CONTEXTUAL RNN-T FOR OPEN DOMAIN ASR

Mahaveer Jain, Gil Keren, Jay Mahadeokar, Geoffrey Zweig, Florian Metze, Yatharth Saraf

Facebook AI, USA

{jainmahaveer,gilkeren,jaym,gzweig,fmetze,ysaraf}@fb.com

Abstract

End-to-end (E2E) systems for automatic speech recognition (ASR), such as RNN Transducer (RNN-T) and Listen-Attend-Spell (LAS) blend the individual components of a traditional hybrid ASR system - acoustic model, language model, pronunciation model - into a single neural network. While this has some nice advantages, it limits the system to be trained using only paired audio and text. Because of this, E2E models tend to have difficulties with correctly recognizing rare words that are not frequently seen during training, such as entity names. In this paper, we propose modifications to the RNN-T model that allow the model to utilize additional metadata text with the objective of improving performance on these named entity words. We evaluate our approach on an in-house dataset sampled from de-identified public social media videos, which represent an open domain ASR task. By using an attention model to leverage the contextual metadata that accompanies a video, we observe a relative improvement of about 16% in Word Error Rate on Named Entities (WER-NE) for videos with related metadata.

Index Terms: RNN-T, Deep Contextualization, Context biasing, E2E ASR.

1. Introduction

Present day ASR models using Deep Neural Networks (DNN) can be broadly classified into two frameworks: hybrid [1] and E2E [2, 3, 4]. A typical hybrid HMM-DNN system consists of three components trained individually: an acoustic model (AM) that estimates the posterior probabilities of Hidden Markov Model (HMM) states, a language model (LM) that estimates probabilities of word sequences, and a pronunciation model (PM) to map phonemes to words. These models are optimized independently [5] and then combined together using a Weighted Finite State Transducer (WFST) [6] for efficient decoding. In an E2E speech recognition model such as the RNN-T [2], a single neural network learns to map audio to text instead of using the distinct components of the hybrid systems. While this generally simplifies overall training and inference pipelines for ASR, E2E models tend to have difficulties with correctly recognizing words that do not appear frequently in paired audio-text training data [7, 8, 9]. Since hybrid ASR systems optimize AM, LM and PM components independently, they can address the rare word recognition issue by 1) training the LM component with large amounts of unpaired text data to model occurrences of rare words and 2) representing pronunciation in the PM.

In this work, we propose an attention-based context biasing approach to address the following underlying issues in the setting of an E2E RNN-T based ASR system:

- **Rare word recognition**: It is common for a vanilla RNN-T ASR system to make mistakes in recognizing words that occur infrequently in the training data. Assuming that *PyTorch* is a rare word in paired ASR training data, a vanilla RNN-T model might produce a hypothesis as *when you look at* ***pie towards*** *itself* whereas the true transcript is *when you look at* ***PyTorch*** *itself*. Although *PyTorch* is a rare word in training data, it's appearance in the video metadata can be used to recognize it correctly.
- **Disambiguation between similar words**: When acoustically similar words appear in similar contexts in the training data, it is difficult for the model to pick the correct word without additional biasing. E.g. the names *Sean* and *Shaun* might appear with similar frequencies in a similar context, but we might want to prefer one of them if it appears also in related text metadata.

Entity names often suffer from both these issues. Further, these often carry a high degree of semantic meaning relative to other words in the transcript, so it is important for ASR systems to recognize these correctly. Therefore, we measure the effectiveness of our approach on recognition of entity names. We aim to address these problems by biasing ASR using additional context from the accompanying text metadata of the video. This metadata is unstructured and potentially irrelevant to the speech being transcribed, so the ASR system needs to learn to selectively use or ignore it.

The rest of the paper is organized as follows: We review prior work around use of contextual words in E2E ASR in Section 2. We describe the base RNN-T model in Section 3. In Section 4, we propose changes to the RNN-T model to allow it to incorporate unpaired and unstructured text context via an attention mechanism. We show experiment results in Section 5 and further analyze the effectiveness of the proposed method in Section 6 by visualizing attention weights towards the metadata.

