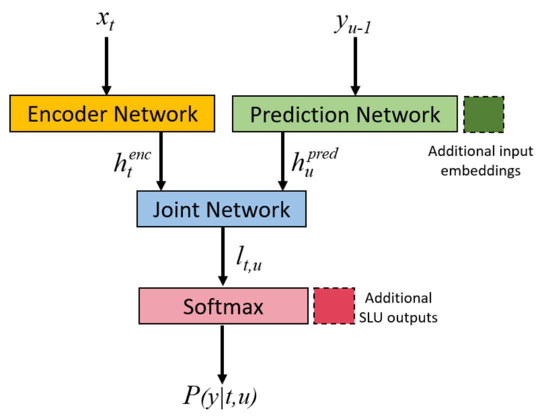

Fig. 1. Architecture of a pre-trained ASR RNN-T model with additional input embeddings in the prediction network and outputs in the joint network for SLU.

tasks, building on recent advances with RNN-T models for speech recognition as described in [23].

Starting from a pre-trained RNN-T ASR model, we construct an SLU model by adapting the ASR model into a domain-specific SLU model. The new SLU labels are integrated by modifying the joint network and the embedding layer of the prediction network to include additional symbols as shown in Fig 1. The new network parameters are randomly initialized, while the remaining parts are initialized from the pre-trained ASR network. The initial SLU model is then further trained using paired speech and text data with transcripts and SLU labels. In this work we develop a novel technique of training SLU models not only with speech paired with transcripts and SLU labels, but also on text-only data annotated with SLU labels. In contrast to prior work, the text-only data is used directly without having to synthesize it using a TTS system.

2. TEXTOGRAMS

In order to train RNN-T based SLU models with text-only data we propose a new feature representation for text and training framework for using it. First, an ASR model is pretrained using both standard speech features and the novel text features, called *textograms*. Subsequently, the ASR model is adapted as an SLU model.

Textograms are constructed to be frame-level representations of text, similar to posteriograms, which are softmax posterior outputs of a trained neural network acoustic model. However, because they are constructed from ground truth text, textograms use 1-hot encodings. For example, given an input text *“ideas”*, graphemic textogram features are constructed by first splitting the word into its constituent graphemes, *"i", "d", "e", "a", "s"*. Each symbol is then allowed to span a fixed time duration, four frames in this case, to create a 2-dimensional representation as shown in Fig. 2. Once constructed in this fashion, these representations are used along with traditional log-mel speech features to train RNN-T models. Because textograms have the same frame level construction as speech features, they can be integrated into an existing RNN training framework by simply stacking them along with traditional speech features: training samples for speech features have the textogram features set to 0.0, and conversely training samples for text features have the speech features set to 0.0.

To allow the model to learn robustly from the textogram representations, variabilities can be added to this representation. These choices include:

1. **Label masking**: To allow the model to learn useful n-gram sequences instead of blindly memorizing the text, active entries of the textogram representation can be randomly dropped. The rate of label masking is a parameter that can be empirically selected.

2. **Label confusions**: The acoustic confusion among various speech sounds can be introduced into the textogram by substituting various labels with their confusable sounds e.g., *p* and *b*

3. **Variable label duration**: The length of each linguistic unit can be varied to model real durations in the speech signal. In Fig. 2 we use four frames per symbol.

4. **Modeling pronunciations**: The input textogram may include different "sounds-like" sequences for a given target output. For example, the target *Miami*, may be associated with textogram sequences, *Miami*, *my Amy*, or *mee Amy*.

5. **Multiple linguistic symbol sets**: The symbol set used with textograms can be different from the output symbol set for the ASR model. For example, phonetic targets can be used at the RNN-T's output while graphemes are used for textograms.

In this work, we use a fixed label duration for various text symbols along with label masking, to construct textogram features.

