INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Leveraging Pretrained ASR Encoders for Efficient End-to-End Speech Intent Classification and Slot Filling

He Huang, Jagadeesh Balam, Boris Ginsburg

NVIDIA, USA

{heh,jbalam,bginsburg}@nvidia.com

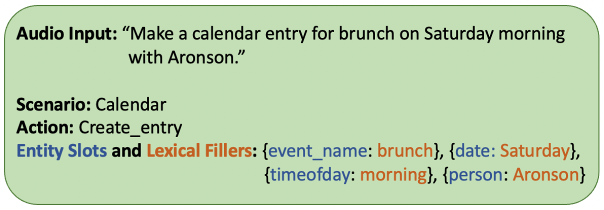
Abstract

We study speech intent classification and slot filling (SICSF) by proposing to use an encoder pretrained on speech recognition (ASR) to initialize an end-to-end (E2E) Conformer-Transformer model, which achieves the new state-of-the-art results on the SLURP dataset, with 90.14% intent accuracy and 82.27% SLURP-F1. We compare our model with encoders pretrained on self-supervised learning (SSL), and show that ASR pretraining is much more effective than SSL for SICSF. To explore parameter efficiency, we freeze the encoder and add Adapter modules, and show that parameter efficiency is only achievable with an ASR-pretrained encoder, while the SSL encoder needs full finetuning to achieve comparable results. In addition, we provide an in-depth comparison on end-to-end models versus cascading models (ASR+NLU), and show that E2E models are better than cascaded models unless an oracle ASR model is provided. Last but not least, our model is the first E2E model that achieves the same performance as cascading models with oracle ASR. Code, checkpoints and configs are available.[1]

1. Introduction

Spoken language understanding (SLU) is an essential component of conversational AI, which aims to extract semantic information directly from speech data. SLU has a very broad scope, including intent classification [1], slot filling [2, 3, 4, 5], speech emotion recognition [6, 7], question answering [8, 9], etc. There are mainly two types of SLU models: (1) cascading (ASR+NLU) models that first perform automatic speech recognition (ASR) and then apply a natural language understanding (NLU) model to the transcribed text; (2) end-to-end (E2E) models that directly predict the semantic output without predicting transcriptions. Compared with its counterpart natural language understanding (NLU), SLU is more challenging. On one hand, errors will propagate from ASR to NLU in cascading SLU models. On the other hand, end-to-end SLU models cannot make use of the pretrained large language models such as BERT [10]. To tackle these challenges, we use end-to-end SLU models that do not have error propagation, and we also show that better performance can be achieved by utilizing ASR encoders pretrained on out-of-domain datasets.

In this paper, we study the *speech intent classification and slot filling* (SICSF) task, which aims to detect user intents and extract the corresponding lexical fillers for detected entity slots at the same time, as illustrated in Figure 1. The most common end-to-end approach in this task is the encoder-decoder framework, where the encoder is responsible for extracting acoustic



Audio Input: "Make a calendar entry for brunch on Saturday morning with Aronson."

Scenario: Calendar
Action: Create_entry
Entity Slots and Lexical Fillers: {event_name: brunch}, {date: Saturday}, {timeofday: morning}, {person: Aronson}

Figure 1: *An example of speech intent classification and slot filling. An intent is composed of a* scenario *and an* action*, while* slots *and* fillers *are represented by* key-value *pairs.*

features from input audios, and the decoder is responsible for decoding the features to semantic output of intents and slots. Current works [3, 5] use encoders pretrained by self-supervised learning (SSL) [11, 12]. However, there is a large domain gap between the self-supervised learning task and the SICSF task, which limits the benefits that SLU models can obtain from the SSL-pretrained encoders. Some recent works [4, 5] also propose to train the SLU model with additional ASR task in a multi-task loss, which is more tricky to train since it's hard to choose a proper weight to balance different loss terms. Also, such multi-task approach wastes some network parameters in learning the ASR decoder which is not used during inference phase of the SLU task.

