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

GigaSpeech: An Evolving, Multi-domain ASR Corpus with 10,000 Hours of Transcribed Audio

Guoguo Chen[1,2*], *Shuzhou Chai*[1,3*], *Guanbo Wang*[1,3,4*], *Jiayu Du*[1*], *Wei-Qiang Zhang*[1,3*], *Chao Weng*[5†], *Dan Su*[5†], *Daniel Povey*[6†], *Jan Trmal*[4†], *Junbo Zhang*[6†], *Mingjie Jin*[5†], *Sanjeev Khudanpur*[4†], *Shinji Watanabe*[4,7†], *Shuaijiang Zhao*[8†], *Wei Zou*[8†], *Xiangang Li*[8†], *Xuchen Yao*[2†], *Yongqing Wang*[6†], *Zhao You*[5†], *Zhiyong Yan*[6†]

[1]SpeechColab, China
[2]Seasalt AI Inc, USA
[3]Department of Electronic Engineering, Tsinghua University, China
[4]CLSP & HLTCOE, The Johns Hopkins University, USA
[5]Tencent AI Lab, China
[6]Xiaomi Corporation, China
[7]Carnegie Mellon University, USA
[8]KE Holdings Inc, China
gigaspeech@speechcolab.org

Abstract

This paper introduces GigaSpeech, an evolving, multi-domain English speech recognition corpus with 10,000 hours of high quality labeled audio suitable for supervised training, and 33,000 hours of total audio suitable for semi-supervised and unsupervised training. Around 33,000 hours of transcribed audio is first collected from audiobooks, podcasts and YouTube, covering both read and spontaneous speaking styles, and a variety of topics, such as arts, science, sports, etc. A new forced alignment and segmentation pipeline is proposed to create sentence segments suitable for speech recognition training, and to filter out segments with low-quality transcription. For system training, GigaSpeech provides five subsets of different sizes, 10h, 250h, 1000h, 2500h, and 10000h. For our 10,000-hour *XL* training subset, we cap the word error rate at 4% during the filtering/validation stage, and for all our other smaller training subsets, we cap it at 0%. The *DEV* and *TEST* evaluation sets, on the other hand, are re-processed by professional human transcribers to ensure high transcription quality. Baseline systems are provided for popular speech recognition toolkits, namely Athena, ESPnet, Kaldi and Pika.

Index Terms: corpus, forced alignment, segmentation, speech recognition

1. Introduction

Thanks to the rapid development of the neural network models, automatic speech recognition (ASR) has made tremendous progress in the past decade. Various system architectures, from hybrid [1] to end-to-end [2], are proposed, and state-of-the-art results on standard benchmarks are being frequently updated.

The mainstream speech recognition corpora, on the other hand, have not changed much in decades. To take the English speech recognition task as an example, the Wall Street Journal corpus, which consists of 80 hours of narrated news articles [3], is almost 20 years old, and has a word error rate (WER) of 2.32% on its *eval92* benchmark [4]. The Switchboard and Fisher corpus, which consists of 262 and 1,698 hours of telephone conversational speech, is also around 20 years old, and has a WER of 5.5% on the Switchboard portion of the *Hub5'00* benchmark [5]. Even LibriSpeech [6], one of the most popular corpora for speech recognition tasks, is more than 5 years old, and has a WER of 1.9% on its *test_clean* benchmark [7]. It consists of 1,000 hours of read English speech. Due to the fast development of speech recognition techniques, ASR performance on those data sets appears to have saturated, making it difficult to track further improvements from new techniques.

There is some progress on creating better corpora/benchmarks for English speech recognition, from both academia and industry. TED-LIUM [8] is a series of corpora created by the Ubiqus company and the University of Le Mans. It consists of 452 hours of audio from TED talks in its latest release TED-LIUM 3. The corpora size, however, is less than 1,000 hours, making it not suitable for algorithms which demand a large amount of data. People's Speech [9] released by ML Commons consists of 87,000 hours of audio, covering 59 different languages. It's source, however, is mostly audiobook, lacking crucial acoustic diversity. Another work is SPGISpeech [10], a corpus released by Kensho Technologies. It consists of 5,000 hours of transcribed audio from earnings calls transcribed by S&P Global, Inc. The corpus by its nature gives an emphasis to the business domain.