3. CONSTRUCTING RNN-T MODELS FOR SLU

3.1. Pre-trained ASR models

Using notation from [17], an RNN-T models the conditional distribution $p(\mathbf{y}|\mathbf{x})$ of an output sequence $\mathbf{y} = (y_i, \ldots, y_U) \in \mathcal{Y}^*$ of length $\mathcal{U}$ given an input sequence $\mathbf{x} = (x_i, \ldots, x_T) \in \mathcal{X}^*$ of length T. In ASR, $\mathbf{x}$ is a sequence of continuous multidimensional speech features and $\mathbf{y}$ is a sequence of discrete output symbols, like the grapheme set of the language being modelled by the network. The distribution $p(\mathbf{y}|\mathbf{x})$ is further expressed as a sum over all possible alignment probabilities between the input and output sequences. To create alignments between the unequal length input-output sequences, it is necessary to introduce an additional BLANK symbol that consumes one element of the input sequence and generates a null output. The probability of a particular alignment is computed in terms of embeddings h^{enc} of the input sequence computed by the encoder network and embeddings h^{pred} of the output sequence computed by the prediction network. The joint network combines these embeddings to produce a posterior distribution over the output symbols. Within this framework, RNN-T models are trained to minimize $-\log p(\mathbf{y}|\mathbf{x})$, the negative log-likelihood loss, via an efficient forward-backward algorithm with $T \times U$ complexity for both loss and gradient computation.

The model is now trained with both speech inputs, represented using speech features, and text inputs, represented using textograms.

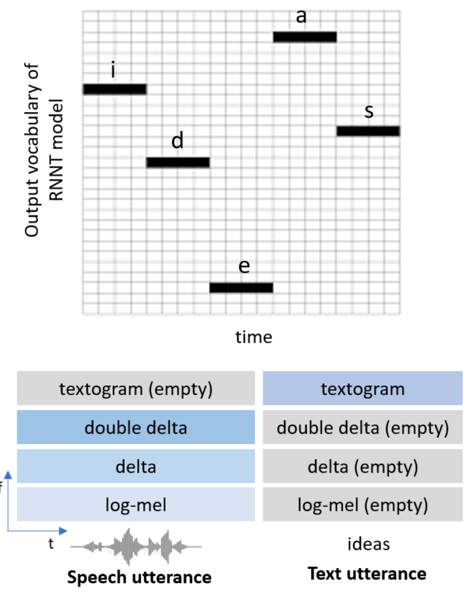

Fig. 2. Schematics of the textogram representation for '*ideas*' (top), feature inputs used to train RNN-T models corresponding to speech and text (bottom).

This effectively doubles the number of training examples, so from one perspective the use of textograms in training is a form of data augmentation. For samples represented by speech features, we extract log-mel features, along with delta and double-delta features. The dimensions in the input corresponding to textogram features are set to 0.0, as shown in Fig. 2. To improve the robustness of the speech training, sequence noise injection [24] and SpecAugment [25] are applied to the speech features. With sequence noise injection, attenuated features from a randomly selected training utterance are added with a given probability to the features of the current training utterance. SpecAugment, on the other hand, masks the spectrum of a training utterance with a random number of blocks of random size in both time and frequency. For the text data, we extract textogram features corresponding to each text transcript and apply label masking with a mask probability of 25%. As shown in Fig. 2, the dimensions corresponding to speech features are set to 0.0 for training samples that use textogram features. By integrating text inputs into the training pipeline, the RNN-T model's transcription network is now trained as a single encoder for two modalities: speech and text. With this joint training, the transcription network produces similar embeddings for both speech and text that can be further used along with a prediction and joint network that are shared by both modalities.

3.2. Adapting RNN-T models for SLU

Once an RNN-T ASR model has been trained on both speech and text, we adapt this pre-trained base model into an SLU model with both speech and text data, following a training procedure similar to that described above for ASR. The SLU training is done as follows:

1. **Creating an initial SLU model**: In the ASR pre-training step, the targets are graphemic/phonetic tokens only, but for SLU adaptation the targets also include semantic labels. Starting with an ASR model, the new SLU labels are integrated by modifying the joint network and the embedding layer of the prediction network to include additional

output symbols. The new network parameters are randomly initialized, while the remaining parts are initialized from the pre-trained network.

2. **Training on text-only SLU data**: Prior to the adaptation process, the text-only SLU data is converted into textogram based features. Note that the textogram does not represent the SLU targets, but only the speech transcripts. The RNN-T model is then adapted using these features to predict various SLU labels. While adapting an RNN-T with text-only data, we keep the transcription network fixed and adapt the prediction and joint networks. This ensures that the model is still acoustically robust while being able to effectively process data from the new domain.

