RefTextLAS: Reference Text Biased Listen, Attend, and Spell Model For Accurate Reading Evaluation

Phani Sankar Nidadavolu, Na Xu, Nick Jutila,
Ravi Teja Gadde, Aswarth Abhilash Dara, Joseph Savold, Sapan Patel,
Aaron Hoff, Veerdhawal Pande, Kevin Crews, Ankur Gandhe, Ariya Rastrow, Roland Maas

Amazon Alexa

{phanisn,amznnxa,jutila,gadderav, aswardar, jsavold, sapanp, aaronhf, pveer, kevcrews, aggandhe, arastrow, rmaas}@amazon.com

Abstract

We present an automatic reading evaluator that listens to novice young readers and offers feedback based on the reading accuracy. In order to not discourage the reader, the model should not misrecognize correctly read tokens (false rejects), which may come at the expense of tolerating some reading mistakes (false accepts). To minimize the former, we explore two approaches to provide reference text – the text user is supposed to read – as context to automatic speech recognition (ASR) models: 1) a finite state transducer (FST) based error detection procedure, that restricts the grammar to tokens from reference text and an out of vocabulary (OOV) *catch-all* token, and 2) RefTextLAS, an attention-based end-to-end (E2E) ASR model, that takes tokens from reference text as an additional input. Our biasing approaches reduce false reject rate (FRR) by 38-56% on an in-house dataset compared to a baseline hybrid model, whose language model (LM) is trained on book texts, with 65-82% compromise on false accept rate (FAR). To reduce FAR, we present an ensemble approach that uses both baseline and RefTextLAS models to determine reading accuracy. The ensemble approach limits the relative degradation on FAR to 26.4% while providing a 42.7% improvement on FRR.

Index Terms: automatic speech recognition, automatic reading assessment, contextual biasing

1. Introduction

The popularity of commercial virtual assistants, like Amazon Alexa and Google Home, has been on the rise in the recent past. The voice interaction capability of such devices can be used to develop reading assessment applications, such as Alexa's *Reading Sidekick*, where a user reads a reference text out loud. The virtual assistant (VA) then recognizes the user's speech using an ASR model, compares it with the reference text, provides encouragement when the reading is accurate (example 1 in Table 1), and offers support when the user struggles (example 2). Accurate ASR is very important for the success of this application. Conventional ASR models [1, 2, 3], that take only audio features as input, can be used to obtain token level reading accuracy, by comparing the model output and truth (transcription) with the reference text. Of the two possible errors the model could make – 1) misrecognizing correctly read token, a false reject (FR) scenario (example 3), and 2) recognizing misread token as expected token from the text, a false accept (FA) scenario (example 4) – the former is the most important metric to minimize, especially in the case of young readers. A model with high FRs could demotivate the reader, which may result in reading discontinuation.

We explore two approaches to provide reference text as context for ASR models to minimize FRs. We take into account that we lack an in-domain training dataset that contains reference text and transcription for each utterance. First, we modify the inference procedure of a pre-trained hybrid ASR model [1] by defining a <*catch-all*> OOV token and build a grammar (G.fst) [4] from tokens in the reference text and the OOV token. The search graph is then modified to output any unexpected tokens in the user's speech as <*catch-all*> token. As a result, if the user reads *I am angry* instead of *I am hungry*, the model outputs *I am* <*catch-all*>. The approach shows promising performance in minimizing FRs, compared to an out-of-domain Listen, Attend, and Spell (LAS) model [2] and a baseline hybrid ASR model whose LM is trained on book texts [5]. Since the biasing is done during decoding, the procedure does not require any special training procedures or in-domain data, and can be applied to any pre-trained hybrid ASR model.

Second, inspired by the recent contextual LAS (CLAS) model [6], we present a RefText biased LAS (RefTextLAS) model that biases recognition towards the reference text using an additional encoder. For training RefTextLAS, we use an out-of-domain ASR dataset that contains only audio and transcription, and sample tokens for biasing from the transcription of each utterance. Similar to the hybrid model with <catch-all> token, RefTextLAS also lowers FRs, but with a compromise on FAs. To address this, we explore two techniques – 1) use ASR errors as negative examples for biasing [7] RefTextLAS, and 2) an ensemble approach that uses RefTextLAS and token level confidence score (ConfScore) [8] of the baseline model for evaluation. To the best of our knowledge, our work is the first large scale effort to use contextual ASR for automatic token level reading assessment.

