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

Phonetically Induced Subwords for End-to-End Speech Recognition

Vasileios Papadourakis, Markus Müller, Jing Liu, Athanasios Mouchtaris and Maurizio Omologo

Alexa Machine Learning, Amazon, USA

{papadour, mumarkus, jlmk, mouchta, omologo}@amazon.com

Abstract

End-to-end automatic speech recognition systems map a sequence of acoustic features to text. In modern systems, text is encoded to grapheme subwords which are generated by methods designed for text processing tasks and therefore don't model or take advantage of the statistics of the acoustic features. Here, we present a novel method for generating grapheme subwords that are derived from phoneme sequences, therefore capturing phonetical statistics. The phonetically induced subwords can be used for training and inference in any system that benefits from subwords, regardless of architecture and without the need of a pronunciation lexicon. We compare our method to other commonly used methods, which are based on text statistics or on text-phoneme correspondence and present experiments on CTC and RNN-T architectures, evaluating subword sets of different sizes. We find that our phonetically induced subwords can improve performance of RNN-T models with relative improvements of up to 15.21% compared to other subword methods.

Index Terms: word piece, RNN-T, CTC, sequence-to-sequence, phoneme

1. Introduction

The goal of an end-to-end Automatic Speech Recognition (ASR) is to directly output a grapheme sequence (text) given the input audio. End-to-end systems allow training without a pronunciation lexicon while achieving similar performance to systems using phonemes as acoustic modeling units. A critical aspect of an end-to-end system is the segmentation of text to tokens which can be done at different levels of granularity: single characters, full words or the intermediate option of subwords. Early systems [1, 2, 3] used the Connectionist Temporal Classification (CTC) loss to train recurrent neural networks that output single characters. However, a drawback of single characters is that the system must also learn proper spelling, therefore an additional language model is needed [4, 5]. On the other end, systems that use full words also perform well, given enough training data [6, 7]. The drawback of full words is that the system can only recognize words included in its finite training vocabulary, therefore failing to recognize any out-of-vocabulary words.

The out-of-vocabulary problem is tackled with the use of subwords, originally introduced in the area of machine translation [8]. Multiple approaches have been used for subword generation: Byte-Pair Encoding (BPE) [9] is based on a compression algorithm which replaces frequent bigrams with new tokens, thereby compressing the input. BPE variants include using bytes instead of characters as input [10] or estimating unigram probabilities for the BPE subwords [11]. The SentencePiece framework [12] introduces a unigram language model tokenization method that can also take advantage of subword regularization [13]. In the context of end-to-end ASR, subwords are considered the norm as they outperform character-based segmentation across a variety of systems: CTC [14], Recurrent Neural Network Transducer (RNN-T) [15] or attention based [16, 17, 18]. Additionally, methods that use probabilistic decomposition such as subword regularization [19, 20] or latent sequence decomposition [21, 22] can further improve accuracy.

Table 1: *Grapheme and X-SAMPA phoneme encoding of the word SPEECH along with two example segmentations to subwords. Segmentation 1 follows the phoneme segmentation.*

Encoding type	**Encoded text**
graphemes	S | P | E | E | C | H
phonemes	`s | p | i | tS`
example segmentation 1	S | P | EE | CH
example segmentation 2	S | PE | EC | H

Given that end-to-end ASR aims to match acoustic features to text, the use of subwords based only on word spelling might be problematic, especially in cases where there is no consistent link between pronunciation and spelling. First, a text-based subword segmentation will not necessarily lead to a meaningful phonetic segmentation (Table 1). Second, a text-based subword will not necessarily correspond to a well-defined phonetic unit. The tetragraph *ough* offers a characteristic example of this issue as its pronunciation is notoriously unpredictable (Figure 1, [23]).

We present a novel method for generating Phonetically Induced Subwords (PhIS) that are based on pronunciation statistics but are still able to seamlessly tokenize text without the need of a lexicon. We conduct experiments on the 960h Librispeech dataset [24], on two different architectures (CTC and RNN-T) and on subword sets of different sizes. Our method outperforms the other subword generation methods in the RNN-T experiments, especially when larger vocabularies are used, with relative improvements in the range of 1.61% - 15.21%.