3. **Training on speech and text SLU data**: When both speech and text data are available to train an SLU model, the RNN-T is adapted using both types of input, much in the same way that the model is pre-trained using both types of input. However, when a speech sample is presented during adaptation on mixed data, the entire network is updated, but when a text sample is presented, only the prediction and joint networks are updated. This allows the model both to adapt to new acoustic conditions and to learn to process SLU targets.

4. EXPERIMENTS AND RESULTS

4.1. Training the base ASR model

The RNN-T models used in our experiments are trained using various telephone speech corpora, including Switchboard, Fisher, and proprietary data. Each RNN-T model has three sub-networks as illustrated in Fig 1. The encoder or transcription network contains 6 bidirectional LSTM layers with 640 cells per layer per direction. The prediction network is a single unidirectional LSTM layer with only 1024 cells. The joint network projects the 1280-dimensional stacked encoder vectors from the last layer of the transcription net and the 1024-dimensional prediction net embedding each to 256 dimensions, combines them multiplicatively, and applies a hyperbolic tangent. Finally, the output is projected to 42 logits, corresponding to 41 characters plus BLANK, followed by a softmax. More details on training settings and design choices can be found in [23]. The RNN-T SLU models are trained using 40-dimensional, global mean and variance normalized log-Mel filterbank features, extracted every 10 ms. These features are augmented with Δ and $\Delta\Delta$ coefficients, every two consecutive frames are stacked, and every second frame is skipped, resulting in 240-dimensional vectors every 20 ms. These speech features are finally appended with empty textogram features to create 324 dimensional vectors.

In addition to the speech data, we use all the available text transcripts as training data as well. These transcripts are first converted into textogram features before they are shuffled along with speech utterances to train an RNN-T model on both modalities. Each text input is split using the same grapheme set modelled at the outputs of the RNN-T. We use a 4 frame duration for each symbol and randomly mask out 25% of the inputs to prevent the model from overfitting on the text inputs. Similar to the speech utterances, textogram features corresponding to text utterances are finally appended with empty speech features. The RNN-T models are trained for 20 epochs using an AdamW optimizer. Once trained, we measure the effectiveness of this base ASR model on the commonly used Hub5 2000 Switchboard (SWB) and CallHome (CH) test sets. The model has a very competitive word error rate (WER) of 6.2% and 10.5% respectively on these test sets.

4.2. Development of SLU models

In the following experiments, we adapt the pre-trained ASR model to build SLU models in various settings. We use three SLU datasets for our experiments.

A. Dialog action recognition on the HarperValleyBank dataset. In our first set of experiments, we adapt the baseline ASR model to the HarperValleyBank corpus [26]. The dataset is a public domain corpus with spoken dialogs that simulate simple consumer banking interactions between users and agents. There are 1,446 human-human conversations between 59 unique speakers. We focus on the dialog action prediction task in this work. In this task the goal is to predict one or more of 16 possible dialog actions for each utterance. The training set contains 1174 conversations (10 hours of audio, 15K text transcripts) and the test set has 199 conversations (1.8 hours hours of audio). As described earlier, once an initial SLU model is constructed, the model is adapted with domain specific SLU data.

SP	F1 SP	F1 SP+TXT
0 hrs (0%)	–	45.05
1 hrs (10%)	47.88	53.84
2.5 hrs (25%)	51.42	54.02
5 hrs (50%)	53.13	55.33
10 hrs (100%)	53.57	54.95

Table 1. Dialog action recognition (F1 scores) on HarperValleyBank Corpus. For each experiment we use all the training transcripts annotated with dialog action labels as text-only data.

Table 1 shows the various experiments and results on this dataset. In our first experiment, we train the SLU model with text-only data annotated with SLU labels. The text data is converted into textogram features and then used to adapt the prediction and joint network of the SLU RNN-T. This text-only adaptation produces a model that performs at 45.05 F1 score. Next, using only speech data in varying quantities, we adapt the RNN-T model (including the transcription network component) to show that the performance improves from 47.88 F1 with 10% of speech data to 53.57 F1 with all of the available speech training data. Finally, we use varying quantities of speech data along with all the text data to adapt the RNN-T model, obtaining models that achieve F1 scores of 53.84 or better.