In prior works on related applications, authors in [9] and [10] proposed various linear regression models to automatically predict human annotated scores of users reading sentences. A phoneme-level text conditioned transformer was proposed in [11] for pronunciation assessment of language learners. A hybrid ASR model with LM trained on book corpus is used in [5] to evaluate reading accuracy of young Italian students. On the other hand, contextual biasing for personalizing E2E ASR models is an active area of research [6, 12, 13, 14, 15, 7].

2. Technical Details

2.1. FST based reading error detection

Possible decode paths through lexicon FST [4] contains a path for in vocabulary (IV) words, and a monophone path to detect OOV words [16]. We define an OOV token that is used as part of

Table 1: *Examples of a user reading a reference text to a virtual assistant (VA), and the feedback provided by the VA. The user and model errors are shown in brown and red respectively. During offline evaluation, comparing individual tokens in output and truth with tokens in reference text will lead to four scenarios – true accept (TA), true reject (TR), false accept (FA) and false reject (FR) (details in Section 1 and Section 3.2). FR errors should be minimized to avoid discouraging the reader. Best viewed in color.*

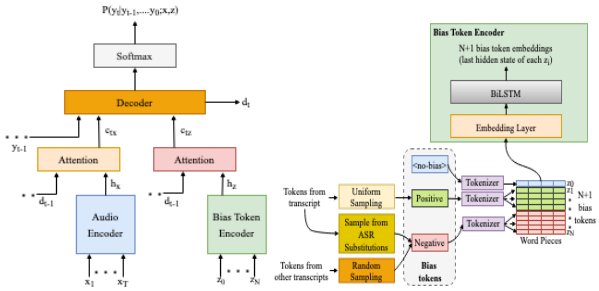
	Example 1	**Example 2**	**Example 3**	**Example 4**
Reference Text	*I am hungry*	*I am hungry*	*I am hungry*	*I am hungry*
User (truth)	*I am hungry*	*I am angry*	*I am hungry*	*I am angry*
Model output	*I am hungry*	*I am angry*	*I am angry*	*I am hungry*
Validation	TA TA TA	TA TA TR	TA TA FR	TA TA FA
VA response	Great! Let us continue	Not quite! Let us try again (or) Let me read it for you	Not quite! Let us try again (or) Let me read it for you	Great! Let us continue

the grammar (G.fst) which outputs a <*catch-all*> token when traversed. Instead of defining a single token's OOV monophone sequence, it is an n-gram that supports any possible monophone sequence that the lexicon is capable of creating. In terms of the search graph, at the beginning of each word, this allows to branch off to a monophone *catch all* fallback path in parallel to the IV and OOV detection paths [16]. While there is no recognition text associated with the <*catch-all*>, the monophone sequence is still available. As a result, if the user reads *I am angry* instead of *I am hungry*, the model outputs *I am* <*catch-all*> and ideally have a monophone sequence matching "*angry*". During decoding, the G.fst would be built from the reference text and the <*catch-all*> OOV. The cost of traversing the <*catch-all*> path is tunable. A low cost assigned to this path generates more <*catch-all*> tokens, resulting in high FRs. On the other hand, a very high cost assigned to this path results in very high FAs. We term this model as *hybrid model with catch-all token*.

2.2. All-neural reading error detection using RefTextLAS

RefTextLAS, shown in Figure 1(a), assumes access to audio features $x = x_1, .., x_T$, bias tokens $z = z_0, z_1, .., z_N$, and corresponding transcripts $y = y_1, .., y_L$ for each utterance. It uses an attention-based encoder-decoder architecture with two encoders: an audio encoder and a bias token encoder (Figure 1(b)) to compute embeddings for audio inputs x and bias tokens z respectively. Each encoder is associated with its own attention mechanism. At each step, the decoder simultaneously attends to audio and bias token embeddings, using decoder state d_t as attention-query. The model outputs a probability distribution $P(y|x, z)$, and is trained to minimize $-log(P(y|x, z))$ loss.