2. Related Work

To the best of our knowledge, few studies leverage phonetic information to generate subwords for end-to-end neural systems. One option is to create and decode phoneme subwords that are generated by applying a subword algorithm like BPE on phoneme sequences such as in [25]. Similarly, in [26], the authors showed that such phoneme subwords can be competitive to grapheme subwords on an encoder-decoder attention model. The downside of such approaches is the requirement of a pronunciation lexicon during model training and inference. Additionally, the challenges of correct mapping of phoneme sequences to words must be tackled with an additional external language model (as in [25]) or with the non-flexible inclusion of extra labels that deal with homophones (as

http://dx.doi.org/10.21437/Interspeech.2021-1787

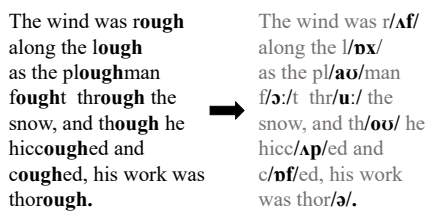

Figure 1: *Multiple pronunciations of one grapheme subword. In the right, tetragraph "ough" is transcribed to different IPA notation depending on word context.*

in [26]).

Closer to our approach is the Pronunciation Assisted Subword Modelling (PASM) that was shown to outperform BPE and single character baselines [27]. Subword generation in PASM is based on consistent alignments between single phonemes and single characters. A downside of this approach is that it tends to choose short subwords and avoids modelling full words with single tokens. As a consequence, subword variability is limited and, along with the method's exclusion criteria, the resulting vocabularies are relatively small (around 100 and 200 subwords for WSJ and Librispeech respectively). We compare our results to PASM in our 200 subword experiments.

3. Method

Our subword creation method is illustrated in Figure 2. The goal is to generate a $grapheme$ subword vocabulary that retains the properties of a $phoneme$ subword vocabulary and can be used in a probabilistic tokenization framework. We first describe the tokenization framework to provide intuition on how a given subword vocabulary is utilized to tokenize text.

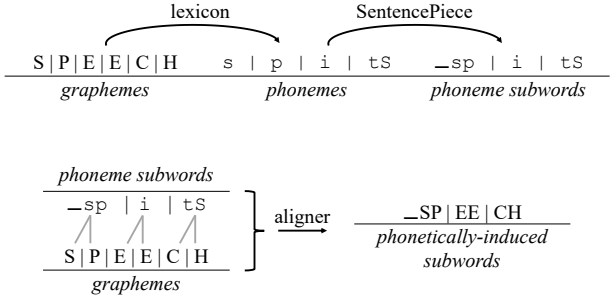

Figure 2: *Illustration of the PhIS vocabulary construction. Top: construction of phoneme subwords. Bottom: generation of PhIS vocabulary.*

3.1. Tokenization frameword

Given a subword vocabulary $\mathcal{V}$, let $\mathbf{x} = (x_1, ..., x_M)$ be a sequence of subwords that defines a segmentation of a character sequence $\mathbf{X}$, where M denotes the number of labels in the sequence. Each subword x_i is equipped with an occurrence probability $p(x_i)$. Following the unigram language model assumption that each subword occurs independently, the probability of a segmentation is given by

$$P(\mathbf{x}) = \prod_{i=1}^{M} p(x_i), \tag{1}$$

with $\sum_{x_i \in \mathcal{V}} p(x_i) = 1$. An input character sequence $\mathbf{X}$ can be segmented to multiple segmentation candidates $\mathcal{S}(\mathbf{X})$ with the most probable segmentation given by

$$\mathbf{x}^* = \underset{\mathbf{x} \in \mathcal{S}(\mathbf{X})}{\operatorname{argmax}} P(\mathbf{x}). \tag{2}$$

In this framework, text tokenization is defined by the vocabulary $\mathcal{V}$ and the occurrence probabilities $p(x_i)$ (also referred as subword weights). A third important parameter is the vocabulary size, as it implicitly affects the generated subword set through the vocabulary construction process. For example, in BPE, new subwords are added by merging frequent bigrams until the desired vocabulary size is reached.