2. Prior Work

Prior work has leveraged contextual words either by on-the-fly (OTF) rescoring [10, 11, 12] or as an additional input to the DNN along with the audio. The first approach is generally referred to as Shallow Fusion whereas the latter as Deep Contextualization [7]. Our work falls in the latter category. It is most closely related to Contextual Listen, Attend And Spell (CLAS) [7], which also used context words from unpaired text to bias an E2E ASR model. The CLAS model was originally evaluated for closed domain ASR tasks like those used for virtual assistants by using entities such as contact names as context words. Further improvements to CLAS were done in [9] and [8] by using representations that leverage phonetic information as well. In this work, different from CLAS, we look at Deep Contextualization in the setting of an RNN-T ASR model, and evaluate our method on an open domain video ASR task using noisy text metadata from videos as context. In a closed domain use case such as making calls through an assistant, there is strong prior information about where entity names can appear in the utterance, whereas in our case the context words may appear anywhere in the conversational speech of the video. Deep con-

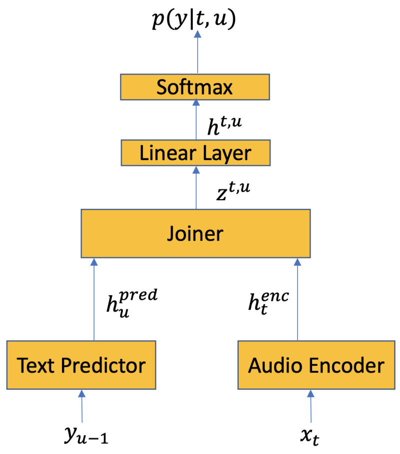

Figure 1: *RNN Transducer for ASR*

textualization of RNN-T was explored in [13] for keyword spotting use case, where the phoneme sequence of the keyword represented as a one-hot vector was used to attend to and recognize the target keyword. An alternate approach for using contextual metadata from videos to improve ASR is explored in [14], where lattices produced by a hybrid ASR system are rescored using metadata.

3. RNN Tranducer

The framework of RNN-T ASR system is illustrated in Fig. 1. RNN-T for ASR has three main components: Audio Encoder, Text Predictor and Joiner.

The Audio Encoder uses audio frame at x_t to produce audio embedding h_t^{enc} (Equation 1). The Audio Encoder used in this work is a stack of bi-directional LSTM (BLSTM) layers.

$$h_t^{enc} = f^{enc}(x_t) \tag{1}$$

The Text Predictor uses the last non-blank target unit y_{u-1} to produce embedding h_u^{pred} (Equation 2). The Text Predictor is a stack of LSTM layers in this work. We use sentence pieces as target units.

$$h_u^{pred} = f^{pred}(y_{u-1}) \tag{2}$$

The Joiner takes in the output of Audio Encoder and Text Predictor and combines them to produce an embedding $z^{t,u}$:

$$z^{t,u} = \phi(Uh_t^{enc} + Vh_u^{pred} + b) \tag{3}$$

U and V are matrices that are used to project audio and text embeddings to the same dimensions. ϕ is a non-linear function such as Relu [15] or tanh.

Finally, the joiner's output, $z^{t,u}$, is passed through a linear transformation followed by a softmax layer to produce a probability distribution over target units (y) , i.e. sentence pieces plus a special $blank$ symbol:

$$h^{t,u} = W_y z^{t,u} + b \tag{4a}$$

$$p(y|t,u) = softmax(h^{t,u}) \tag{4b}$$

By incorporating both audio and text for producing $p(y|t,u)$ (Equation 4b), RNN-T can overcome the conditional

