♠ RED-ACE: Robust Error Detection for ASR using Confidence Embeddings

Zorik Gekhman* **Dina Zverinski*** **Jonathan Mallinson** **Genady Beryozkin**
Google Research
{zorik,zverinski,jonmall,genady}@google.com

Abstract

ASR Error Detection (AED) models aim to post-process the output of Automatic Speech Recognition (ASR) systems, in order to detect transcription errors. Modern approaches usually use text-based input, comprised solely of the ASR transcription hypothesis, disregarding additional signals from the ASR model. Instead, we utilize the ASR system's word-level confidence scores for improving AED performance. Specifically, we add an ASR Confidence Embedding (ACE) layer to the AED model's encoder, allowing us to jointly encode the confidence scores and the transcribed text into a contextualized representation. Our experiments show the benefits of ASR confidence scores for AED, their complementary effect over the textual signal, as well as the effectiveness and robustness of ACE for combining these signals. To foster further research, we publish a novel AED dataset consisting of ASR outputs on the LibriSpeech corpus with annotated transcription errors.[1]

1 Introduction

Automatic Speech Recognition (ASR) systems transcribe audio signals, consisting of speech, into text. While state-of-the-art ASR systems reached high transcription quality, training them requires large amounts of data and compute resources. Fortunately, many high performing systems are available as off-the-shelf cloud services. However, a performance drop can be observed when applying them to specific domains or accents (Khandelwal et al., 2020; Mani et al., 2020), or when transcribing noisy audio. Moreover, cloud services usually expose the ASR models as a black box, making it impossible to further fine-tune them.

ASR Error Detection (AED) models are designed to post-process the ASR output, in order to detect transcription errors and avoid their propagation to downstream tasks (Errattahi et al., 2018). AED models are widely used in interactive systems, to engage the user to resolve the detected errors. For example, AED systems can be found in *Google Docs Voice Typing*, where low confidence words are underlined, making it easier for users to spot errors and take actions to correct them.

Modern NLP models usually build upon the Transformer architecture (Vaswani et al., 2017). However, no Transformer-based AED models have been proposed yet. Recently, the Transformer has been applied to ASR *error correction* (Mani et al., 2020; Liao et al., 2020; Leng et al., 2021a,b), another ASR post-processing task. These models use only the transcription hypothesis text as input and discard other signals from the ASR model. However, earlier work on AED (not Transformer-based) has shown the benefits of such signals (Allauzen, 2007; Pellegrini and Trancoso, 2009; Chen et al., 2013) and specifically the benefits of ASR word-level confidence scores (Zhou et al., 2005), which are often provided in addition to the transcribed

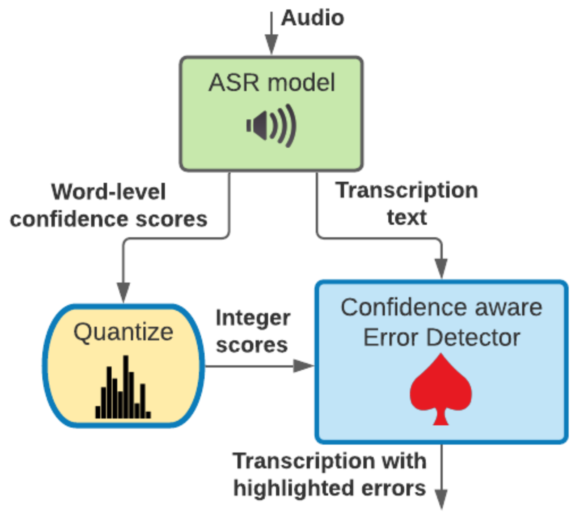

Figure 1: Our AED pipeline. The confidence scores are quantized and jointly encoded with the transcription text into a contextualized representation.

* Equal contribution.
[1]Our code and data are available at https://github.com/google-research/google-research/tree/master/red-ace.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 2800–2808
December 7-11, 2022 ©2022 Association for Computational Linguistics

text (Jiang, 2005; Qiu et al., 2021; Li et al., 2021).