We tackle the SICSF problem with an end-to-end approach, by using a Conformer-Transformer framework that casts the task as a sequence-to-sequence problem. Based on the intuition that *the SICSF task can be treated as an audio-to-text problem*, we propose to use an encoder pretrained by automatic speech recognition (ASR), which is also an audio-to-text task. The SSL objective, however, focuses on distinguishing one feature from the other features in the same sequence, which is very different from the SICSF task. We propose that, since the ASR objective is closer to the SICSF objective than the SSL objective, ASR-pretrained encoders are more beneficial to the SICSF task than SSL-pretrained encoders. Our main contributions are summarized as follows:

- **Effectiveness**: We present a Conformer-Transformer model with ASR-pretrained encoder that achieves new state-of-the-art performance on the SLURP dataset [2], outperforming the other end-to-end (E2E) baselines by a large margin. This validates our hypothesis that ASR-pretrained encoders are more suitable for this task than SSL-pretrained encoders because of the task similarity between ASR and SICSF.
- **Efficiency**: We conduct extensive experiments on exploring parameter efficiency of E2E models, including freezing the encoder and using Adapters [13] in the encoder and finetun-

[1]https://github.com/NVIDIA/NeMo/tree/main/examples/slu/speech_intent_slot

10.21437/Interspeech.2023-461

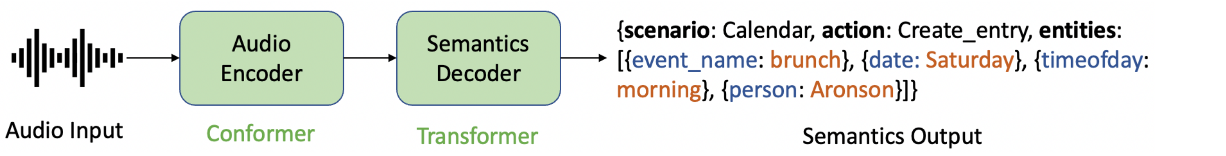

Audio Input
Audio Encoder
Conformer
Semantics Decoder
Transformer
{scenario: Calendar, action: Create_entry, entities: [{event_name: brunch}, {date: Saturday}, {timeofday: morning}, {person: Aronson}]}
Semantics Output

Figure 2: *The proposed model for* speech intent classification and slot filling *(SICSF). Here the semantics output, originally expressed in Python dictionary format, is flattened as a text string.*

ing on SLURP speech recognition data. Our results show that the best parameter efficiency is only achievable when using ASR-pretrained encoders, while models with SSL-pretrained encoders need finetuning all parameters to work well.

- **E2E vs. Cascading**: We also compare the proposed end-to-end (E2E) model with cascading models (ASR+NLU), and show that cascading model can only outperform E2E models when using an oracle ASR model. Also, our E2E model with ASR-pretrained encoder is able to match the performance of cascading model with oracle ASR, while all previous E2E models fall behind.
- Our code and model checkpoints are open-sourced for use in future research.

2. The Proposed End-To-End Approach

2.1. Model

Our model is based on the encoder-decoder framework. We choose Conformer [14] as the encoder since it is one of most popular ASR architectures used in production by many companies (Microsoft [15], Google [14]). Based on the superior performance of Conformer ASR models, we hypothesize that the Conformer encoder can also be applied to a broader set of audio-to-text tasks such as speech intent classification and slot filling. However, unlike the ASR task that requires monotonic input-output alignment, the speech intent classification and slot filling task is not affected by the orders of predicted entities and thus doesn't require such monotonic property. Therefore, different from the monotonic CTC [16] and RNNT [17] decoders that ASR models usually use, we choose Transformer [18] as the decoder (see Figure 2), since it has the best global context by letting features of each timestamp to attend to that of all other timestamps.

The intent and slot semantics is formatted as a Python dictionary, which is further flattened as a Python string object. The semantics string is tokenized by SentencePiece [19], while begin-of-sentence (BOS), end-of-sentence (EOS) and padding (PAD) tokens are also added to obtain the target token sequence. In this case, the SICSF task is formulated as an audio-to-text problem similar to speech recognition (ASR).