The architecture is designed to bias ASR towards the reference text using a bias token encoder. During training, since our training corpora does not contain any reference text, as shown in Figure 1(b), we uniformly sample bias tokens for each utterance from its transcription – referred as *positive* (bias) tokens. By partially biasing the model towards the transcript during training, it relies on the audio encoder alone to accurately recognize any misread and additional tokens spoken by the user. In the latter case, the model operates in a *no bias* mode, and attends to a filler <*no-bias*> token [13, 6]. As we show in Section 3.3.1, limiting biasing to positive and <*no-bias*> tokens alone results in high FAs. To overcome this, we augment positive tokens with some *negative* tokens sampled from two other sources – 1) for a given token in the transcript, include possible substitution (SUB) errors an ASR model can make when available, or 2) include random tokens from other transcripts (details in Section 3.3.2). During inference, all the tokens from the reference text are used as bias tokens (no uniform sampling) along with the <*no-bias*> token. Each bias token is converted into word pieces (WPs), and then to embeddings. The sequence of



(a) RefTextLAS (b) Bias token encoder

Figure 1: *Architecture of RefTextLAS and bias token encoder. Best viewed in color.*

word piece embeddings for each bias token will then be processed by a bidirectional LSTM (BiLSTM), and the last hidden state would be used as token embedding for attention. The audio encoder and corresponding attention mechanism are similar to LAS model [2].

3. Experiments and Results

3.1. Implementation Details

The architecture of RefTextLAS, shown in Figure 1(a), is as follows: the audio encoder has 5 long short-term memory (LSTM) layers with 1024 neurons in each layer. We use Log-Mel filterbank energies (LFBE) of 64 dimensions, with a window size of 25ms and hop size of 10ms, as features. Each frame is stacked 2 frames to the left and downsampled by a factor of 3 to have a frame rate of 30ms. The bias token encoder, shown in Figure 1(b), has one BiLSTM layer with 64 neurons. The embedding layer's dimension is 64. The decoder is a 2-layer LSTM with 768 hidden units and an output vocabulary projection of 4000-dimensions which correspond to 4000 WPs. We use multi-head attention [17] with 4 heads per encoder. To compare RefTextLAS, we also train a LAS model without biasing (audio encoder alone) with same details as above.

The training details of both the LAS models are as follows: we use 60 thousand (K) hours of de-identified far-field in house data for training. Development set, used to observe loss after each epoch, consists of 50 hours of speech. The models are randomly initialized and trained for 80 epochs (400K steps). We use an adaptive variant of SpecAugment [18], as proposed in [19], and Adam algorithm [20] for optimization. We use a rise-hold and decay learning rate scheduler where the learning rate linearly increases from 1e-7 to 5e-4 (warm-up) for 3K steps, stays constant at 5e-4 for 150K steps (hold), and then exponentially decays to final learning rate of 1e-3 / 16.0 at 620K steps.

As a baseline, we train the LM of a well-trained hybrid

model used in production, similar to [21], on text extracted from 7500 books. We call this model *hybrid model with LM adaptation*. For building hybrid model with <*catch-all*> token, we apply the procedure mentioned in Section 2.1 to a well-trained production model, similar to [21].

For evaluation, we use de-identified in house data containing 1417 utterances, collected by assigning users to read reference texts sampled from books. Each utterance contains reference text and transcription for the audio. The dataset consists of 5 hours of audio, with mean (median) duration of 12.6 (10.3) seconds. The mean (median) number of tokens in reference text and transcription are 23.9 (19.0) and 19.2 (15.0) respectively. All the models are evaluated on the last epoch.

3.2. Evaluation metrics

To compare the performance of all the models in this work, we use two metrics – FAR and FRR. The computation procedure using the reference text, transcription (truth) and ASR output is as follows: first, we separately align the transcription[1] and model output with the reference text. We, then, obtain a confusion matrix by comparing the individual tokens in these two aligned sequences, which leads to four possible scenarios as shown in Table 1: 1) true accept (TA) – both the user read token (truth) and the token in model output matches with the corresponding token in the reference text, 2) true reject (TR) – the truth and output does not match with the reference text, 3) false reject (FR) – the truth matches with the reference text but output differs from it, and finally 4) false accept (FA) – the truth does not match with the reference text but model output matches with it. We don't consider repetitions (insertions) as reading mistakes, as it is common for some readers (children, for instance) to repeat certain tokens. We also don't consider reading a sentence partially (deletions made by the user) as a reading mistake. FRR and FAR are computed as follows:

$$FRR = \frac{FR}{(TA + FR)}; FAR = \frac{FA}{(FA + TR)} \quad (1)$$

Instead of presenting the results using absolute FAR and FRR, we use *relative difference in FAR* (rFAR) and *relative difference in FRR* (rFRR). Given the FRRs of candidate and baseline systems, say FRR_{cand} and FRR_{ref} respectively, rFRR is defined as $(FRR_{cand} - FRR_{ref}) * 100 / FRR_{ref}$. rFRR<0 indicates that the candidate system has lower FRR than the baseline (more negative, the better and more positive, the worse). rFAR is computed and interpreted similarly.