We release a complementary English speech recognition corpus named GigaSpeech, an evolving, multi-domain ASR corpus with 10,000 hours of transcribed Audio. We make two contributions in this work. First, we release an evolving, multi-domain speech recognition corpus with 10,000 hours of labeled audio. Second we provide a scalable, reliable pipeline for generating speech recognition corpora.

2. GigaSpeech Corpus

This section explains the structure of the GigaSpeech corpus, including metadata, data partition, audio format, etc. Instructions and scripts for downloading GigaSpeech can be found on

[*] equal contribution
[†] in first-name order

http://dx.doi.org/10.21437/Interspeech.2021-1965

Table 1: *GigaSpeech training subsets*

Subset	Audiobook	Podcast	YouTube	Total
XL	2,655h	3,499h	3,846h	10,000h
L	650h	875h	975h	2,500h
M	260h	350h	390h	1,000h
S	65h	87.5h	97.5h	250h
XS	2.6h	3.5h	3.9h	10h

Table 2: *GigaSpeech evaluation sets*

Sets	Podcast	YouTube	Total
DEV	6.3h	6.2h	12.5h
TEST	16.1h	24.2h	40.3h

GigaSpeech's GitHub repository [11].

2.1. Metadata

We save all the metadata information to a single JSON file named GigaSpeech.json.

To use the corpus, users are expected to extract the relevant information from GigaSpeech.json. For example, for the speech recognition task, one should first follow the "audios" entry, and work out a list of audio files. One can then follow the "url" entry to download the original audio file, or "path" if preprocessed audio files have been downloaded to the disk. After that, for each audio file, one can follow the "segments" entry, as well as their corresponding transcripts. Of course, we also have various supplementary entries, such as "subsets", "md5", which will also be helpful for your task.

The metadata file GigaSpeech.json is version controlled, and is supposed to get updated over the time. In future releases, we plan to add speaker information so that it will be suitable for speaker identification/verification tasks. We also plan to add more data from different sources to increase the diversity.

2.2. Training Subsets

We provide 5 training subsets, namely *XS*, *S*, *M*, *L* and *XL*, listed here in order of increasing audio hours. Table 1 shows a detailed breakdown of the 5 GigaSpeech training subsets.

2.3. Evaluation Sets

We provide 2 evaluation sets: a *DEV* set for development and tuning, which consists of 12.5 hours of audio, and a *TEST* set for final evaluation, which consists of 40.3 hours.

A breakdown of our evaluation sets is illustrated in Table 2. Note that our evaluation sets do not have a coverage for the audiobooks. We make sure that audio files from the LibriSpeech [6] evaluation sets (*dev-clean*, *dev-other*, *test-clean* and *test-other*) are not presented in our corpus, therefore, the LibriSpeech evaluation sets can be used as our evaluation sets.

2.4. Audio Format

To reduce the file size of the GigaSpeech corpus, we compress the original audio using the Opus audio codec. Original audio files are first converted to 16kHz sampling rate, single channel and 16-bit signed-integer format. Opus compression is then applied to achieve an output bit rate of 32 kpbs, which results in a compression ratio of 8.

Table 3 shows the impact of Opus audio compression in terms of WER (%). Kaldi systems (see Section 4.3, but without recurrent neural network language model rescoring) are built for our *M* (1000h) training subset, with or without Opus compression. These two systems are then used to decode the *DEV* and *TEST* evaluation sets, with or without Opus compression. From Table 3, it is clear that compressing the training data with the Opus codec at 32 kpbs output bit rate has very small impact on the *DEV* and *TEST* set (0.1 - 0.2% WER degradation).

Table 3: *Impact of Opus audio compression (WER in %)*

Train	Eval	*DEV*		*TEST*	
		Opus	Wav	Opus	Wav
M	Opus	19.0	18.7	18.5	18.3
	Wav	18.8	18.5	18.3	18.2

3. GigaSpeech Creation Pipeline

This section presents the detailed pipeline for creating the GigaSpeech corpus, which can be applied to other data generation tasks as well.