2.2. Training

We initialize the encoder with a Conformer-CTC-large model pretrained on NeMo ASR-Set [2], while the 3-layer Transformer decoder is randomly initialized. To circumvent the input-output alignment problem, we follow the seq2seq framework [20] and optimize the model with negative log-likelihood (NLL) loss and teacher-forcing. Specifically, given the encoder output hidden states as $H = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$, and the target sequence labels $Y = [y_1, y_2, ..., y_L]$, the training objective is defined as:

$$\mathcal{L} = -\sum_{i=1}^{L-1} \log P(y_{i+1}|y_i, y_{i-1}, ..., y_1, H). \qquad (1)$$

2.3. Inference

During inference, the input to the decoder is a BOS token and the encoded audio features, and beam search with width 32 and temperature 1.25 is used to obtain the predicted semantics strings, which are then converted to Python dictionaries for evaluation. The strings that have syntax errors during conversion will be treated as empty dictionaries. Missing or invalid values are replaced with "None" during conversion.

3. Experiments

3.1. Dataset and Settings

We use the popular SLURP [2] dataset, which contains around 84 hours of train, 6.9 hours of dev and 10.2 hours of test audios. We use intent classification accuracy and the SLURP metrics proposed in [2] for evaluation.

3.2. Implementation Details

Our model is implemented with PyTorch and NeMo[3]. We set the tokenizer vocabulary size to 58, and each token is embedded as a 512-dimension feature. We use Adam optimizer, with momentum $[0.9, 0.98]$ and no weight decay. The learning rates for encoder and decoder are 2e-4 and 3e-4 respectively. A Cosine annealing scheduler with 2000 linear warm-up steps is applied. The proposed model and its variants are trained on an NVIDIA RTX A6000 GPU with batch size 32 for 100 epochs.

For the NLU model in cascading baselines, we use a vocabulary size of 1024 for input text from ASR, while the output vocabulary size is 512. The ASR model in cascading baseline is initialized as the same in our E2E SLU model. Before finetuning on SLURP ASR, the ASR model has word-error-rate (WER) of 23.5 on SLURP test set. The WER decreased to 9.5 after finetuning on SLURP ASR data.

3.3. Comparison with Baselines

We compare the proposed model with three E2E baselines: SpeechBrain-HuBERT [3], ESPnet-Conformer [4] and Wav2Vec-CTI-RoBERTa [5]. We also compare with two cascading (ASR+NLU) baselines, where the ASR model is a Conformer-CTC-large mode finetuned on SLURP, and the NLU model is a Transformer with 3 encoder and 3 decoder layers. Another cascading baseline with oracle ASR is also included.

We show the main results in Table 1. As we can see, the cascading baseline with oracle ASR works better than the other

[2]https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_conformer_ctc_large

[3]https://github.com/NVIDIA/NeMo

Table 1: *Comparison with cascading and end-to-end baselines on SLURP [2]. LL60k refers to the LibriLight dataset [21], while LS960 refers to the LibriSpeech dataset [22]. "Trainable Params" refers to the number of parameters that are optimized during training on SLURP dataset.*

Model	Trainable Params (M)	SSL Pretrained	ASR Pretrained	Intent Accuracy	SLURP Metrics Precsion	Recall	F1
Cascading (ASR+NLU)							
Oracle ASR + Transformer	4	None	None	**90.40**	83.09	80.19	81.61
Conformer-CTC + Transformer	127	None	NeMo ASR-Set, SLURP ASR	87.18	75.83	72.72	74.24
Conformer-CTC + Transformer	4	None	NeMo ASR-Set	71.10	64.49	58.27	61.22
End-to-End							
SpeechBrain-HuBERT-large [3]	317	LL-60k	None	89.37	80.54	77.44	78.96
SpeechBrain-HuBERT-base [3]	96	LS-960	None	87.7	77.65	74.78	76.19
ESPnet-Conformer [4]	110	N/A	N/A	86.30	N/A	N/A	71.40
Wav2vec-CTI-RoBERTa [5]	200	LS-960	LS-960, SLURP ASR	86.92	N/A	N/A	74.66
NeMo-Conformer-Transformer-large	127	None	NeMo ASR-Set	90.14	**84.31**	**80.33**	**82.27**