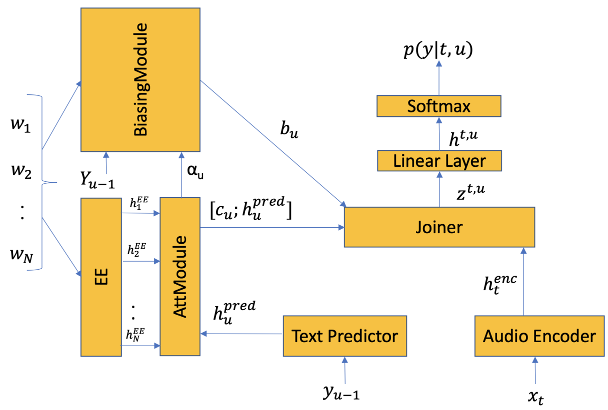

Figure 2: *Contextual RNN Transducer for ASR*

independence assumption of CTC models [16]. The emission of $blank$ as output unit results in an update of the audio embedding by moving ahead in time axis t whereas emission of non $blank$ results in a change in the text embedding. This results in various possible alignment paths as shown in the lattice of size $T*U$ in Figure 1 of [2]. The sum of probabilities of these paths gives the probability of an output sequence, Y, given the input sequence, X, where Y is the sequence of non $blank$ output target units and X is the input sequence of audio frames.

4. Contextual RNN-T

We modify the base RNN-T model described in Section 3 and add three additional components: an Embedding Extractor (EE), an Attention Module (AttModule) and (optionally) a Biasing Module (BiasingModule) as shown in Figure 2.

As in [7], each context word, w_i, is first represented as a sequence of target sentence piece units, e.g. the word "Jarred" may be mapped to *[Ja, r, re, d]*. This sequence is then fed to an BLSTM, and the last state of the BLSTM is used as the embedding of the given context word (shown as h_i^{EE} in Figure 2).

In the vanilla RNN-T system described in Section 3, probabilities over target units $p(y|t,u)$ (Equation (4b)) are conditionally dependent on the outputs of the Audio Encoder, h_t^{enc}, and Text Predictor, h_u^{pred}. In contextual RNN-T, we would like to make $p(y|t,u)$ conditionally dependent on contextual metadata words as well. This dependency can be achieved by incorporating the context word information into any of the Audio Encoder, Text Predictor and Joiner components. In this work, we explore incorporating the context word information into the Text Predictor and Joiner of the RNN-T.

An Attention Module (AttModule) is used to compute attention for each word in the metadata text. AttModule uses the predictor output for non-blank text history up to u (h_u^{pred}) and word embedding, h_i^{EE}, to compute attention weight, $e_{u,i}$, as shown in Equation (5b). We use location-aware attention that takes into account the attention weights from the previous predictor state, α_{u-1}, while computing alignments at the current step [4].

$$F = Q * \alpha_{u-1} \tag{5a}$$

$$e_{u,i} = w^{\intercal} tanh(Ah_u^{pred} + Bh_i^{EE} + Cf_i + b) \tag{5b}$$

$$\alpha_{u,i} = exp(e_{u,i}) / \sum_{j=1}^{N}(exp(e_{u,j})) \tag{5c}$$

F is output of convolution of α_{u-1} with matrix Q (Equation (5a)). f_i in Equation (5b) is used to denote the output for the ith word in F. A, B, C are matrices, b and w are vectors.

We next compute a *context vector*, c_u, [7, 13] as the weighted sum of *word embedding* as:

$$c_u = \sum_{i=1}^{N}(\alpha_{u,i} * h_i^{EE}) \tag{6}$$

.

We concat this context vector, c_u, to the output of the Text Predictor (h_u^{pred}). This results in a modification of the Joiner equation (3) to equation (7):