3.1. Stage 1: Audio Collection

We start the task by manually defining the categories that we are interested in. We selected 24 categories in total, namely *Arts*, *Business*, *Education*, *Autos and Vehicles*, *Comedy*, *Crime*, *Entertainment*, *Film and Animation*, *Gaming*, *Health and Fitness*, *History*, *Howto and Style*, *Kids and Family*, *Leisure*, *Music*, *News and Politics*, *Nonprofits and Activism*, *People and Blogs*, *Pets and Animals*, *Religion and Spirituality*, *Science and Technology*, *Society and Culture*, *Sports*, *Travel and Events*.

For podcasts, we follow the above categories, and select episodes that come with manual transcriptions. For YouTube, we use the above categories as seed keywords, and select videos with human-generated closed captions. For audiobooks, we do not enforce those categories.

Once we have the list of audio files, we create tools and download all audio files with their corresponding transcripts.

3.2. Stage 2: Text Normalization

The audio transcripts we download from various sources are created by different transcribers with diversified transcription standards and styles, therefore it is necessary to apply text normalization to the original transcripts. We perform standard text normalization, including case normalization, special symbol removal, number to word rewriting, date/time rewriting, etc.

For audiobooks and podcasts, transcripts are usually at the episodes or chapter/book level. For speech recognition, however, smaller segments less than 20 seconds are needed for training. The next step is to segment the long audio file into smaller segments. For YouTube, closed captions are provided at the sentence level, but unfortunately we find that the timestamps of closed captions are not reliable for segmentation. As a result, we decide to splice the closed captions all together, and perform the same segmentation as audiobooks and podcasts.

3.3. Stage 3: Forced Alignment

Our aligner is implemented with Kaldi [12], and the alignment procedure follows the work here [13], which adopts a divide and conquer strategy to tackle the alignment problem. First, both audio and transcript are uniformly chunked into smaller pieces. Second, audio segments are decoded with a biased language model (LM), and hypotheses with timestamps are generated. Third, each hypothesis segment is matched to one transcript segment via TF-IDF similarity. For each matched pair, hypothesis is further aligned with the transcript segment us-

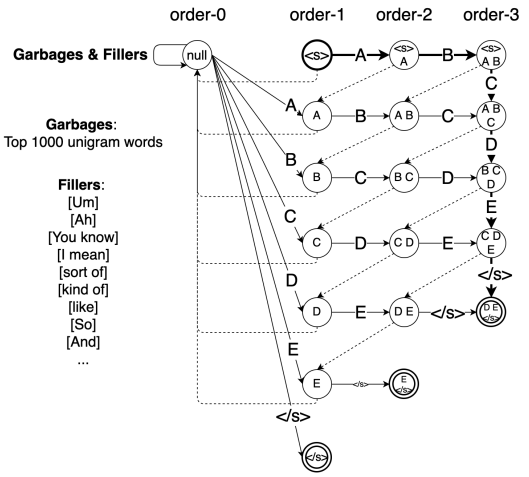

Figure 1: *A forced alignment graph for the sentence "<s> A B C D E </s>" (4-gram)*

ing the Smith-Waterman algorithm [14]. Finally, through this alignment, timestamps are attached to the transcript segments, and eventually to the whole transcripts by stitching the independently aligned segments together. Note that we modify the Smith-Waterman algorithm to handle silence and punctuation, and this is essential to enable the sentence-based segmentation in the next section.

To achieve better alignment performance, we first align and segment the downloaded audio with a close-domain acoustic model. We then train an in-domain acoustic model with the audio segments created (around 3,000 hours). This model is used to align the whole corpus.

3.4. Stage 4: Audio Segmentation

We work out the audio segments from the alignment information. Several rules are applied during the segmentation process:

- Split allowed at silence that is longer than 1 second.
- Split allowed at punctuation (",", ".", "!" or "?") that is longer than 0.2 seconds.
- Segments with alignment WER $\geq 75\%$ are removed.
- Segments with length ≥ 20 seconds are removed.
- Silence at segment boundaries are truncated to 0.15 seconds.