In this work we focus exclusively on AED and propose a natural way to embed the ASR confidence scores into the Transformer architecture. We introduce ♠ RED-ACE, a modified Transformer encoder with an additional embedding layer, that jointly encodes the textual input and the word-level confidence scores into a contextualized representation (Figure 2). Our AED pipeline first quantizes the confidence scores into integers and then feeds the quantized scores with the transcribed text into the modified Transformer encoder (Figure 1). Our experiments demonstrate the effectiveness of RED-ACE in improving AED performance. In addition, we demonstrate the robustness of RED-ACE to changes in the transcribed audio quality. Finally, we release a novel dataset that can be used to train and evaluate AED models.

2 ♠ RED-ACE

Following recent trends in NLP, we use a pre-trained Transformer-based language model, leveraging its rich language representation. RED-ACE is based on a pre-trained BERT (Devlin et al., 2019), adapted to be confidence-aware and further fine-tuned for sequence tagging. Concretely, our AED model is a binary sequence tagger that given the ASR output, consisting of the transcription hypothesis words and their corresponding word-level confidence scores, predicts an ERROR or NOTERROR tag for each input token.[2]

Our AED pipeline is illustrated in Figure 1. First, we quantize the floating-point confidence scores into integers using a binning algorithm.[3] Next, the quantized scores and the transcription text are fed into a confidence-aware BERT tagger.

In BERT, each input token has 3 embeddings: token, segment and position.[4] To adapt BERT to be confidence-aware, we implement an additional dedicated embedding layer, indicating the confidence bin that the input token belongs to. We construct a learned confidence embedding lookup matrix $M \in \mathbb{R}^{B \times H}$, where B is the number of confidence bins and H is BERT's embedding vector's size. For a given token, its input representation is constructed by summing the corresponding BERT's embeddings with its confidence embedding (Figure 2). This allows the model to learn a dedicated dense representation vector for each confidence bin, as well as naturally combine it with the final contextualized representation of each input token.

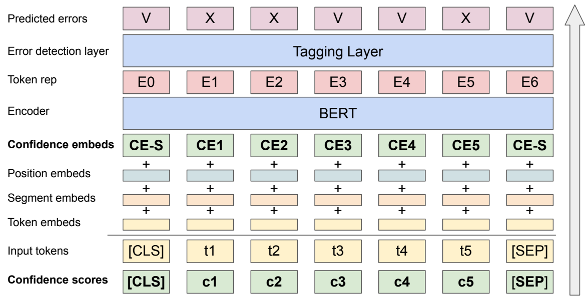

Figure 2: Our confidence-aware AED model. We use a BERT-based tagger with modifications colored in green. An additional embedding layer is added to represent the embedding of the quantized confidence scores.

ASR Model	Pool	Split	# Examples	# Words	# Errors
default	*clean*	Train	103,895	3,574,027	357,145 (10.0%)
		Dev	2,697	54,062	5,111 (9.5%)
		Test	2,615	52,235	4,934 (9.4%)
	other	Train	146,550	4,650,779	770,553 (16.6%)
		Dev	2,809	48,389	9,876 (20.4%)
		Test	2,925	50,730	10,317 (20.3%)
video	*clean*	Train	104,013	3,589,136	210,324 (5.9%)
		Dev	2,703	54,357	3,109 (5.7%)
		Test	2,620	52,557	2,963 (5.6%)
	other	Train	148,678	4,810,226	148,678 (7.9%)
		Dev	2,809	50,983	5,901 (11.6%)
		Test	2,939	52,192	6,033 (11.6%)

Table 1: AED dataset statistics.

3 Dataset Creation and Annotation

To train and evaluate AED models, we generate a dataset with labeled transcription errors.

Labeling of ASR Errors. We decode audio data using an ASR model and obtain the transcription hypothesis. Then, we align the hypothesis words with the reference (correct) transcription. Specifically, we find an edit path, between the hypothesis and the reference, with the minimum edit distance and obtain a sequence of edit operations (insertions, deletions and substitutions) that can be used to transform the hypothesis into the reference. Every incorrect hypothesis word (i.e needs to be deleted or substituted) is labeled as ERROR, the rest are labeled as NOTERROR.

Audio Data Source. We use the LibriSpeech corpus (Panayotov et al., 2015), containing 1000 hours of transcribed English speech from audio books.[5]

[2] We discuss words to tokens conversion in §A.1.

[3] Typical confidence scores range between 0.0 to 1.0. We perform experiments with simple equal-width binning and quantile-based discretization (equal-sized buckets), as well as different bin numbers. More details in §A.1.

[4] We refer the reader to Devlin et al. (2019) for more details.