We can make a number of observations based on these results. First, with no speech data at all, the model is able to process the SLU test set at nearly 82% of full speech performance (53.57 F1 score). These results demonstrate the usefulness of our proposed method for constructing SLU models with just text data. Second, only a small amount of speech data is required to train a strong SLU model if all of the text data is used. With 10% of speech data, adding text data improves model performance to 53.84 F1, which is at 98% of the performance with full speech (53.57 F1 score). This result shows that while the text data provides information to learn the SLU targets (45.05 F1), acoustic robustness to this new domain comes from the speech training (53.84 F1). Finally, as the amount of speech data is increased, we see very modest improvements as the model has already learnt to process SLU targets from the text inputs and adapted to acoustic conditions of the new domain.

B. Intent Recognition on Call Center data. The second data set is based on an internal data collection consisting of call center recordings of open-ended first utterances by customers describing the reasons for their calls [27]. The 8kHz telephony speech data was manually transcribed and labeled with one of 29 intent classes. The corpus contains real, spontaneous utterances from customers, not crowdsourced scripted or role-played data, and it includes a wide variety of ways that customers naturally described their intents. The training data consists of 19.5 hours (22K utterances) of speech that was first divided into a training set of 17.5 hours and a held-out set of 2 hours. A separate data set containing 5592 sentences (5h, 40K words) was used as the final test set [8]. This task contains only intent labels and does not have any labeled semantic entities.

SP	Acc. SP	Acc. SP+TXT
0 hrs (0%)	–	76.97
1.7 hrs (10%)	72.46	88.34
4.4 hrs (25%)	82.67	89.32
8.7 hrs (50%)	87.05	89.56
17 hrs (100%)	89.06	89.59

Table 2. Intent recognition accuracy (%) on the call center corpus. For each experiment we use all the training transcripts annotated with intent labels as text-only data.

Table 2 shows the results of training an SLU model for intent recognition on this dataset. With just text-only training, the model achieves a intent recognition accuracy of 76.97%, which is about 86% of full performance with speech SLU data (89.06% intent recognition), similar to previous experiments on the HarperValleyBank dataset. With an additional 10% of speech data, the model performance rises to 88.34%, which is 99% of full performance with speech. As the amount of speech data increases, although we observe slight improvements, the model is clearly able to learn about the SLU domain and novel conditions already with very limited amounts of transcribed data and large amounts of text-only SLU data. These results clearly show the benefit of our approach in many practical SLU settings where there are large amounts of text-only training data and limited or almost no speech training data. With our proposed method, an SLU system can be effectively bootstrapped with just the text-only data and then improved to close to full performance with very limited amounts of speech data. This helps to significantly reduce the cost overhead in building speech based E2E SLU systems in terms of data collection and additional resources like TTS systems.

C. Entity and Intent Recognition on ATIS. In our final set of experiments we use the ATIS [28] training and test sets: 4976 training utterances from Class A (context independent) training data in the ATIS-2 and ATIS-3 corpora and 893 test utterances from the ATIS-3 Nov93 and Dec94 data sets. The test utterances comprise about 1.5 hours of audio from 55 speakers. The data was originally collected at 16 kHz, but is downsampled to 8 kHz to match the base telephony model. The ATIS task includes both entity (slot filling) and intent recognition. Similar to previous experiments, we first conduct intent recognition experiments on the ATIS corpus.

SP	Acc. SP	Acc. SP+TXT
0 hrs (0%)	–	90.59
0.95 hrs (10%)	84.77	92.16
2.4 hrs (25%)	88.47	93.06
4.8 hrs (50%)	92.50	95.07
9.6 hrs (100%)	96.42	96.75

Table 3. Intent recognition accuracy (%) on the ATIS corpus. For each experiment we use all the training transcripts annotated with intent labels as text-only data.

Table 3 shows the intent recognition results of various SLU systems trained on the ATIS corpus. Similar to previous results, the text-only model is able to perform relatively well at 93% of the full speech performance (90.59% vs. 96.42% intent recognition accuracy). Although adding 10% of speech data improves performance, we only achieve almost 99% of full performance with 50% of additional speech data. We hypothesize ithat this is because the ATIS test set is quite varied in terms of speaker coverage compared to the other test sets, and hence requires more domain specific speech data. Regardless, the model is able to learn about the SLU domain and SLU targets with just the available text data.