It is worth pointing out that we keep 4 types of punctuation, namely comma, period, question mark and exclamation mark, so that split can happen at sentence boundaries (second rule above). We map them to special words "<COMMA>", "<PERIOD>", "<QUESTIONMARK>" and "<EXCLAMATIONMARK>" respectively. Besides, this also allows us to build end-to-end speech recognition systems that includes punctuation tagging, and end point detection.

3.5. Stage 5: Segment Validation

The segmentation stage generates a list of candidate segments, but potentially with high transcription error rate. We therefore apply segment validation to filter out bad segments.

3.5.1. Forced Alignment Graph

To better detect transcription errors made by human transcribers, we propose a variation of alignment graph in n-gram framework, as shown in Figure 1. The bold arrow path represents a typical LM-free forced alignment graph. Each state on the forced alignment path has a dotted "leaky" arc (with weight) that allows the token to leak out the forced alignment path, from higher order n-gram states down to lower order n-gram states, until reaching the null state. A garbage word loop (containing top 1,000 uni-gram words) is added around the null state to consume additional acoustic frames. Besides, there are extra states and arcs that allow the token to return to the forced alignment path. In general, this alignment graph allows the decoder to perform insertion/deletion/substitution to the reference. This essentially brings more flexibility to the forced alignment stage, making it possible to capture the discrepancy between the audio and the corresponding transcript.

3.5.2. Validation Decoding Pass

During the validation decoding pass, we detect transcription errors and filter out segments with high error rate. Figure 2 gives two examples of how errors are being detected. In the first example, the transcriber misses a word "YOU" in the transcript, which is caught by our decoder. In the second example, the transcriber writes a typo which is also successfully detected.

For the podcast and YouTube portion of our *XL* training subset, we cap the maximum WER at 4%, and throw away all segments with higher WER. For the audiobook portion of the *XL* training subset, as well as all other smaller subsets, we cap the maximum WER at 0%, meaning we don't allow any transcription errors.

3.5.3. Reference Rewriting

Investigation into the validated segments reveals three common types of transcriber errors:

- Fillers ignored, such as AH, UH, UM, ER, ERR, YOU KNOW, I MEAN, SORT OF, etc.
- Conjunctions ignored, such as AND, OR, BUT, etc.
- Disfluency removed, such as "It's it's it's a great thing!".

Discarding these segments will fundamentally limit the diversity of the corpus. To fix those common errors, we add a filler loop (see Figure 1) to the forced alignment graph, which contains the above common fillers and conjunctions. Besides, we also employ a disfluency detector. The filler loop and the disfluency detector may modify the reference (reference rewriting), and if that happens, those words will be counted as correct. We only apply reference rewriting to our *XL* subset (10,000h).

Since our evaluation sets are manually processed by professional human transcribers, we take that as the ground truth, and use it to compute the frame level segmentation precision and recall. Here recall tells us how many frames can be retrieved by our segmentation and validation pipeline, and precision tells us how many of these retrieved frames are correctly labeled (consistent with the human labels).

Figure 3 illustrates the precision-recall curve on the podcast portion of the *DEV* evaluation set. For the *XL* training subset, we select a working point that gives us 10,000 hours of validated audio data, while keeping the maximum WER under 4%. And for all our other training subsets, we select the working point that keeps the maximum WER at 0% (the left most working point on Figure 3).

```
ref  : THANK   <eps>   FOR    THIS    FOOD    AND     THIS    DAY     AND     THAT    MY      CONCERT'S TODAY   <PERIOD>
hyp  : THANK   YOU#G   FOR    THIS    FOOD    AND     THIS    DAY     AND     THAT    MY      CONCERT'S TODAY   <eps>
edit : C       I       C      C       C       C       C       C       C       C       C       C         C       SIL

ref  : <eps>   THAT    WE     <eps>   UNDERSTAND <eps>   THAT    IT      <eps>   W       ILL     TAKE    THEM    LONGER  TO      RESPOND TO      US
hyp  : <eps>   THAT    WE     <eps>   UNDERSTAND <eps>   THAT    IT      <eps>   WILL#G  <eps>   TAKE    THEM    LONGER  TO      RESPOND TO      US
edit : SIL     C       C      SIL     C          SIL     C       C       SIL     S       D       C       C       C       C       C       C       C
```