[5] https://www.openslr.org/12/

The corpus contains *clean* and *other* pools, where *clean* is of higher recording quality.[6]

ASR Models. In this work we focus exclusively on a black-box setting, where the exact implementation details of the ASR and the confidence models are unknown. This setting is particularly relevant since many applications rely on strong performance of black-box ASR models which are exposed as cloud services. We use Google Cloud Speech-to-Text API as our candidate ASR model.[7] In our main experiments we select the *default* ASR model.[8] To ensure the generalization ability of RED-ACE, we repeat our main experiments using a different ASR model, in this case we choose the *video* model. Table 1 presents the statistics of our dataset. It is notable that the main model's error rate (*default*) is about twice as high as the additional model's error rate (*video*), which shows that (even though both models are part of the Google Cloud API) this additional ASR model is substantially different from the main ASR model we used.

Data Release. Since Google Cloud requires a paid subscription and since the underlying ASR models may change over time, we make our dataset publicly available.[9] This ensures full reproducibility of our results (in case the ASR models change) and makes it easier for researchers to train and evaluate AED models, removing the need to run inference on paid cloud-based ASR models or train dedicated models in order to transcribe audio.

4 Experimental Setup

Our experiments examine the informativeness of the confidence scores as well as the effectiveness of RED-ACE in combining them with text. We provide extensive implementation details in §A.1.

4.1 Baselines

C-O (Confidence Only) Uses the word-level scores from the ASR confidence model directly. Predicts ERROR if the token's confidence score is below a threshold.[10]

BERT-MLM Masks out the input words one at a time and uses a pre-trained BERT (Devlin et al., 2019) as a Masked Language Model (MLM) in order to infill them. Predicts ERROR for input words that are not in the top k BERT's suggestions.[9]

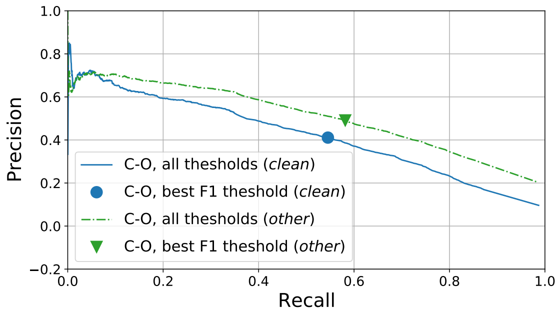

Figure 3: Threshold tuning process for the C-O baseline. Models are evaluated using different confidence scores thresholds and the threshold that yields the best F1 is chosen. A similar process is performed for BERT & C and BERT | C. For BERT-MLM we tune the values for k.

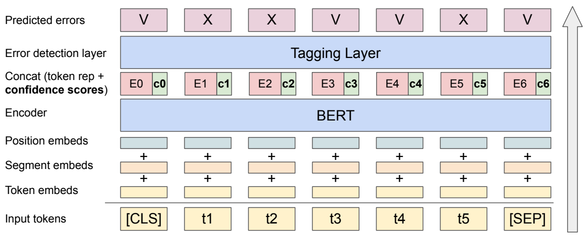

Figure 4: The BERT$_C$ baseline, which modifies the input to the tagger, unlike RED-ACE which modifies BERT's embeddings. The value of the respective confidence score is appended to the final contextualized representation.

BERT We fine-tune BERT (Devlin et al., 2019) for sequence tagging (only on text, *without* adding RED-ACE). As Transformers have not beeen applied for AED yet, we choose BERT as a pre-trained LM following Cheng and Duan (2020), who applied it for Grammatical Error Detection (GED) and achieved the highest performance in the NLPTEA-2020 Shared Task (Rao et al., 2020).

BERT & C Predicts ERROR if BERT predicts ERROR **and** confidence is below a threshold.[9]

BERT | C Predicts ERROR if BERT predicts ERROR **or** confidence is below a threshold.[9]

BERT$_C$ We fine-tune BERT *jointly* with the confidence scores by concatenating the score value to the token's contextualized representation produced by BERT (directly before it is fed into the sequence tagger). BERT's last hidden layer dimension is increased by 1, and the corresponding value populated with the token's confidence score. An illustration is provided in Figure 4.