3.2. Construction of phoneme subword vocabulary

Assuming that phoneme subwords capture important regularities in phoneme space, similar to the way text subwords capture text statistics, we first construct a vocabulary of phoneme subwords. Text data are translated to phonemes in X-SAMPA notation with a human transcribed pronunciation lexicon which is modified so that each phoneme is a single ASCII character to ensure proper phoneme subword generation. Phoneme subwords are created by simply using the phoneme sequences as input to the unigram language model vocabulary construction algorithm of [13]. Briefly, the algorithm begins with a reasonably large seed vocabulary that is iteratively pruned to the desired size. At each iteration, subword weights $p(x_i)$ are estimated with the EM algorithm and subwords are removed based on their contribution to the overall segmentation likelihood. The vocabulary size $|\mathcal{V}|$, which will also be the size of the final PhIS vocabulary is defined at this step.

The phoneme subwords, along with the corresponding weights, define the desired phonetical segmentation of our corpus. Importantly, phoneme subwords can model full words with single tokens as needed, depending on the word frequency distribution.

3.3. Generation of PhIS vocabulary

Our end goal is a grapheme subword vocabulary which can be used seamlessly by any end-to-end ASR system without the need of a lexicon during training or inference and without the need of additional language models to deal with incorrect spelling. To achieve this, we match each phoneme subword to a grapheme sequence with `fast align` [28]. Originally used for word alignment in statistical machine translation, `fast align` operates on a source sequence of length n and a target sequence of length m. It generates an alignment $\mathbf{a} = (a_1, a_2, \ldots a_m)$ that matches a source (or null) token $i \in 1, ..., n$ to each target token $j \in 1, ..., m$. We used phoneme subwords as the source sequence and single letters as the target sequence. As we expect pronunciation and text to be correlated, we used the symmetrical alignment which combines source-target and target-source alignments. Because phoneme subwords are by definition larger units than single letters ($n \leq m$), a phoneme subword was almost always aligned to one or more letters. Connected regions of aligned letters provided the grapheme subword candidates for each phoneme subword. For

example, in the alignment shown in Figure 2, each phoneme subword is aligned to two graphemes.

3.4. PhIS vocabulary size and subword weights

As a phoneme subword can naturally belong to multiple words, the aligner returned multiple grapheme subword candidates for each phoneme subword. The candidates were ranked based on occurrence frequency and the 1-best were added to the PhIS vocabulary. Duplicates were replaced with the most frequent $2^{nd}-$ or $3^{rd}-$best candidates, pooled over all candidates. On average, 15% of the PhIS subwords originated from this list of n-best candidates. Importantly, each PhiS subword inherited the occurrence probability of the phoneme subwords it originated from. By using the phoneme subword weights in equations 1 and 2, we implicitly bias the text segmentation towards the natural phonetic segmentation. Weights were normalized post generation to retain the probability property.

3.5. End-to-end model training and architectures

All experiments were performed with the same hyperparameters: in the acoustic front-end, we computed 64-dimensional log Mel filterbank energy features, extracted with 25ms windows spaced at 10ms intervals. The features were normalized using global mean-variance statistics, stacked with 2 frames to the left and downsampled to a frame rate of 30ms. For training, we used the Adam optimizer [29] with learning rate scheduling which included a short warm up phase, a constant hold phase and an exponential decay phase. For decoding, we used beam search with a beam width of 16.

The CTC model was a simple multilayer LSTM [30] trained with the CTC loss [1]. The RNN-T [31, 32] consisted of an encoder, a decoder and a joint network. Both encoder and decoder were multilayer LSTM networks. In the RNN-T, the encoder converts the acoustic feature vectors into a hidden representation, while the decoder adds feedback over the output labels. The joint network combines the output of both encoder and decoder to generate label predictions.

4. Experiments

We compare the phonetically induced subwords (PhIS) against four different sets of subwords: single characters (char), BPE, SentencePiece Unigram language model (Unigram) and PASM and report absolute and relative Word Error Rates (WER). Because our method is closer to PASM and because PASM vocabulary size is restricted, we first present results on a small vocabulary of 200 subwords. In these experiments, relative WER is reported against the single character model. For experiments with larger vocabularies (2500 and 4096 subwords) we report relative WER against the PhIS subwords because our method achieved the best results in both experiments.