Figure 2: *Examples of transcription errors detected by forced alignment*

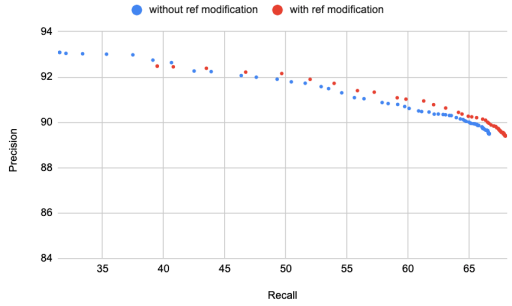

Figure 3: *Frame level precision-recall curve for the segmentation and validation pipeline*

4. Experiments

This section describes the baseline systems and experimental results for four popular speech recognition toolkits, namely Athena, ESPnet [15], Kaldi [12] and Pika [16].

4.1. Athena Baseline System [17]

The Athena baseline implements an encoder-decoder based transformer model, which is similar to [18], with parameters $e = 12$, $d = 6$, $d_{\text{model}} = 1024$, $d_{\text{ff}} = 2048$ and $d_{\text{head}} = 8$. During the training, the output sequence of the encoder is also used for connectionist temporal classification (CTC) for joint training to enforce monotonic alignment between speech and label sequences. The model is trained with a total batch size of 128 for 5 epochs. During the decoding, beam search with a beam size of 8 is used, which combines the scores of the decoder, the CTC with weight 0.5 and the LM with weight 0.2.

4.2. ESPnet Baseline System [19]

The ESPnet baseline uses conformer (Convolution-Augmented Transformer) [7], which is a recently proposed architecture combining the local sensitivities of convolutional neural networks with the long-range interactions of transformers [20]. We use the implementation provided in the ESPnet toolkit [21]. We use a set of 5k BPE tokens generated by the SentencePiece tokenizer [22]. The model has a 12 conformer blocks with an output dimension of 512 and a kernel size of 31 in the encoder, and 6 transformer blocks in the decoder. Both encoder and decoder have 8 attention heads with 2048 feed-forward unit dimension. SpecAug used 2 frequency masks and 5 time masks. The last 10 best checkpoints were averaged as the final model.

4.3. Kaldi Baseline System [23]

The Kaldi baseline implements a typical chain model. First, a GMM-HMM model is trained to obtain the alignments with no data cleaning. Second, volume and speed augmentation techniques are applied. I-vectors are then extracted and pasted to the basic acoustic features, as most Kaldi's recipes do. Finally, a neural network is trained with both cross-entropy and LF-MMI criteria. Our neural network stacks 6 convolutional neural network (CNN) layers, 10 TDNN-F layers, 1 attention-relu-renorm-layer, 1 TDNN-F layer, 1 fast-lstmp-layer, 1 TDNN-F layer and finally 1 more fast-lstmp-layer. During the decoding stage, a 4-gram LM is first used for decoding, followed by a Recurrent Neural Network Language Model (RNNLM) rescoring pass.

Table 4: *GigaSpeech baselines for the XL training subset (WER in %)*

Toolkit	Model	*DEV*	*TEST*
Athena	Transformer-AED + RNNLM	13.60	12.70
ESPnet	Conformer/Transformer-AED	10.90	10.80
Kaldi	Chain + RNNLM	14.78	14.84
Pika	RNN-T	12.30	12.30

Table 5: *Kaldi baselines for GigaSpeech training subsets (WER in %)*

Subset	*DEV*	*TEST*
XL	14.78	14.84
L	16.60	16.28
M	17.96	17.53
S	22.59	22.14
XS	48.83	48.17

4.4. Pika Baseline System [24]

The Pika baseline adopts a convolution and transformer based architecture [16] for the encoder of our RNN-T system. Five-layer transformer is used for the decoder and the hidden dimension of each layer is 512. The targets of our RNN-T system are a set of English wordpieces plus blank symbol which lead to an output dimension of 5000. Our distributed training strategy is based on block-wise model-update filtering (BMUF) with a Nesterov momentum scheme but with different learning rate scheduling [16] where both initial and final learning rates are set before training and number of training epochs are fixed (therefore there is no early stop and no development/validation set is used). For decoding, we set beam-size to 8 and the temperature of softmax to 1.25.