[6] We provide additional details about the corpus in §A.2.
[7] https://cloud.google.com/speech-to-text
[8] https://cloud.google.com/speech-to-text/docs/basics#select-model
[9] Additional details about the dataset are provided in §A.2.
[10] We choose the confidence threshold or k value (in case of BERT-MLM) with the best F1 on the dev set (Figure 3).

	clean			*other*		
	R	P	F1	R	P	F1
C-O	52.1	42.5	46.8	63.5	45.6	53.1
BERT-MLM	58.0	26.5	36.4	**72.7**	35.9	48.1
BERT	58.5	77.6	66.7	58.0	77.1	66.2
BERT & C	55.8	75.0	64.0	55.5	75.5	64.0
BERT ∣ C	**63.3**	68.1	65.6	68.1	67.1	67.6
BERT_C	51.7	78.9	66.3	58.1	78.8	66.9
RED-ACE	61.1	**81.9***	**70.0***	64.1	**79.9***	**71.1***
F1 Δ%			+4.9%			+7.4%

Table 2: Main settings using the errors from the *default* ASR model (see Table 1). R and P stands for Recall and Precision. F1 $\Delta\%$ compares RED-ACE to the BERT baseline. Results with $*$ indicate a statistically significant improvement compared to the strongest baseline.

	other → *clean*			*clean* → *other*		
	R	P	F1	R	P	F1
BERT	64.3	71.9	67.9	47.1	80.3	59.4
RED-ACE	**67.9***	**77.0***	**72.2***	**53.7***	**83.3***	**65.3***
F1 Δ%			+6.3%			+9.9%

Table 3: Robustness settings with the *default* ASR model (Table 1). *other* $\rightarrow$ *clean* means train on *other* and eval on *clean*. Format is similar to Table 2.

4.2 Evaluation

Main Settings. In the main settings we train the models on the *clean* and *other* training sets and evaluate them *clean* and *other* test sets respectively.

Robustness Settings. A real-word AED system should remain effective when the audio stems from different recording qualities. Changes in recording quality, can affect the ASR model's errors distribution and thus can potentially reduce the effectiveness of the AED model. As our dataset contains 2 pools with different recording quality (Table 1), we can measure whether RED-ACE's performance deteriorates when the audio quality of the training data changes. To this end we define the *robustness settings* (Table 3), where we perform a cross-pools evaluation, evaluating models that were trained on *clean* and *other* training sets using the *other* and the *clean* test sets respectively.

Metric. We measure errors detection *Precision (P)*, *Recall (R)* and *F1*. *Recall* measures the percent of real errors that were detected, while *Precision* measures the percent of the real errors out of all detected errors. We calculate the *P* and *R* on the word-level. We also report span-level results for the main settings in Table 9 in the appendix.

5 Results

Table 2 presents our main results, evaluating the models on the *main settings* using errors from the main (*default*) ASR model. Table 3 presents the results on the *robustness settings*, also using errors from the main ASR model.

The low F1 of C-O suggest that the ASR confidence has low effectiveness without textual signal. The low F1 of BERT-MLM indicates that supervised training on real transcription errors is crucial.

We next observe that BERT & C performs worse than BERT on all metrics. When comparing BERT ∣ C to BERT we observe the expected increase in recall (BERT's errors are a subset of the errors from BERT ∣ C) and a decrease in precision, F1 decreases on *clean* and increases on *other*. The results on BERT_C are particularly surprising. Similarly to RED-ACE, BERT_C trains BERT *jointly* with the scores. However, unlike RED-ACE, BERT_C performs worse than BERT. This demonstrates the effectiveness and importance of our modeling approach, that represents the scores using a learned dense embedding vectors. As RED-ACE is the only method that successfully combines the scores with text, we focus the rest of the analysis on comparing it to the text-based BERT tagger.

In the *main settings* (Table 2), RED-ACE consistently outperforms BERT on all evaluation metrics in both pools. This demonstrates the usefulness of the confidence signal on top of the textual input, as well as the effectiveness of RED-ACE in combining those signals. RED-ACE's F1 $\Delta\%$ on *clean* is lower than on *other*. This can be attributed to the fact that the error rate in *clean* is twice lower than in *other* (Table 1), which means that the model is exposed to fewer errors during training.