$$z^{t,u} = \phi(Uh_t^{enc} + V[c_u; h_u^{pred}] + b) \tag{7}$$

An additional biasing module (BiasingModule) may be used to find an active subset of the context words that have the same prefix as the last unfinished word in the text history (Y_{u-1}) to create an additional vector that biases the Joiner towards selecting from the subset of sentence pieces that correspond to the active context words using the attention values $\alpha_{u,i}$ from Equation (5c). For example, if the decoded form of Y_{u-1} is *Africa An* and the list of the context word is *Android, Antenna* and *Pytorch* then the active context words are *Android* and *Antenna*. The BiasingModule computes a biasing value, $bias_{u,i,k}$, for each sentence piece k of word w_i at a given text history Y_{u-1} as shown in Equation (8). $I_{SP}(w_i, Y_{u-1}, k)$ returns 1 if w_i is active word at Y_{u-1} and k is the sentence piece in w_i followed by the shared prefix, otherwise it returns 0. We then compute $bias_{u,k}$ for each sentence piece k at Y_{u-1} by summing over all context words as in Equation (9). A vector of all biases, b_u, with length equal to number of sentence pieces is then linearly projected and passed through an optional Dropout layer before feeding to the Joiner as shown in Equation (10).

$$bias_{u,i,k} = \alpha_{u,i} * I_{SP}(w_i, Y_{u-1}, k) \tag{8}$$

$$bias_{u,k} = \sum_{i=1}^{N}(bias_{u,i,k}) \tag{9}$$

$$z^{t,u} = \phi(Uh_t^{enc} + V[c_u; h_u^{pred}] + Dropout(Bb_u) + b) \tag{10}$$

Equations 4a and 4b remain unchanged.

5. Experiments

5.1. Dataset

The dataset used for our experiments was sampled from English videos shared publicly on Facebook. The data is de-identified by removing information such as the user who posted the video, and only use the content of the video and the text metadata for training and evaluation.

We segment the data in duration of 10 seconds for training and evaluation. Metadata words are obtained from the title and description text of the video, if available, after doing simple text cleaning and filtering such as removing words with hyperlinks in them. If a word in the text is capitalized then we also add its lowered case version as metadata word. We do not preserve the ordering of words in metadata both in training and evaluations. Each segment of the same video shares the same list of contextual words. We use about 8k hours of data for training and about 170 hours for evaluation. We further divide the evaluation test set based on whether there is any word in the reference for that segment that also appears in the metadata words for that video. The segments for which there is at least one common word are referred to below as the *CommonNonZero* set and the remainder as the *CommonZero* set. We present results on these two evaluation sets.

5.2. Model

The architecture of the Contextual RNN-T model from Section 4 used for the experiments in this paper is as follows. The Audio Encoder is a 4-layer BLSTM with 604 dimensions. We use subsampling of 2 across the time dimension after the first and second BLSTM layers. The output of the last layer of the BLSTM is projected to 1024 dimensions. The Text Predictor is a 2-layer LSTM of 256 dimensions whose output is also projected to 1024 dimensions. We used a token set consisting of 200 sentence pieces, trained using the sentence piece library [17]. The Embedding Extractor is a 2-layer BLSTM of size 100. The Attention Module has the following parameters: 1) Convolution (Q) with 2 out channels and kernel of size 1, 2) Attention is computed over 64 dimensions with A being of size $1024 * 64$, B of size $200 * 64$ and C of size $2 * 64$ (in Equations (5a), (5b) and (5c)). The output of the Biasing Module, when used, is also linearly projected to 1024 dimensions (Equation (10)). The baseline model (Figure 1) does not use Embedding Extractor, Attention Module or Biasing Module. All components of both the baseline and Contextual RNN-T models are trained from scratch.

The input to the network consists of globally normalized 80-dimensonal log Mel-filterbank features, extracted with 25ms FFT windows and 10ms frame shifts. Sentence piece encoding of each word, w_i, in the metadata is appended with a special sentence piece unit. We use the Adam optimizer [18], with a learning rate of 0.0002, and SpecAugment [19] with policy LB during training. All models were trained for 80 epochs. A beam size of 10 was used during inference.

5.3. Impact on WER-NE and WER

We measure performance of our models using WER and WER-NE on the two test sets described in Section 5.1. An in-house Entity tagger was used to tag named entities in transcripts and metadata.