We used the dev set to verify convergence and report results on the test-clean and test-other Librispeech evaluation sets. Though the reported WERs are not as low as some state-of-the-art results obtained in recent benchmarking works on LibriSpeech [18, 26], it is worth noting that in our case we did not use any pretraining, second-pass rescoring or external language models, and did not fine-tune any parameter for any of the subwords sets to avoid biasing the experimental study. For reference purposes, the most comparable neural model to our baseline RNN-T (in terms of model architecture and size) was described in [33], which reported 12.31% and 23.16% WER, on test-clean and test-other, respectively. On such tasks, a similar performance was also reported in [17, 19, 20].

4.1. Baseline vocabulary generation

We used the SentencePiece framework to generate single-character, BPE and Unigram subword sets. PASM subwords were generated using the authors' publicly available code[1]. PASM and PhIS vocabularies were converted to SentencePiece models and all data were encoded for training and decoded for evaluation with SentencePiece.

To ensure coverage, compatibility with the SentencePiece framework and fair comparisons, all vocabularies included all single characters (26 letters and apostrophe) as well as the whitespace token and three auxiliary symbols for unknown character, start and end of sentence. Using the parameters for Librispeech reported in [27], the PASM method returned 184 subwords, which were augmented with 28 single characters (26 letters, apostrophe, whitespace) and the three sentence piece markers. We rounded this set to 200 subwords by removing the PASM tokens that appeared less frequently in the encoded data. Marginalized counts of subword occurrences were used as weights for the PASM subwords.

4.2. Experiments with 200 subwords

Our first experiment is on the simple CTC model with a 5-layer encoder of 728 units per layer. The motivation behind this experiment was to verify and expand the results reported by [27] at a model that does not have language model rescoring or recurrence at the label level. We found that all subword methods improved WER compared to the character-based baseline (Table 2). Out of the subword methods, BPE showed the lowest improvement while Unigram showed the best accuracy.

Table 2: *Absolute and relative WER of the CTC experiment with 200 subwords.*

Testset		Char	BPE	PASM	Unigram	PhIS
clean	abs.	13.34	13.18	12.88	**12.78**	12.87
	rel.		1.20	3.49	**4.20**	3.52
other	abs.	32.91	32.45	32.68	**31.89**	32.36
	rel.		1.38	0.68	**3.10**	1.66

Further experiments were done with the more complex RNN-T model with 5 encoder layers of 640 units per layer and 2 decoder layers of 640 units per layer. To directly compare with the first experiment and the PASM generated models, we first show results with a small vocabulary of 200 subwords (Table 3). Similar to the first experiment, all subword vocabularies outperform the character-based baseline with BPE showing the lowest improvement. In contrast to the CTC experiment, pronunciation methods outperformed text-based subwords and the PhIS subwords achieved the best relative improvement of 11.13% at test-clean and 6.28% at test-other.

4.3. RNN-T experiment with 2500 and 4096 subwords.

Compared to vocabularies of few hundred subwords, a larger vocabulary might be beneficial as it can capture more regularities and model more full words. On the other hand, larger vocabularies will increase model parameters and potentially challenge resource-constrained implementations. In

[1] https://github.com/hainan-xv/PASM

Table 3: *Absolute and relative WER of the RNN-T experiment with 200 subwords.*

Testset		Char	BPE	PASM	Unigram	PhIS
clean	abs.	10.88	10.05	9.94	9.91	**9.67**
	rel.		7.59	8.60	8.92	**11.13**
other	abs.	26.68	26.33	25.08	25.63	**25.00**
	rel.		1.31	6.00	3.94	**6.28**

the case of RNN-T, vocabulary size only affects the size of the joint network as well as the decoder input embedding. Therefore, in this case, a larger vocabulary may not lead to a significant increase in total model size. In Table 4, we break down the RNN-T model size to different model components for different vocabulary sizes. Our RNN-T model has 5 encoder layers with 640 units each (39,211,648 parameters) and 2 decoder layers with 640 units each (14,321,536 parameters).