Finally, we analyze the *robustness settings* from Table 3. We first note that RED-ACE outperforms BERT in both settings, indicating its robustness across different settings, and that it can remain effective with recording quality differences between train and test time. When observing the performance on the *clean* test set, we observe that training AED models on *other* instead of *clean*, leads to improvement in F1. This can be attributed to the higher error rate and larger number of training examples in *other* (see Table 1), which exposes the models to larger amount of errors during training. The F1 $\Delta\%$ on *other* $\rightarrow$ *clean* (Table 3) is comparable to *clean* (Table 2), with a statistically insignificant improvement. An opposite trend can be seen on the *other* test set. The performance of models that were trained on *clean* instead of *other* deteriorates. Yet, RED-ACE's relative per-

	clean			*other*		
	R	P	F1	R	P	F1
BERT	54.9	**77.2**	64.2	52.7	78.8	63.2
RED-ACE	**58.6***	75.4	**65.9***	**55.2***	**80.7***	**65.6***
F1 Δ%			+2.6%			+3.8%

Table 4: Main settings using the errors from the *video* ASR model. Format is similar to Table 2.

	other → *clean*			*clean* → *other*		
	R	P	F1	R	P	F1
BERT	61.2	73.5	66.8	42.9	**82.2**	56.4
RED-ACE	**62.8***	**75.8***	**68.7***	**47.7***	79.8*	**59.7***
F1 Δ%			+2.8%			+5.9%

Table 5: Robustness settings using the errors from the *video* ASR model. Format is similar to Table 3.

formance drop is smaller than BERT's. RED-ACE drops by 8.2% (from 71.1 to 65.3) while BERT by 10.3% (from 66.2 to 59.4). This is also demonstrated by the statistically significant increase in F1 Δ%, from 7.4% in *other* → *other* to 9.9% in *clean* → *other*. This serves as additional evidence for the robustness of RED-ACE. We also note that *clean* → *other* is the most challenging setting, with BERT's F1 significantly lower than the other 3 settings, meaning that RED-ACE shows the largest improvement (F1 Δ%) in the hardest setting.

Generalization Across ASR Models. As discussed in §3, to ensure that RED-ACE is applicable to not only one specific ASR model, we repeat our experiments using a different ASR model. The results are presented in Table 4 and Table 5. RED-ACE outperforms BERT in all settings, with statistically significant F1 improvements, further highlighting RED-ACE robustness.

6 Related Work

ASR Confidence Scores are used to evaluate reliability of recognition results (Jiang, 2005). In modern ASR models, a separate confidence network is usually trained using a held-out dataset (Qiu et al., 2021; Wessel et al., 2001).

Uncertainty Calibration adapts a models prediction probabilities to better reflect their true correctness likelihood (Guo et al., 2017). We provide the Brier Scores (evaluating calibration) for our dataset in Table 8. AED models, which perform a binary classification - ERROR or NOTERROR, do not explicitly use calibration. For example in C-O, BERT | C and BERT & C we tune the threshold to an optimal value, and since most calibration techniques will preserve the relative scores ordering, better calibration will not improve performance. BERT_C and RED-ACE do not rely on calibrated scores, since deep neural networks can model non linear relationships (Hornik et al., 1989).

AED. We provide a brief summary of relevant AED papers, for a more thorough review of AED we refer the reader to Errattahi et al. (2018). Zhou et al. (2005) used data mining models, leveraging features from confidence scores and a linguistics parser. Allauzen (2007) used logistic regression with features extracted from confusion networks. Pellegrini and Trancoso (2009) used a Markov Chains classifier. Chen et al. (2013) focused on spoken translation using confidence from a machine translation model, posteriors from entity detector and a word boundary detector.

Modern Transformer-based approaches have not addressed the AED task directly. A few attempts were made to apply Transformers for ASR *error correction*, using a sequence-to-sequence models to map directly between the ASR hypothesis and the correct (reference) transcription (Mani et al., 2020; Liao et al., 2020; Leng et al., 2021a,b). To the best of our knowledge, our work is the first to address AED using the Transformer and to introduce representation for ASR confidence scores in a Transformer-based ASR post-processing model.

7 Conclusion

We introduced ♠ RED-ACE, an approach for embedding ASR word-level confidence scores into a Transformer-based ASR error detector. RED-ACE jointly encodes the scores and the transcription hypothesis into a contextualized representation.