3.3. Results

As we show below, there exists a trade-off between the two metrics. We aim to build models with low rFRR but without a heavy compromise on FAR (high rFAR), as discussed in Section 1.

3.3.1. Comparison of hybrid and RefTextLAS models

Table 2 presents results of hybrid, LAS and RefTextLAS models. We consider hybrid model with LM adaptation as baseline to compute rFAR and rFRR[2]. Compared to the baseline (row E1), LAS model trained on out-of-domain corpora (E0) has 36.5% and 7.3% higher rFRR and rFAR respectively. Hybrid model with *catch-all* token (E2) has significantly lower rFRR (-38.2%) but higher rFAR (65.7%). The results indicate that incorporating domain knowledge from the books improves rFRR (E1 and E2), and explicit biasing towards the reference text lowers rFRR significantly but with a compromise on rFAR (E2).

Table 2: *Comparison of hybrid, LAS and RefTextLAS models. RefTextLAS is biased with positive tokens and <no-bias> during training, and all tokens from reference text and <no-bias> during evaluation. We use metrics rFRR and rFAR. More negative, the better and more positive, the worse (details in Section 3.2)*

ID	Model	rFRR	rFAR
E0	**LAS**	+36.5	+7.3
	Hybrid model		
E1	with LM adaptation	0.0	0.0
E2	with *catch-all* token	-38.2	+65.7
	RefTextLAS with bias limits		
E3	[0, 50]	-39.1	+73.3
E4	[40, 60]	-54.2	+144.0
E5	[100, 100]	-65.0	+162.0

Our motivation to work with RefTextLAS is to improve (reduce) rFRR compared to the hybrid model with *catch-all* token, and to obtain similar rFAR as the hybrid model. For training RefTextLAS, as mentioned in Section 2.2, we uniformly sample positive tokens from the transcripts. To observe the impact of number of positive tokens on the performance, we train different models by further subsampling them. More specifically, instead of providing all positive tokens as input to the bias token encoder, we randomly select $x \in [minBias, maxBias]$ percentage of tokens, where $minBias$ and $maxBias$ correspond to the minimum and maximum percentage of positive tokens respectively. Setting the bias limits $[minBias, maxBias]$ to $[0, 50]$ implies that at least 0 (no biasing) and utmost half of positive tokens are given as input to the text encoder. On the other hand, setting them to $[100, 100]$ means that all the positive tokens are given as input – a more aggressive biasing scenario. In all cases, the <*no-bias*> token is always present in the input. During inference, all the tokens from the reference text and <*no-bias*> are given as bias tokens. As shown in Table 2, RefTextLAS with bias limits $[0, 50]$ (E3) offers comparable performance to the hybrid model with *catch-all* token (E2). As we increase the bias limits, rFRR improves but with a heavy compromise on rFAR (E4 and E5). Of all the models, hybrid ASR model with *catch-all* token (E2) and RefTextLAS models (E3-E5) yielded significantly lower rFRR, with the later providing much larger improvements. Since our main goal is low rFRR, we continue our experiments with RefTextLAS by providing all the positive tokens as input to the text encoder (E5), instead of a subset of them. However, in order to reduce the high rFAR observed in E5, we investigate using negative bias tokens as discussed below.

3.3.2. Biasing RefTextLAS with negative tokens to reduce rFAR

To lower the rFAR, we modify our bias tokens sampling strategy to augment positive tokens with some negative tokens (shown in Figure 1(b)). The procedure to sample negative tokens is as follows: we first use a pre-trained ASR model, similar to [22], to decode several test corpora (different from the one used for reading evaluation). For each token in test corpora that faced SUB errors, we gather the list of all substituted tokens. We, thus, obtain a mapping of token to possible SUBs for a total

[1] We adopted an open-source sequence comparator for aligning the text sequences (source: https://blog.robertelder.org/diff-algorithm/)

[2] Due to company internal confidentiality policies we cannot report absolute FAR and FRR metrics. Both the metrics of baseline system are slightly higher than 15% absolute, with lower FAR than FRR

