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

Anonymous ACL submission

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 propose to 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 model as a black box, making it impossible to further fine-tune it.

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).

[1]The code will be released upon publication.

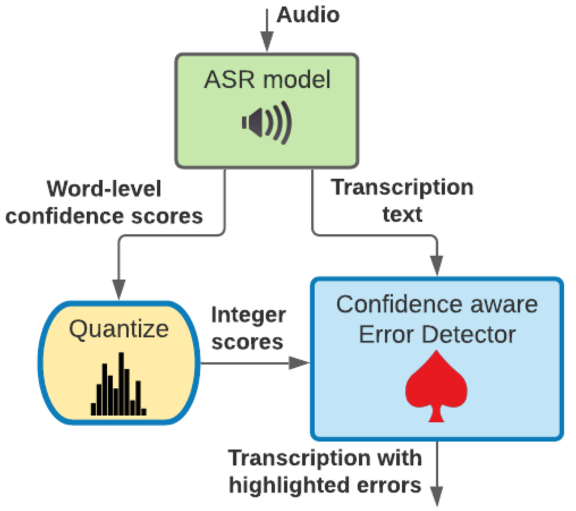

Figure 1: Our ASR Error Detection pipeline. The word-level confidence scores are quantized and jointly encoded with the transcription text. The resulting contextualized representation is fed into a sequence tagger.

AED models are widely used in interactive systems, to engage the user to resolve the detected errors. One example of an AED system 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 ASR structured 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 text (Jiang, 2005; 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 (fig. 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 (fig. 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. Our AED model 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.

An overview of our AED pipeline can be seen in fig. 1. Given the ASR output, we first quantize the floating-point confidence scores into integers using a binning algorithm.[2] The binning algorithm and the number of bins are hyper-parameters of our algorithm.[3]

The quantized scores and the transcription text are fed into our confidence-aware BERT (fig. 2). In BERT, each input token has 3 different embeddings.[4] To adapt BERT to be confidence-aware, we add additional embedding to every input token, indicating the confidence bin it belongs to. We construct a learned confidence embedding lookup matrix $M \in \mathbb{R}^{B \times H}$, where B is the number of bins and H is BERT's embedding vector size. For a given token, its input representation is constructed by summing the corresponding BERT's embeddings with its confidence embedding.

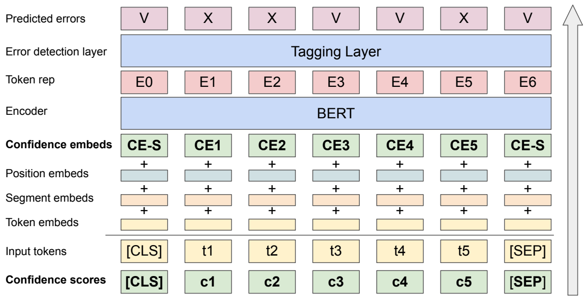

Figure 2: Our confidence-aware AED model. We use a BERT-based tagger with modifications colored in green. An additional embedding layer is added to represent the embedding of the quantized confidence scores.

Pool	Split	# Examples	# Words	# Errors
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%)

Table 1: AED dataset statistics.

3 Dataset Creation and Annotation

To train and evaluate our model, we generate a dataset with labeled transcription errors. First, we decode audio data using the candidate 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 and the rest are labeled as NOTERROR.

For the ASR model, we use Google Cloud Speech-to-Text API[5] (more details in §A.2). For an audio data source, we use the LibriSpeech corpus (Panayotov et al., 2015), containing approximately 1000 hours of transcribed English speech from audio books.[6] The corpus contains *clean* and *other* pools, where *clean* is of higher recording quality. Table 1 contains our generated dataset statistics. To encourage further research we make our dataset

[2]Typical confidence scores range between 0.0 to 1.0.

[3]We experiment with different binning strategies, see §A.1.

[4]Token, Segment and Position embeddings. See fig. 2 in Devlin et al. (2019).

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

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

	Main setups						Robustness setups					
	Train *clean* -> Eval *clean*			Train *other* -> Eval *other*			Train *other* -> Eval *clean*			Train *clean* -> Eval *other*		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
C-O	52.1	42.5	46.8	63.5	45.6	53.1	63.6	34.7	44.9	52.3	52.3	52.3
BERT	58.5	77.6	66.7	58.0	77.1	66.2	64.3	71.9	67.9	47.1	80.3	59.4
RED-ACE	**61.1***	**81.9***	**70.0***	**64.1**	**79.9***	**71.1***	**67.9***	**77.0***	**72.2***	**53.7***	**83.3***	**65.3***
F1 Δ%			+4.9%			+7.4%			+6.3%			+9.9%

Table 2: AED results. R and P stands for Recall and Precision. F1 Δ% compares RED-ACE to the strongest baseline. RED-ACE results with $*$ indicate a statistically significant difference compared to the strongest baseline.