Table 2: *Study on parameter efficiency. All models use the same Conformer-Transformer architecture.*

Freeze Encoder	Use Adapter	Trainable Params (M)	Pretrained (Task: Dataset)	SLURP-F1
No	No	127	SSL: LL60kh	77.22
Yes	No	12	SSL: LL60kh	36.21
Yes	Yes	13	SSL: LL60kh	43.28
No	No	127	ASR: NeMo ASR-Set	**82.27**
Yes	No	12	ASR: NeMo ASR-Set	72.59
Yes	Yes	13	ASR: NeMo ASR-Set	77.54

Table 3: *Ablation study on effectiveness of pretraining. All models use the same Conformer-Transformer-large network.*

Pretrained (Task: Dataset)	Intent Accuracy	SLURP Metrics Precision	Recall	F1
ASR: NeMo ASR-Set	**90.14**	**84.31**	**80.33**	**82.27**
ASR: LS960h	92.17	81.15	77.33	79.19
SSL: LL60kh	89.40	77.90	76.65	77.22
None	72.56	53.59	53.92	53.76

SLU baselines, as is also observed in previous works [2, 5]. However, our E2E SLU model, with 82.27% SLURP-F1, is able to match the performance of cascading models with oracle ASR, and outperforms the other E2E SLU models by a noticeable margin. It can also be noted that our model with 127M parameters is able to achieve 2.3% higher SLURP-F1 than the second best SLU model with 317M parameters. Compared with Wav2vec-CTI-RoBERTa [5], our model does not require finetuning on SLURP ASR, which makes our model more efficient to train. Overall, the superior performance of our model suggests that using an encoder pretrained on large ASR dataset is much more beneficial than encoders pretrained by self-supervised learning in speech intent classification and slot filling (SICSF). We will discuss this finding with more details in Section 3.5.1.

The cascading baseline, when not finetuned with ASR on SLURP, has much lower performance than the other baselines. Meanwhile, when the cascading baseline is finetuned on SLURP ASR, it's able to reach better performance than ESPnet-Conformer [4] and comparable slot filling performance with Wav2vec-CTI-RoBERTa [5]. This shows that, cascading ASR+NLU models, although have the drawback of propagating errors from ASR to NLU, still have the potential of getting good performance.

3.4. Exploring Parameter Efficiency

Adapters [13] were proposed to improve models' parameter efficiency in transfer learning for NLP, by adding only about 1% of the whole network parameters to the frozen model while achieving performance comparable to full finetuning. Each Adapter consists of several multi-layer perceptrons with very small intermediate dimensions (e.g., 32) and residual connections (Figure 3-Right). Since Adapters are proven to be very effective in speech recognition as well [23, 24], here we also study whether Adapters can help the SICSF task. As shown in Figure 3-Left, Adapters are added to each Conformer layer in the encoder of our model, while the encoder is frozen during training. We apply Adapters to both ASR-pretrained and SSL-pretrained encoders, and show the results in Table 2.

As we can see, for the SSL-pretrained encoder, merely freezing it without adding Adapters leads to very low performance, which is 40% absolute decrease from training the full model. By adding about 1M parameters, Adapters are able to improve the performance of SSl-pretrained encoders from 36.21% to 43.28%, which is about 7% improvement. On the other hand, merely freezing the ASR-pretrained encoder has 72.59% SLURP-F1, which is almost double the performance of SSL-pretrained encoder. This shows that the ASR-pretrained encoder itself is already able to generate representative features for the SICSF task, even without any finetuning. Meanwhile, adding Adapters to the frozen ASR-pretrained encoder only improves the performance by around 5%, which is an even smaller improvement than it is in the case of SSL-pretrained encoder. However, we can also see that adding Adapters to frozen ASR-pretrained encoder is able to match the performance of training the full model with SSL-pretrained encoder, and also outperforms some baselines [4, 5] in Table 1. Although using Adapters in frozen ASR-pretrained encoder still cannot match the performance of finetuning the whole encoder, considering the size of the model, it still achieves a good balance between performance and network size. From these results, we can con-