Our experiments demonstrated that the ASR word-level confidence scores are useful on top of the transcription hypothesis text, yet it is not trivial to effectivelly combine these signals. We showed that performing such combination using RED-ACE leads to significant performance gains, as well as increased robustness to changes in the audio quality, which can be crucial for real-world applications.

In addition, we published a novel AED dataset that allows researchers to train and evaluate AED models, without the need to run ASR models. It also ensures the full reproducibility of our results in case Google Cloud models will change over time.

In future work, we would like to leverage additional signals from the ASR model (such as alternative hypotheses), as well as explore the benefits of confidence scores for *error correction* models.

8 Limitations

Limitations Our approach does not account for ASR errors where the ASR system simply deletes output words. However, it is not clear whether those cases are of a practical use for an AED application that highlights incorrect words in the hypothesis, as in this case there is nothing to highlight. More specifically, our approach does not consider *isolated* deletions.

To illustrate that, let's first consider an example in which 2 words were transcribed as 1 word, meaning that 1 word was omitted in the transcription. For example, if *"a very big cat"* was transcribed as *"a small cat"*. An AED application would ideally highlight the word *"small"* as a transcription error. This case is actually covered by our approach, even though one word is omitted in the transcription, because when creating the AED dataset we will label "small" as an error and train the model accordingly (details in section 3).

The cases that are not covered are when the ASR model omits words while all the surrounding words are transcribed correctly. For example *"a very big cat"* that was transcribed as *"a big cat"*. In this case, all the words in the transcription hypothesis are correct words and our approach is not expected to discover any error. We chose not to cover those cases as it is not clear if they are useful for an error detection application, that usually needs to highlight incorrect words in the hypothesis. In addition, ignoring those cases is also in-line with previous work (Zhou et al., 2005). Finally, our analysis showed that those cases are extremely rare, for example in *clean* they occur only in 0.37% of the words.

Risks A possible risk posed by an AED system could be caused by an over-reliance on it. Whereas without AED, the entire output of an ASR system may have been manually verified, with AED only parts of output which the AED flagged may be verified, leading to errors remaining that were not found by the AED system.

Acknowledgements

We thank the reviewers for their valuable suggestions for improving this work. We would also like to thank Gal Elidan, Idan Szpektor, Eric Malmi, Yochai Blau, Amir Feder, Andrew Rosenberg, Françoise Beaufays and Avinatan Hassidim for reviewing the paper and providing useful feedback.

References

Alexandre Allauzen. 2007. Error detection in confusion network. In *INTERSPEECH 2007, 8th Annual Conference of the International Speech Communication Association, Antwerp, Belgium, August 27-31, 2007*, pages 1749–1752. ISCA.

Wei Chen, Sankaranarayanan Ananthakrishnan, Rohit Kumar, Rohit Prasad, and Prem Natarajan. 2013. ASR error detection in a conversational spoken language translation system. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 7418–7422. IEEE.

Yong Cheng and Mofan Duan. 2020. Chinese grammatical error detection based on BERT model. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 108–113, Suzhou, China. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Rahhal Errattahi, Asmaa El Hannani, and Hassan Ouahmane. 2018. Automatic speech recognition errors detection and correction: A review. *Procedia Computer Science*, 128:32–37. 1st International Conference on Natural Language and Speech Processing.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR.

Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hui Jiang. 2005. Confidence measures for speech recognition: A survey. *Speech Commun.*, 45(4):455–470.

Kartik Khandelwal, Preethi Jyothi, Abhijeet Awasthi, and Sunita Sarawagi. 2020. Black-box adaptation of ASR for accented speech. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 1281–1285. ISCA.

Yichong Leng, Xu Tan, Rui Wang, Linchen Zhu, Jin Xu, Wenjie Liu, Linquan Liu, Xiang-Yang Li, Tao Qin, Edward Lin, and Tie-Yan Liu. 2021a. Fastcorrect 2: Fast error correction on multiple candidates for automatic speech recognition. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 4328–4337. Association for Computational Linguistics.

Yichong Leng, Xu Tan, Linchen Zhu, Jin Xu, Renqian Luo, Linquan Liu, Tao Qin, Xiang-Yang Li, Ed Lin, and Tie-Yan Liu. 2021b. Fastcorrect: Fast error correction with edit alignment for automatic speech recognition. *CoRR*, abs/2105.03842.