Table 4: *RNN-T model size for different subword sets.*

Vocabulary size	Decoder input embedding	Joint network	Total model size
200	51,456	128,841	53,713,481
2500	640,256	1,603,141	55,776,581
4096	1,048,832	2,626,177	57,208,193

Table 5 shows the performance of the different subword methods with larger vocabularies of 2500 and 4096 subwords. Note that relative WER is given against PhIS in this Table and that PASM was not included as the method does not generate vocabularies of this size. In both experiments, PhIS subwords outperform models trained with BPE or unigram subwords, achieving relative improvements that range from 1.61% to 15.21%.

Table 5: *Absolute and relative WER of the RNN-T experiment with 2.5k and 4k subwords.*

Testset		PhIS	BPE	Unigram
clean (2.5k subwords)	abs.	**9.07**	9.29	10.45
	rel.		-2.43	-15.21
other (2.5k subwords)	abs.	**24.15**	24.54	26.17
	rel.		-1.61	-8.36
clean (4k subwords)	abs.	**9.20**	10.13	10.31
	rel.		-10.11	-12.07
other (4k subwords)	abs.	**23.98**	25.19	24.79
	rel.		-5.05	-3.38

4.4. Analysis

We analyzed common tokens and differences in tokenization across the subword methods. Table 6 shows the number of similar units between all possible pairs of the four 200 subword candidate sets. There is a significant overlap between the subwords of BPE, Unigram and PhIS. On the contrary, PASM has lower overlap to the other methods. Analyzing the tokenized text, we found that PASM uses more labels per word than the other methods (BPE: 2.45; Unigram: 2.40; PhIS: 2.49; PASM:3.40) and rarely tokenizes a full word with a single token (percentage of words tokenized with single token, BPE: 44%, Unigram: 47%, PhIS: 50%, PASM: 4%). This is attributed to the way PASM aligns phonemes to letters. For example, in the word "the", the two phonemes will always be aligned to different letters and never merged as a single token.

Table 6: *Common subwords across methods.*

	BPE	PASM	Unigram	PhIS
PhIS	130	56	134	-
Unigram	137	49	-	
PASM	53	-		
BPE	-			

Finally, we present different segmentations of an example phrase in Table 7. In this example, PhIS avoids the phonetically ambiguous "$ough$", "gh" and "ugh" subwords. It is also able to capture the word "$look$" and use it as a subword to encode "$looking$". Similar to text-based methods, PhIS uses a single token for the word "the".

Table 7: *Segmentation examples across subword methods.*

Subwords	**Text**
Original	LOOKING THROUGH THE WINDOW
PASM	L|OO|K|I|NG TH|R|OUGH TH|E W|I|ND|OW
BPE	L|OO|K|ING TH|R|OU|GH THE W|IN|D|OW
Unigram	LO|O|K|ING TH|RO|UGH THE W|IN|D|OW
PhIS	LOOK|ING TH|R|OU|G|H THE W|IN|D|OW

5. Conclusions

We presented a method to generate phonetically induced subwords for end-to-end ASR applications and benchmarked our method against popular subword generation methods that are based on text statistics or on text-phoneme correspondence. Our results showed that the PhIS subwords outperform the other subword methods in systems with feedback over the labels, observing most improvement with vocabularies of 2500 and 4096 subwords. For future work, we plan to experiment with even larger subword sets and expand our method by experimenting with more grapheme candidates for each phonetic subword.

6. Acknowledgements

We would like to thank Hieu Nguyen, Carlos Ramirez and Rupak Vingesh Swaminathan for their valuable input.

7. References

[1] 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, pp. 369–376.

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

[3] A. Maas, Z. Xie, D. Jurafsky, and A. Ng, "Lexicon-free conversational speech recognition with neural networks," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2015, pp. 345–354.

[4] A. Hannun, C. Case, J. Casper, B. Catanzaro, G. Diamos, E. Elsen, R. Prenger, S. Satheesh, S. Sengupta, A. Coates, and A. Y. Ng, "Deep speech: Scaling up end-to-end speech recognition," *arXiv preprint arXiv:1412.5567*, 2014.