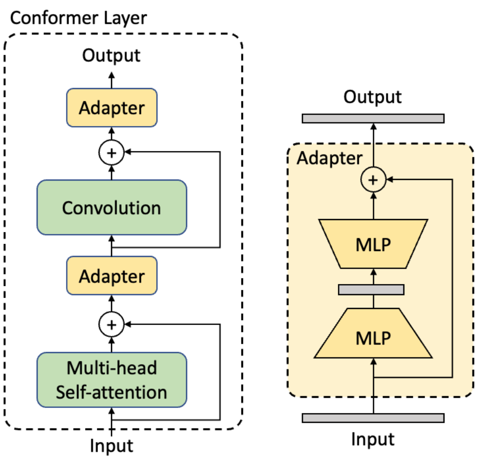


Figure 3: ***Left***: *Illustration of adding Adapter [13] modules to a Conformer [14] layer, where multi-layer perceptrons (MLPs) and layer-normalization are omitted for clarity.* ***Right***: *Illustration of an Adapter module, where non-linear activation is omitted for clarity.*

clude that having better parameter efficiency while maintaining good performance is only achievable for ASR-pretrained encoders, while model with SSL-pretrained encoder still needs a lot of parameters to work well, since the SICSF task is closer to audio-to-text in ASR rather than the frame discrimination task in SSL.

3.5. Ablation Study

3.5.1. Effect of Pretraining: ASR vs. SSL

We also try using SSL-pretrained encoder [4], and compare the results with ASR-pretrained encoders. We also add a baseline without any pretraining, and show the results in Table 3. As we can see, training from scratch has almost 30% decrease from the best model, while SSL-pretrained encoder with 77.22% SLURP-F1 lies in between. This shows that pretraining is still essential to the model's performance. We also try using ASR encoder pretrained on the LibriSpeech [22] dataset, which has about 2% higher SLURP-F1 than SSL-pretrained encoder. Considering the fact that LibriLight [21] has 60k hours audio, while LibriSpeech [22] only has 960 hours, we can see that ASR-pretrained encoder is more efficient in utilizing pretraining data than SSL-pretrained encoder. In addition, we try a Conformer-Xlarge encoder with 5x the size of Conformer-large, pretrained on LL60k [21], and find that it's only with this extra large model that using SSL-pretrained encoder can have a similar performance (SLURP-F1=81.28%) as using ASR-pretrained encoder (SLURP-F1=82.27%). In other words, using ASR-pretrained encoders is also more cost-efficient in terms of model size.

3.5.2. Effect of Finetuning on SLURP ASR

We further explore the effect of finetuning the ASR-pretrained encoder on SLURP ASR, and found that the performance does not change much. The intent accuracy and SLURP-F1 for the finetuned model are 90.28% and 82.08%, while with the ones without finetuning on SLURP ASR are 90.14% and 82.27%.

[4]`https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/ssl_en_conformer_large`

Table 4: *Ablation study on output vocabulary size. All models use the same Conformer-Transformer architecture.*

	Intent	**SLURP Metrics**		
Vocab	**Accuracy**	**Precision**	**Recall**	**F1**
58	90.14	**84.31**	**80.33**	**82.27**
256	90.12	83.09	79.47	81.24
512	90.13	83.19	79.77	81.44
1024	**90.17**	82.31	79.45	80.86

The reason for the similar performance is that the encoder pretrained on large ASR datasets already has the knowledge of audio-to-text task, while finetuning on SLURP ASR only improves the model's knowledge on dataset statistics. However, since later training on the SICSF task can also help the model learn dataset statistics, the effect of finetuning on SLURP ASR is less obvious.