Qiujia Li, David Qiu, Yu Zhang, Bo Li, Yanzhang He, Philip C. Woodland, Liangliang Cao, and Trevor Strohman. 2021. Confidence estimation for attention-based sequence-to-sequence models for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 6388–6392. IEEE.

Junwei Liao, Sefik Emre Eskimez, Liyang Lu, Yu Shi, Ming Gong, Linjun Shou, Hong Qu, and Michael Zeng. 2020. Improving readability for automatic speech recognition transcription. *CoRR*, abs/2004.04438.

Anirudh Mani, Shruti Palaskar, Nimshi Venkat Meripo, Sandeep Konam, and Florian Metze. 2020. ASR error correction and domain adaptation using machine translation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 6344–6348. IEEE.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, South Brisbane, Queensland, Australia, April 19-24, 2015*, pages 5206–5210. IEEE.

Thomas Pellegrini and Isabel Trancoso. 2009. Error detection in broadcast news ASR using markov chains. In *Human Language Technology. Challenges for Computer Science and Linguistics - 4th Language and Technology Conference, LTC 2009, Poznan, Poland, November 6-8, 2009, Revised Selected Papers*, volume 6562 of *Lecture Notes in Computer Science*, pages 59–69. Springer.

David Qiu, Qiujia Li, Yanzhang He, Yu Zhang, Bo Li, Liangliang Cao, Rohit Prabhavalkar, Deepti Bhatia, Wei Li, Ke Hu, Tara N. Sainath, and Ian McGraw. 2021. Learning word-level confidence for subword end-to-end ASR. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 6393–6397. IEEE.

Gaoqi Rao, Erhong Yang, and Baolin Zhang. 2020. Overview of nlptea-2020 shared task for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 25–35.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Frank Wessel, Ralf Schlüter, Klaus Macherey, and Hermann Ney. 2001. Confidence measures for large vocabulary continuous speech recognition. *IEEE Trans. Speech Audio Process.*, 9(3):288–298.

Lina Zhou, Yongmei Shi, Jinjuan Feng, and Andrew Sears. 2005. Data mining for detecting errors in dictation speech recognition. *IEEE Trans. Speech Audio Process.*, 13(5-1):681–688.

A Appendix

A.1 Implementation Details

Training. We fine-tune our BERT-based (Devlin et al., 2019) model with a batch size of 512[11], a weight decay of 0.01, and a learning rate of 3e-5[12]. The maximum input length is set to 128 tokens. We pad shorter sequences and truncate longer ones to the maximum input length. We use the cross-entropy loss function, optimizing the parameters with the AdamW optimizer. We train for a maximum of 500 epochs and choose the checkpoint with the maximum tagging accuracy on the development set.[13] The best checkpoint was found at epochs 100-150 after approximately 8 hours of training time. All models were trained on TPUs (4x4). BERT-base has 110 million parameters, the inclusion of confidences embeddings for RED-ACE added 10k additional parameters. The confidence embedding matrix is randomly initialized with truncated normal distribution[14].

If a single word is split into several tokens during BERT's tokenization, all the corresponding tokens get the confidence score of the original word. To predict word-level errors (used throughout the paper), we treat a word as an error if one of its tokens was tagged as error by the model. To predict span-level errors (reported for completeness in Table 9), we treat every sequence of errors as one error-span and every sequence of correct words as a correct-span.

Binning. Table 6 presents results for different binning algorithms and bin sizes. For binning algorithms we use: (1) simple equal-width binning and (2) quantile-based discretization (equal-sized buckets). We note that there is no significant difference between the results. In our main experiments we used equal width binning with 10 bins. For special tokens,[15] that do not have confidence scores, we chose to allocate a dedicated bin.

Statistics Significance Test. In table 2, in addition to the main results, we provide a statistic significance tests results. For this purpose we pseudo-randomly shuffle all words in our test set, split them up into 100 approximately equally sized subsets, and compute recall, precision and F1 for each of them for the baseline and RED-ACE models. We then apply the Student's paired t-test with $p < 0.05$ to these sets of metrics. To determine statistical significance in F1 $\Delta\%$ between different setups evaluated on the same data set, F1 $\Delta\%$ is computed for each of the given subsets, and the same significance test is applied to the resulting sets of F1 $\Delta\%$ between two setups.