[5] N.-Q. Pham, T.-S. Nguyen, J. Niehues, M. Müller, and A. Waibel, "Very deep self-attention networks for end-to-end speech recognition," in *Interspeech 2019*, 2019, pp. 66–70.

[6] H. Soltau, H. Liao, and H. Sak, "Neural speech recognizer: Acoustic-to-word LSTM model for large vocabulary speech recognition," *arXiv preprint arXiv:1610.09975*, 2016.

[7] C. Yu, C. Zhang, C. Weng, J. Cui, and D. Yu, "A multistage training framework for acoustic-to-word model," in *Interspeech 2018*. ISCA, 2018, pp. 786–790.

[8] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1715–1725.

[9] P. Gage, "A new algorithm for data compression," *The C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.

[10] A. Radford, J. Wu, R. Child, D. Luan, and I. Sutskever, "Language models are unsupervised multitask learners," *Technical report, Open AI*, 2019.

[11] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 5149–5152.

[12] T. Kudo and J. Richardson, "SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, pp. 66–71.

[13] T. Kudo, "Subword regularization: Improving neural network translation models with multiple subword candidates," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018, pp. 66–75.

[14] T. Zenkel, R. Sanabria, F. Metze, and A. Waibel, "Subword and crossword units for CTC acoustic models," in *Interspeech 2018*, 2018, pp. 396–400.

[15] K. Rao, H. Sak, and R. Prabhavalkar, "Exploring architectures, data and units for streaming end-to-end speech recognition with RNN-transducer," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 193–199.

[16] C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4774–4778.

[17] Z. Xiao, Z. Ou, W. Chu, and H. Lin, "Hybrid CTC-attention based end-to-end speech recognition using subword units," in *2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, 2018, pp. 146–150.

[18] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved training of end-to-end attention models for speech recognition," in *Interspeech 2018*. ISCA, 2018, pp. 7–11.

[19] E. Lakomkin, J. Heymann, I. Sklyar, and S. Wiesler, "Subword regularization: An analysis of scalability and generalization for end-to-end automatic speech recognition," in *Interspeech 2020*, 2020, pp. 3600–3604.

[20] J. Drexler and J. Glass, "Subword regularization and beam search decoding for end-to-end automatic speech recognition," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6266–6270.

[21] W. Chan, Y. Zhang, Q. Le, and N. Jaitly, "Latent sequence decompositions," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[22] J. Drexler and J. Glass, "Learning a subword inventory jointly with end-to-end automatic speech recognition," in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6439–6443.

[23] A. Brown, *Understanding and Teaching English Spelling: A Strategic Guide*. Taylor & Francis Group, 2018.

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

[25] W. Wang, Y. Zhou, C. Xiong, and R. Socher, "An investigation of phone-based subword units for end-to-end speech recognition," in *Interspeech 2020*, 2020.

[26] M. Zeineldeen, A. Zeyer, W. Zhou, T. Ng, R. Schlüter, and H. Ney, "A systematic comparison of grapheme-based vs. phoneme-based label units for encoder-decoder-attention models," *arXiv preprint arXiv:2005.09336*, 2020.

[27] H. Xu, S. Ding, and S. Watanabe, "Improving end-to-end speech recognition with pronunciation-assisted sub-word modeling," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 7110–7114.

[28] C. Dyer, V. Chahuneau, and N. A. Smith, "A simple, fast, and effective reparameterization of IBM model 2," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2013, pp. 644–648.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, pp. 1735–1780, 1997.

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

[32] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S. Chang, K. Rao, and A. Gruenstein, "Streaming end-to-end speech recognition for mobile devices," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6381–6385.

[33] C.-F. Yeh, J. Mahadeokar, K. Kalgaonkar, Y. Wang, D. Le, M. Jain, K. Schubert, C. Fuegen, and M. L. Seltzer, "Transformer-transducer: End-to-end speech recognition with self-attention," *arXiv preprint arXiv:1910.12977*, 2019.