3.5.3. Effect of Vocabulary Size

As performance of Confomer-based ASR models are usually affected by vocabulary sizes, where CTC [16] decoder usually works better with smaller (e.g., 128) vocabulary while RNNT [17] decoder is better with larger ones (e.g., 1024), here We study the effect of different vocabulary size on the proposed model, and show the results in Table 4. The intent accuracy remains pretty stable with different vocabulary sizes, while the best SLURP-F1 is obtained with the smallest vocabulary size. This is different from what we have observed for cascading models, where their performance grows as the output vocabulary size grows, and saturates around size 512. It should also be noted that cascading models perform badly with small input vocabulary size (e.g., 58), which is because the input has more diverse natural language while the output semantics has a more limited set of words.

4. Conclusion

We present a Conformer-Transformer model for end-to-end speech intent classification and slot filling (SICSF), where the Conformer encoder is pretrained on a large collection of speech recognition (ASR) datasets. Our model is able to achieve new stat-of-the-art results on the SLURP dataset. We also compare with cascading models, and show that our model can match the performance of cascading model with oracle ASR, while previous end-to-end models fall behind. We also study the effect of encoders pretrained by self-supervised learning (SSL), and show that ASR-pretrained encoder achieves noticeably better performance, since the SICSF objective is more similar to ASR than SSL. We also explore the parameter efficiency of our model, and show that using adapters in frozen ASR-pretrained encoder can still achieve very good performance, while SSL-pretrained encoder needs full finetuning to work well. Our code and checkpoints are publicly available to benefit future research.

5. References

[1] B. Sharma, M. Madhavi, and H. Li, "Leveraging acoustic and linguistic embeddings from pretrained speech and language models for intent classification," in *ICASSP*. IEEE, 2021.

[2] E. Bastianelli, A. Vanzo, P. Swietojanski, and V. Rieser, "SLURP: A Spoken Language Understanding Resource Package," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[3] Y. Wang, A. Boumadane, and A. Heba, "A fine-tuned wav2vec 2.0/hubert benchmark for speech emotion recognition, speaker verification and spoken language understanding," *arXiv:2111.02735*, 2021.

[4] S. Arora, S. Dalmia, P. Denisov, X. Chang, Y. Ueda, Y. Peng, Y. Zhang, S. Kumar, K. Ganesan, B. Yan *et al.*, "Espnet-slu: Advancing spoken language understanding through espnet," in *ICASSP*. IEEE, 2022.

[5] S. Seo, D. Kwak, and B. Lee, "Integration of pre-trained networks with continuous token interface for end-to-end spoken language understanding," in *ICASSP*, 2022.

[6] E. Chen, Z. Lu, H. Xu, L. Cao, Y. Zhang, and J. Fan, "A large scale speech sentiment corpus," in *Proceedings of the 12th Language Resources and Evaluation Conference*, 2020, pp. 6549–6555.

[7] S. Shon, P. Brusco, J. Pan, K. J. Han, and S. Watanabe, "Leveraging pre-trained language model for speech sentiment analysis," *Interspeech*, 2021.

[8] C.-H. Lee, S.-M. Wang, H.-C. Chang, and H.-Y. Lee, "Odsqa: Open-domain spoken question answering dataset," in *2018 IEEE Spoken Language Technology Workshop (SLT)*, 2018.

[9] C. You, N. Chen, and Y. Zou, "Knowledge distillation for improved accuracy in spoken question answering," in *ICASSP*, 2021.

[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, 2018.

[11] 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.

[12] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[13] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *ICML*, 2019.

[14] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *Interspeech*, 2020.

[15] S. Chen, Y. Wu, Z. Chen, J. Wu, J. Li, T. Yoshioka, C. Wang, S. Liu, and M. Zhou, "Continuous speech separation with conformer," in *ICASSP*, 2021.

[16] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

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

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[19] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *Conference on Empirical Methods in Natural Language Processing*, 2018.

[20] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[21] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP*, 2020.

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

[23] B. Thomas, S. Kessler, and S. Karout, "Efficient adapter transfer of self-supervised speech models for automatic speech recognition," in *ICASSP*, 2022.

[24] K. M. Sathyendra, T. Muniyappa, F.-J. Chang, J. Liu, J. Su, G. P. Strimel, A. Mouchtaris, and S. Kunzmann, "Contextual adapters for personalized speech recognition in neural transducers," in *ICASSP*, 2022.