As seen in Table 2, the Contextual RNN-T (Equation (7)) model (row 2) improves on WER-NE by about 13% relative compared to the baseline model (row 1) on the *CommonNonZero* evaluation set. Introducing BiasingModule (as in Equation (10)) improves WER-NE further by another 3%. As shown in Table 3, both WER and WER-NE for the *CommonZero* test set does not get significantly impacted by the Contextual RNN-T models when there is no intersection between the metadata words and the reference.

We also measure robustness of our system using precision and recall of the emission of context words in the model's hypotheses. A *True Positive* occurs when a context word from the metadata of the video is correctly output by the model as compared to the reference. A *False Positive* occurs if the model outputs a context word but it does not appear in the reference. We show aggregated precision and recall over both test sets for

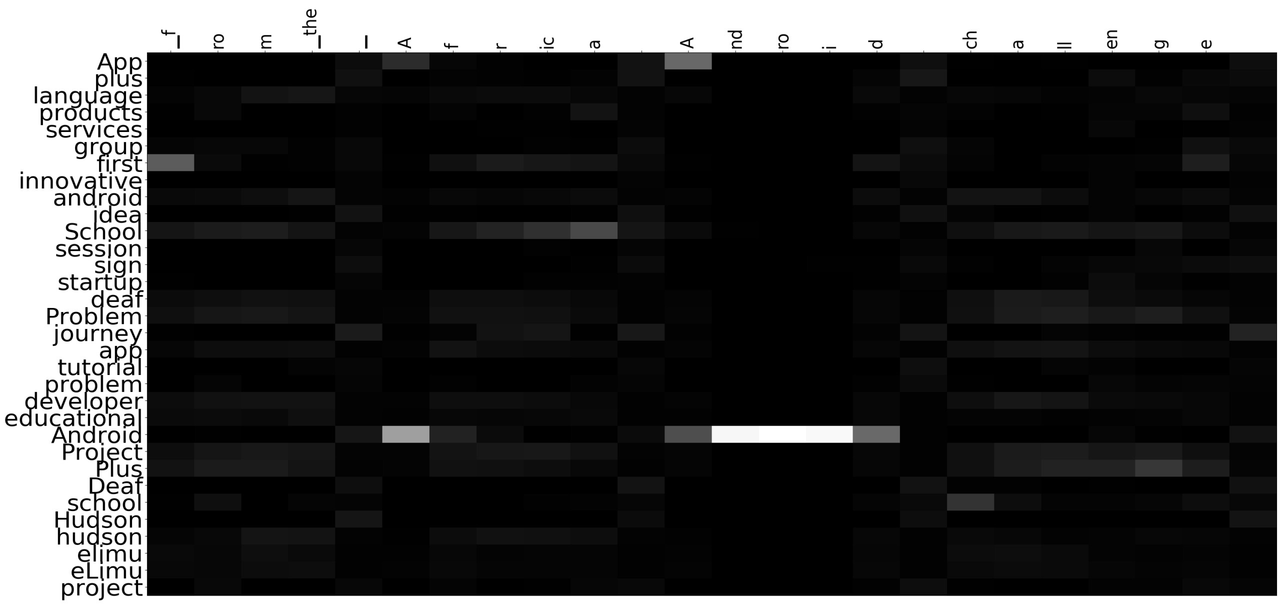

Figure 3: *Visualizing attention weights, $\alpha_{u,n}$ from Equation* (5c)*, for the example in Table 1, row 1. The x-axis shows the target units from the hypothesis output by the Contextual RNN-T Model, and the y-axis shows the contextual metadata words (w_i). Darker colors represent values close to zero while brighter colors represent values closer to 1.*

Reference Snippet	Baseline Output	Contextualization Output	Metadata Words (truncated)
from the **Africa Android** Challenge	from the **Africa** and red challenge	from the **Africa Android** challenge	innovative, School, language, Plus, app, tutorial, products, educational, startup Android, problem, ...
its very intuitive so when you look at **PyTorch** itself	its very intuitive so when you look at pie towards itself	its very intuitive so when you look at **PyTorch** itself	experiences, novel, PyTorch updates, Facebook, machine, AI, language, research, ...