Binning algorithm	# Bins	R	P	F1
Equal width bins	10	**64.1**	79.9	**71.1**
	100	62.5	80.5	70.4
	1000	63.2	80.7	70.9
Equal size bins	10	63.0	**81.5**	**71.1**

Table 6: Effect on different binning strategies (*other*).

Pool	Subset Name	Audio Hours	# Examples
Clean	*train-clean-100*	100.6	28,539
	train-clean-360	363.6	104,014
	dev-clean	5.4	2,703
	test-clean	5.4	2,620
Other	*train-other-500*	496.7	148,688
	dev-other	5.3	2,864
	test-other	5.1	2,939

Table 7: LibriSpeech corpus subsets statistics.

A.2 Published AED Dataset

As described in §3, we generate our own AED dataset. To this end we transcribe the LibriSpeech corpus using 2 modes from Google Cloud Speech-to-Text API.[16] We choose the *default* model as our main model and the *video* model as the additional model[17]. We also enable the word-level confidence in the API.[18] Our submission includes the AED dataset as well as the predictions of our models on the test sets. We hope that our dataset will help future researchers and encourage them to work on AED.

The LibriSpeech Corpus Details. We provide here additional details abut the LibriSpeech corpus.[19] The corpus contains approximately 1000 hours of English speech from read audio books.

[11] We choose the best among 128, 512 and 1024, based on tagging accuracy on the development set.

[12] We choose the best among 5e-5, 4e-5, 3e-5, and 2e-5, based on tagging accuracy on the development set.

[13] For RED-ACE the tagging accuracy was 95.4 on *clean* and 89.7 on *other*.

[14] https://www.tensorflow.org/api_docs/python/tf/keras/initializers/TruncatedNormal

[15] [CLS] and [SEP] in case of BERT.

[16] https://cloud.google.com/speech-to-text

[17] https://cloud.google.com/speech-to-text/docs/basics#select-model

[18] https://cloud.google.com/speech-to-text/docs/word-confidence#word-level_confidence

[19] https://www.openslr.org/12/

```
{
  "id": "test-other/2414/128292/2414-128292-0002",
  "truth": "what matter about my shadow",
  "asr": [
    ["foot", 0.5593389272689819, 1],
    ["doctor", 0.9715939164161682, 1],
    ["about", 0.9719187617301941, 0],
    ["my", 0.8484553694725037, 0],
    ["shadow", 0.9790922999382019, 0]
  ]
}
```


Figure 5: A single example from our AED dataset.

ASR Model	Pool	Brier Score
default	*clean*	0.069
	other	0.142
video	*clean*	0.06
	other	0.1

Table 8: Brier Scores (evaluating confidence scores calibration, lower is better) for our dataset.

The corpus contains *clean* and *other* pools. The training data is split into three subsets: *train-clean-100*, *train-clean-360* and *train-other-500*, with approximate sizes of 100, 360 and 500 hours respectively. Each pool contains also a development and test sets with approximately 5 hours of audio. Full data split details can be seen in table 7. We note that the #Examples is slightly different than the numbers in our dataset (see table 1). When transcribing with Google Cloud API, we occasionally reached a quota limit and a negligible number of examples was not transcribed successfully (up to 2% per split). The *clean* pool contains 2 training sets, we used the larger one in our dataset (*train-clean-360*).

Annotation Description. A single example from our AED dataset can be seen is fig. 5. The annotation contains the ASR hypothesis words, the corresponding word-level confidence scores and the ERROR or NOTERROR label.

License. This data as well as the underlying LibrSpeech ASR corpus are licensed under a Creative Commons Attribution 4.0 International License[20].

	clean R	*clean* P	*clean* F1	*other* R	*other* P	*other* F1
C-O	31.0	27.5	29.1	23.4	20.2	21.7
BERT-MLM	27.4	12.3	17.0	22.2	11.6	15.2
BERT	47.1	59.6	52.6	37.1	46.6	41.3
BERT & C	44.5	55.5	49.4	35.5	43.7	39.2
BERT | C	48.9	51.0	49.9	40.6	40.2	40.4
BERT_C	45.4	59.9	51.7	37.5	48.0	42.1
RED-ACE	**49.4**	**63.6***	**55.6***	**42.1***	**50.9***	**46.1***
F1 Δ%			+5.4%			+9.5%

Table 9: Span-level results for the main settings using the errors from the *default* ASR model. The format is similar to Table 2.

[20] http://creativecommons.org/licenses/by/4.0/