4.5. Experimental Results

Table 4 demonstrates baseline results for Athena, ESPnet, Kaldi and Pika. Results listed here are purely for the purpose of providing baseline systems for each toolkit. They do not reflect the state-of-the-art performance of each toolkit, and cannot be used to compare the performance across toolkits.

Table 5 illustrates the Kaldi baseline results for 5 training subsets. Generally speaking, as the training subset gets bigger, the performance goes up. Our smallest *XS* training subset (10h) is designed for system building and debugging only, and is not expected to give strong performance.

5. Acknowledgements

The authors would like to thank Xingyu Na and Yujun Wang for their various suggestions on the Kaldi baseline system. The authors also would like to thank Speechocean for transcribing the GigaSpeech evaluation sets.

6. References

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2011.

[2] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. International Conference on Machine Learning (ICML)*. PMLR, 2014, pp. 1764–1772.

[3] D. B. Paul and J. Baker, "The design for the wall street journal-based csr corpus," in *Proc. International Conference on Spoken Language Processing (ICSLP)*. ISCA, 1992.

[4] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely Sequence-Trained Neural Networks for ASR Based on Lattice-Free MMI," in *Proc. Interspeech 2016*. ISCA, 2016, pp. 2751–2755.

[5] G. Saon, G. Kurata, T. Sercu, K. Audhkhasi, S. Thomas, D. Dimitriadis, X. Cui, B. Ramabhadran, M. Picheny, L.-L. Lim *et al.*, "English conversational telephone speech recognition by humans and machines," *arXiv preprint arXiv:1703.02136*, 2017.

[6] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Proc. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[7] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. Interspeech 2020*, 2020, pp. 5036–5040.

[8] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *Proc. International Conference on Speech and Computer*. Springer, 2018, pp. 198–208.

[9] "People's Speech," https://mlcommons.org/en/peoples-speech/, accessed April 1, 2021.

[10] P. K. O'Neill, V. Lavrukhin, S. Majumdar, V. Noroozi, Y. Zhang, O. Kuchaiev, J. Balam, Y. Dovzhenko, K. Freyberg, M. D. Shulman, B. Ginsburg, S. Watanabe, and G. Kucsko, "SPGISpeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition," in *Submitted to Interspeech*, 2021.

[11] https://github.com/SpeechColab/GigaSpeech.

[12] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *Proc. 2011 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2011.

[13] V. Manohar, D. Povey, and S. Khudanpur, "JHU Kaldi system for Arabic MGB-3 ASR challenge using diarization, audio-transcript alignment and transfer learning," in *Proc. 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 346–352.

[14] W. R. Pearson, "Searching protein sequence libraries: Comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms," *Genomics*, vol. 11, no. 3, pp. 635–650, 1991.

[15] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, and N. Chen, "ESPnet: End-to-end speech processing toolkit," in *Proc. Interspeech 2018*. ISCA, 2018.

[16] C. Weng, C. Yu, J. Cui, C. Zhang, and D. Yu, "Minimum bayes risk training of RNN-transducer for end-to-end speech recognition," in *Proc. Interspeech 2020*. ISCA, 2020, pp. 966–970.

[17] https://github.com/athena-team/athena/tree/master/examples/asr/gigaspeech.

[18] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang *et al.*, "A comparative study on transformer vs rnn in speech applications," in *Proc. 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 449–456.

[19] https://github.com/espnet/espnet/tree/master/egs2/gigaspeech/asr1.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017.

[21] P. Guo, F. Boyer, X. Chang, T. Hayashi, Y. Higuchi, H. Inaguma, N. Kamo, C. Li, D. Garcia-Romero, J. Shi *et al.*, "Recent developments on espnet toolkit boosted by conformer," *arXiv preprint arXiv:2010.13956*, 2020.

[22] T. Kudo and J. Richardson, "Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," *arXiv preprint arXiv:1808.06226*, 2018.

[23] https://github.com/kaldi-asr/kaldi/tree/master/egs/gigaspeech.

[24] https://github.com/tencent-ailab/pika/tree/main/egs/gigaspeech.