Table 3: *Impact of negative biasing on rFAR and rFRR*

ID	# of negative tokens during Training	# of negative tokens during Decoding	RefTextLAS rFRR	RefTextLAS rFAR
No negative tokens				
E5	0	0	-65.0	+162.0
Negative tokens during decoding only				
E6	0	6	-55.8	+158.5
E7	0	10	-48.0	+153.8
Negative tokens during training only				
E8	3	0	-61.9	+108.5
E9	5	0	-60.3	+102.6
Negative tokens during training and decoding				
E10	5	6	-56.7	+94.3
E11	5	10	-55.5	+81.5

of 23320 tokens. During training, we first get a list of *unsampled* tokens – all the tokens from the transcript except the positive tokens – for each utterance. For each unsampled token, we randomly select 5 possible SUBs (obtained using the procedure above) if they exist, or select 5 random tokens from other transcripts, otherwise. The number of negative tokens for each utterance depends on the number of unsampled (total − number of positive) tokens. During inference, since, we use all the tokens from the reference text for biasing (no unsampled tokens), for each utterance we select a fixed number of negative examples randomly from all possible SUBs of tokens from the reference text and random tokens from other reference texts.

We analyze the impact of negative tokens and their quantity in three stages – during decoding only, during training only and during both. Results are presented in Table 3. Firstly, presenting negative tokens during decoding only improved rFAR (E6 and E7 compared to E5), but rFRR degraded significantly (-65% for E5 compared to -48% for E7). Secondly, presenting negative tokens during training only improved rFAR significantly (162.0 for E5 compared to 108.5 for E8 and 102.6 for E9) with slight compromise on rFRR (-65.0 for E5 vs -60.3 for E9). Finally, giving negative tokens during training and decoding lowered rFAR further with some compromise on rFRR (E10 and E11). E11, with 5 and 10 negative tokens during training and decoding respectively, improved rFAR from 162.0 to 81.5, with degradation on rFRR (-65.0 to -55.5). The results suggest that using more negative examples, as suggested in [7], to lower rFAR comes at the expense of increase in rFRR. On the other hand, as discussed in Section 3.3.1, biasing with positive tokens reduces the rFRR, but with a compromise on rFAR.

3.3.3. Ensemble approach

Models without explicit biasing towards the reference text (E0 and E1) yielded lower rFAR and higher rFRR compared to the models with biasing (E2-E11). We now experiment with an ensemble approach, where we make use of both the baseline system with lower rFAR (E1) and RefTextLAS in E11 to lower the metrics further. The approach makes use of token level ConfScore [8] of baseline model and a tunable threshold Δ as explained below. We first modify the baseline evaluation logic (gray highlighted section in Table 4) as follows: when the output token of baseline does not match with the reference text, instead of rejecting it, we check the ConfScore (ranging in 0-999). We reject the token only when the ConfScore is above a certain threshold. Otherwise, we accept the token (benefit of doubt given to the user since the model is not confident of its prediction). Observe that setting Δ to 0 (low), rejects all tokens (low rFAR) and setting it high (800) accepts most tokens (high rFAR).

Table 4: *Ensemble approach based on ConfScore and threshold (Δ). Highlighted section in gray is used to evaluate the baseline. Best viewed in color*

Token in RefTextLAS	Token in Baseline == Ref Text	Token in Baseline != Ref Text
== Ref Text	accept	accept if ConfScore $\leq \Delta$ reject if ConfScore $> \Delta$
!= Ref Text	accept	reject

In the ensemble approach (entire Table 4), when output of both RefTextLAS and baseline agree (disagree) with the reference text we accept (reject) the token. When output of RefTextLAS does not match with the reference text but output of baseline matches with it, we accept the token (improves rFRR). To improve rFAR, instead of accepting the output token of RefTextLAS when it matches with the reference text but baseline's output token does not, we check the ConfScore of baseline. As explained above, we accept the token only when the baseline is not confident of its prediction.

Table 5: *Results with ensemble approach*

Δ	Baseline rFRR / rFAR	RefTextLAS rFRR / rFAR	Ensemble rFRR / rFAR
0	0.0 / 0.0		-42.7 / +26.4
300	-21.9 / +118.2	-55.5 / +81.5	**-57.7 / +69.0**
600	-34.4 / +198.6	(E11)	-66.6 / +82.2
800	-39.7 / +292.5		-70.1 / +103.5

Results are presented in Table 5. Firstly, for the baseline, as we increase Δ, rFAR degrades severely (292.5% at Δ = 800), but the improvement in rFRR is not very significant ($\sim$-40%). Ensemble approach at Δ = 0 shows only 26.4% degradation in rFAR compared to the baseline, and with -42.7% improvement in rFRR. Similar to the baseline, the ensemble approach also shows some degradation in rFAR as we increase Δ. At Δ = 300, the ensemble approach yielded similar rFRR as RefTextLAS (E11), but with a lower rFAR. More importanly, rFAR is similar to hybrid model with catch-all path (E2 in Table 2) but ensemble system has better rFRR (-57.7% instead of -38.2%). At Δ = 600, the ensemble system yielded similar rFAR as RefTextLAS but with lower rFRR (-66.6% instead of -55.5%). Results suggest that by tuning the threshold Δ, we could leverage the advantages of both the baseline (low rFAR) and RefTextLAS (low rFRR) models.