Table 1: *Comparing outputs generated by the baseline and Contextual RNN-T models. Named entities are represented with bold font in these examples.*

triggering of the context words in Table 4. We see an improvement in recall of 8.3% and degradation in precision by 1.3% relative for the best contextual model compared to the baseline.

Model	WER	WER-NE
Baseline	16.04	24.69
Contextual RNN-T w/o BiasingModule	15.45	21.41
Contextual RNN-T with BiasingModule	15.37	20.66

Table 2: *WER and WER-NE results on CommonNonZero test set*

Model	WER	WER-NE
Baseline	23.07	29.18
Contextual RNN-T w/o BiasingModule	22.95	29.71
Contextual RNN-T with BiasingModule	22.89	29.93

Table 3: *WER and WER-NE results on CommonZero test set*

Model	Precision	Recall
Baseline	0.934	0.844
Contextual RNN-T w/o BiasingModule	0.920	0.898
Contextual RNN-T with BiasingModule	0.922	0.914

Table 4: *Precision and Recall for context words across both test sets*

6. Analysis

To understand better what the Contextual RNN-T model is doing, we visualize attention values for a few test segments where it correctly recognizes named entities that the baseline model makes errors on. These examples are shown in Table 1.

For the example shown in row 1 of Table 1, both the Contextual and baseline models are able to recognize common entities such as **Africa**. However, the baseline model has difficulties in recognizing entities that are not frequent in training data set, such as **Android** and **PyTorch**. Since **Android** appears in the metadata, the Contextual RNN-T model is able to attend to it and transcribe it correctly. This can be seen in the visualization of attention given to the words in the metadata at each output target unit (u) of the Contextual RNN-T Model in Figure 3.

7. Conclusion

We show that contextual metadata text, even if it is noisy, can be used to improve recognition of named entities for a challenging open domain ASR task such as social media videos within the framework of an E2E RNN-T ASR model. Some future explorations could be: i) Using contextual embeddings from other modalities such as images from video, ii) Using semantic embeddings to represent the metadata.

8. References

[1] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Springer Science & Business Media, 2012, vol. 247.

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

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4960–4964.

[4] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, 2015, pp. 577–585.

[5] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.

[6] M. Mohri, F. Pereira, and M. Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[7] G. Pundak, T. N. Sainath, R. Prabhavalkar, A. Kannan, and D. Zhao, "Deep context: end-to-end contextual speech recognition," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 418–425.

[8] Z. Chen, M. Jain, Y. Wang, M. L. Seltzer, and C. Fuegen, "Joint grapheme and phoneme embeddings for contextual end-to-end asr," in *INTERSPEECH*, 2019.

[9] A. Bruguier, R. Prabhavalkar, G. Pundak, and T. N. Sainath, "Phoebe: Pronunciation-aware contextualization for end-to-end speech recognition," 2019.

[10] D. Zhao, T. N. Sainath, D. Rybach, D. Bhatia, B. Li, and R. Pang, "Shallow-fusion end-to-end contextual biasing," *Submitted to Interspeech*, vol. 2019, 2019.

[11] I. Williams, A. Kannan, P. S. Aleksic, D. Rybach, and T. N. Sainath, "Contextual speech recognition in end-to-end neural network systems using beam search." in *Interspeech*, 2018, pp. 2227–2231.

[12] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6381–6385.

[13] Y. He, R. Prabhavalkar, K. Rao, W. Li, A. Bakhtin, and I. McGraw, "Streaming small-footprint keyword spotting using sequence-to-sequence models," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 474–481.

[14] D.-R. Liu, C. Liu, F. Zhang, G. Synnaeve, Y. Saraf, and G. Zweig, "Contextualizing asr lattice rescoring with hybrid pointer network language model," in *Submitted ro Proc. Interspeech*, 2020.

[15] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[16] 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*. ACM, 2006, pp. 369–376.

[17] 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.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.