SP	F1 SP (+aug)	F1 SP+TXT (+aug)
0 hrs (0%)	–	85.95
0.95 hrs (10%)	46.25 (82.74)	91.07 (90.82)
2.4 hrs (25%)	75.74 (89.21)	91.24 (91.86)
4.8 hrs (50%)	84.17 (91.56)	92.95 (93.37)
9.6 hrs (100%)	90.01 (92.88)	93.58 (93.64)

Table 4. Entity recognition (F1 scores) on the ATIS corpus. Results with additional speed-tempo data augmentation in parenthesis. For each experiment we use all the training transcripts annotated with entity labels as text-only data.

In our next set of experiments (see Table 4) we measure the effectiveness of our approach on entity recognition using the ATIS corpus. Similar to previous intent recognition experiments, in this case as well, the SLU model is able to learn on text-only data and a mix of text and speech data. An SLU model trained on text-only data with SLU labels achieves an F1 score of 85.95%. This is 95% of full performance with speech data at 90.01 F1 score. Adding 10% of speech data improves this to 91.07 F1 score, which even outperforms a model trained on all the speech data. Given this result, we add speed and tempo data augmentation, resulting in 4 additional training data replicas. While the speech-only results improve significantly (see results in parenthesis in Table 4), we still clearly see the additional benefit from adding text-only data. As mentioned earlier, the multi-modal speech and text training allows the model to learn both SLU targets and also novel acoustic variabilities from the dataset. While the information from the text-only data is very useful for transferring SLU knowledge, it is also necessary for the model to be acoustically robust. For this, only a very small amount of speech data is necessary within our proposed training framework.

For all our experiments, the RNN-T models are trained in Pytorch on V100 GPUs for 20 epochs using an AdamW optimizer. Similar to the base ASR model training, the maximum learning rate is set to 2e-4 and a OneCycleLR policy with a linear warmup phase from 2e-5 to 2e-4 over the first 6 epochs followed by a linear annealing phase to 0 for the remaining 14 epochs is employed. We use an effective batch size of 128 utterances. Batches are constructed from feature sequences of similar lengths without regard to whether the features are mel spectrograms or textograms.

5. CONCLUSION

In this work we have proposed and demonstrated the efficacy of a novel method that alleviates the need for annotated speech training data to build SLU systems. Using a novel frame-level text representation we first pre-train an ASR model that can process both speech and text data. With text-only SLU data and very limited amounts of speech, these models are further adapted to various SLU tasks. These SLU models perform at comparable levels as similar systems built on fully annotated speech SLU datasets. With text only training, we achieve up to 90% of the performance possible with full speech training. With just an additional 10% of speech data, these models significantly improve further to 97% of full performance.

6. REFERENCES

[1] Dmitriy Serdyuk, Yongqiang Wang, Christian Fuegen, Anuj Kumar, Baiyang Liu, and Yoshua Bengio, "Towards end-to-end spoken language understanding," in *Proc. ICASSP*, 2018.

[2] Parisa Haghani, Arun Narayanan, Michiel Bacchiani, Galen Chuang, Neeraj Gaur, Pedro Moreno, Rohit Prabhavalkar, Zhongdi Qu, and Austin Waters, "From audio to semantics: Approaches to end-to-end spoken language understanding," in *Proc. SLT*, 2018.

[3] Yao Qian, Rutuja Ubale, Vikram Ramanaryanan, Patrick Lange, David Suendermann-Oeft, Keelan Evanini, and Eugene Tsuprun, "Exploring ASR-free end-to-end modeling to improve spoken language understanding in a cloud-based dialog system," in *Proc. ASRU*, 2017.

[4] Yuan-Ping Chen, Ryan Price, and Srinivas Bangalore, "Spoken language understanding without speech recognition," in *Proc. ICASSP*, 2018.

[5] Sahar Ghannay, Antoine Caubrière, Yannick Estève, Nathalie Camelin, Edwin Simonnet, Antoine Laurent, and Emmanuel Morin, "End-to-end named entity and semantic concept extraction from speech," in *Proc. IEEE SLT*, 2018.

[6] Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio, "Speech model pre-training for end-to-end spoken language understanding," in *Proc. Interspeech*, 2019.