4. Conclusions

We explored two ASR models conditioned on the reference text for automatic reading evaluation: 1) a hybrid model with *catch-all* token, and 2) a reference text biased LAS (RefTextLAS) model. To train RefTextLAS, we sampled positive bias tokens from the corresponding transcript for each utterance and negative tokens from the possible ASR substitutions. Both the models significantly reduced FRR. We observed that incorporating positive and negative tokens led to a trade-off between rFRR and rFAR (more the positive tokens, rFRR was better, and more the negative tokens, rFAR was better). In the future, instead of contextual biasing, we will explore using a spelling correction model [23] on the output of the baseline model.

5. References

[1] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[2] 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.

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

[4] M. Mohri, F. Pereira, and M. Riley, "Speech recognition with weighted finite-state transducers," in *Springer Handbook of Speech Processing*. Springer, 2008, pp. 559–584.

[5] O. Mich, N. Mana, R. Gretter, M. Matassoni, and D. Falavigna, "Automatically assess children's reading skills," in *Proceedings of the 1st Workshop on Tools and Resources to Empower People with REAding DIfficulties (READI)*, 2020, pp. 20–26.

[6] 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.

[7] U. Alon, G. Pundak, and T. N. Sainath, "Contextual speech recognition with difficult negative training examples," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6440–6444.

[8] P. Swarup, R. Maas, S. Garimella, S. H. Mallidi, and B. Hoffmeister, "Improving asr confidence scores for alexa using acoustic and hypothesis embeddings." in *Interspeech*, 2019, pp. 2175–2179.

[9] J. Proença, C. Lopes, M. Tjalve, A. Stolcke, S. Candeias, and F. Perdigão, "Automatic evaluation of reading aloud performance in children," *Speech Communication*, vol. 94, pp. 1–14, 2017.

[10] M. P. Black, J. Tepperman, and S. S. Narayanan, "Automatic prediction of children's reading ability for high-level literacy assessment," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 1015–1028, 2010.

[11] Z. Zhang, Y. Wang, and J. Yang, "Text-conditioned transformer for automatic pronunciation error detection," *Speech Communication*, vol. 130, pp. 55–63, 2021.

[12] A. Bruguier, R. Prabhavalkar, G. Pundak, and T. N. Sainath, "Phoebe: Pronunciation-aware contextualization for end-to-end speech recognition," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6171–6175.

[13] K. Mysore 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 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022.

[14] A. Gourav, L. Liu, A. Gandhe, Y. Gu, G. Lan, X. Huang, S. Kalmane, G. Tiwari, D. Filimonov, A. Rastrow *et al.*, "Personalization strategies for end-to-end speech recognition systems," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7348–7352.

[15] 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, pp. 3490–3494.

[16] I. Bazzi, "Modelling out-of-vocabulary words for robust speech recognition," Ph.D. dissertation, Massachusetts Institute of Technology, 2002.

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

[18] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 702–703.

[19] D. S. Park, Y. Zhang, C.-C. Chiu, Y. Chen, B. Li, W. Chan, Q. V. Le, and Y. Wu, "Specaugment on large scale datasets," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6879–6883.

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

[21] M. Van Segbroeck, H. Mallidih, B. King, I.-F. Chen, G. Chadha, and R. Maas, "Multi-view frequency lstm: An efficient frontend for automatic speech recognition," *arXiv preprint arXiv:2007.00131*, 2020.

[22] J. Guo, G. Tiwari, J. Droppo, M. Van Segbroeck, C.-W. Huang, A. Stolcke, and R. Maas, "Efficient minimum word error rate training of rnn-transducer for end-to-end speech recognition," *arXiv preprint arXiv:2007.13802*, 2020.

[23] M. Namazifar, J. Malik, L. E. Li, G. Tur, and D. H. Tür, "Correcting automated and manual speech transcription errors using warped language models," *arXiv preprint arXiv:2103.14580*, 2021.