[7] Antoine Caubrière, Natalia Tomashenko, Antoine Laurent, Emmanuel Morin, Nathalie Camelin, and Yannick Estève, "Curriculum-based transfer learning for an effective end-to-end spoken language understanding and domain portability," in *Proc. Interspeech*, 2019.

[8] Yinghui Huang, Hong-Kwang Kuo, Samuel Thomas, Zvi Kons, Kartik Audhkhasi, Brian Kingsbury, Ron Hoory, and Michael Picheny, "Leveraging unpaired text data for training end-to-end speech-to-intent systems," in *Proc. ICASSP*, 2020.

[9] Loren Lugosch, Brett H Meyer, Derek Nowrouzezahrai, and Mirco Ravanelli, "Using speech synthesis to train end-to-end spoken language understanding models," in *Proc. ICASSP*, 2020.

[10] Ryan Price, Mahnoosh Mehrabani, and Srinivas Bangalore, "Improved end-to-end spoken utterance classification with a self-attention acoustic classifier," in *Proc. ICASSP*, 2020.

[11] Martin Radfar, Athanasios Mouchtaris, and Siegfried Kunzmann, "End-to-end neural transformer based spoken language understanding," in *Proc. Interspeech*, 2020.

[12] Yusheng Tian and Philip John Gorinski, "Improving end-to-end speech-to-intent classification with Reptile," in *Proc. Interspeech*, 2020.

[13] Xueli Jia, Jianzong Wang, Zhiyong Zhang, Ning Cheng, and Jing Xiao, "Large-scale transfer learning for low-resource spoken language understanding," in *Proc. Interspeech*, 2020.

[14] Hong-Kwang J. Kuo, Zoltán Tüske, Samuel Thomas, Yinghui Huang, Kartik Audhkhasi, Brian Kingsbury, Gakuto Kurata, Zvi Kons, Ron Hoory, and Luis Lastras, "End-to-end spoken language understanding without full transcripts," in *Proc. Interspeech*, 2020.

[15] Elisavet Palogiannidi, Ioannis Gkinis, George Mastrapas, Petr Mizera, and Themos Stafylakis, "End-to-end architectures for ASR-free spoken language understanding," in *Proc. ICASSP*, 2020.

[16] Samuel Thomas, Hong-Kwang J Kuo, George Saon, Zoltán Tüske, Brian Kingsbury, Gakuto Kurata, Zvi Kons, and Ron Hoory, "Rnn transducer models for spoken language understanding," in *Proc. ICASSP*, 2021.

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

[18] Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al., "Streaming end-to-end speech recognition for mobile devices," in *Proc. ICASSP*, 2019.

[19] Kanishka Rao, Haşim Sak, and Rohit Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *Proc. ASRU*, 2017.

[20] Jinyu Li, Rui Zhao, Hu Hu, and Yifan Gong, "Improving RNN transducer modeling for end-to-end speech recognition," in *Proc. ASRU Workshop*, 2019.

[21] Laurent El Shafey, Hagen Soltau, and Izhak Shafran, "Joint speech recognition and speaker diarization via sequence transduction," in *Proc. Interspeech*, 2019.

[22] Mohammadreza Ghodsi, Xiaofeng Liu, James Apfel, Rodrigo Cabrera, and Eugene Weinstein, "RNN-transducer with stateless prediction network," in *Proc. ICASSP*, 2020.

[23] George Saon, Zoltán Tüske, Daniel Bolanos, and Brian Kingsbury, "Advancing RNN transducer technology for speech recognition," in *Proc. ICASSP*, 2021.

[24] George Saon, Zoltán Tüske, Kartik Audhkhasi, and Brian Kingsbury, "Sequence noise injected training for end-to-end speech recognition," in *Proc. ICASSP*, 2019.

[25] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[26] Mike Wu, Jonathan Nafziger, Anthony Scodary, and Andrew Maas, "Harpervalleybank: A domain-specific spoken dialog corpus," *arXiv preprint arXiv:2010.13929*, 2020.

[27] Vaibhava Goel, Hong-Kwang J. Kuo, Sabine Deligne, and Cheng Wu, "Language model estimation for optimizing end-to-end performance of a natural language call routing system," in *Proc. ICASSP*, 2005.

[28] Charles T Hemphill, John J Godfrey, and George R Doddington, "The ATIS spoken language systems pilot corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania*, 1990.