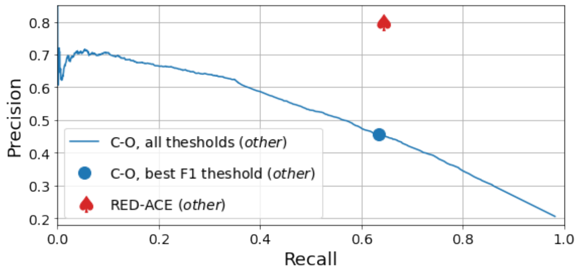

Figure 3: Precision/Recall values that can be obtained by using thresholds on confidence to detect errors.

publicly available. For additional details about the corpus and our generated dataset see §A.4.

4 Experimental Setup

As described in §2, we use pre-trained BERT (Devlin et al., 2019) and adapt it to be confidence-aware by introducing RED-ACE. We then fine-tune it for sequence tagging using the annotated transcription errors from our dataset (§3). We provide extensive implementation details in §A.1.

4.1 Baselines

To evaluate the complementary effect of the textual and the confidence signals, as well as the effectiveness of RED-ACE in combining those signals, we compare RED-ACE to the following baselines:

Confidence Only (C-O) As the primary purpose of the ASR confidence scores is to allow downstream applications to detect transcription errors, our first baseline is based on confidence only. We use a score threshold to predict errors, meaning that tokens with scores below the threshold are classified as ERROR. We choose the threshold that yields the best F1 on the development set (see fig. 3).

BERT We fine-tune BERT (Devlin et al., 2019) for sequence tagging (without RED-ACE), using the annotated transcription errors from our dataset (§3). This baseline is based on the Grammatical Error Detection (GED) model proposed by Cheng and Duan (2020), where BERT based taggers achieved the highest performance in the NLPTEA-2020 Shared Task for Chinese GED (Rao et al., 2020). We used a GED model as we could not find any modern AED models (see §6). In addition, leveraging a Transformer that uses only the ASR hypothesis text as input, is also in line with recent work on ASR *error correction*, an additional ASR post-processing task (§6).

4.2 Evaluation

We measure *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.

Robustness A real-word transcription system should be robust to changes in the audio quality. Such changes can affect the ASR model's errors distribution and thus can potentially reduce the effectiveness of the AED model. Luckily, our dataset contains 2 pools with different audio quality (§3), allowing us to evaluate RED-ACE's robustness. We perform a cross-pools evaluation, evaluating models that were trained on *clean* and *other* pools using the *other* and the *clean* test sets respectively.

5 Results

Table 2 contains the main results, comparing RED-ACE to the baseline models (§4.1). When observing the F1 results for C-O, the advantage of the models that use textual input is evident. Thus, we focus our analysis on comparing RED-ACE to the text-based BERT tagger.

We first analyze the *main setups* and observe that RED-ACE consistently outperforms BERT on all evaluation metrics in both pools. This demonstrates the usefulness of the confidence scores signal on top of the textual input, as well as the effectiveness of our approach in combining those signals. RED-ACE F1 Δ% drop a little on *clean*, compared to *other*. This is expected since errors in *clean* are rare (thus harder to detect), with an error rate twice lower than in *other* (see table 1), making it more challenging to further improve on top of the strong BERT baseline.

Next we analyze the *robustness setups*. When analyzing *other -> clean*, we observe that BERT and RED-ACE achieve higher recall and lower precision, compared to *clean -> clean*. This is probably caused by the higher error rate of *other* (table 1), which leads to a model with a higher tendency to mark words as errors. Interestingly, the overall F1 actually increases for both models, suggesting that even though *other* is more noisy, exposing the model to a larger number of errors is crucial. An opposite trend can be seen when comparing *clean -> other* to *other -> other*. In this case, the recall drops dramatically for both models, while the precision is improving. Overall F1 drops significantly, again demonstrating the importance of exposing the AED model to a larger amount of errors.

Finally, we examine RED-ACE's robustness by comparing the F1 Δ% between the *main* and *robustness* setups. In the *other -> clean* setup, RED-ACE achieves a relative F1 improvement comparable to *clean -> clean* (6.3% compared to 4.9%)[7], which indicates that RED-ACE effectiveness is robust to transcriptions from different audio quality. The results on *clean -> other* are even more impressive. RED-ACE improves the F1 by 9.9%, compared to 7.4% improvement on *other -> other*.[8] *clean -> other* is the hardest setup with BERT's F1 significantly lower than the rest 3 setups, meaning that RED-ACE shows the strongest improvement in the hardest setup. This is another strong indication of the robustness of RED-ACE.

Robustness to Candidate ASR In order to make sure that RED-ACE is applicable to not only one specific ASR model, we repeat our main experiments using a different ASR model.[9] The results can be seen in table 3. RED-ACE outperforms all baselines, which provides additional evidence for its robustness, this time to errors that stem from different ASR models.

6 Related Work

AED has been studied for many years, we refer the reader to Errattahi et al. (2018) for a thorough review. 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 scores 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 the Transformer for the *error correction* task. Some used autoregressive sequence-to-sequence models to map directly between the ASR hypothesis to the correct (reference) transcription (Mani et al., 2020; Liao et al., 2020), while others used non-autoregressive models (Leng et al., 2021a,b). To the best of our knowledge, our work is the first to address the AED task using the Transformer architecture and to introduce representation for ASR confidence scores in a Transformer-based ASR post-processing model.

	clean -> clean			*other -> other*		
	R	P	F1	R	P	F1
C-O	28.7	22.4	25.2	34.5	26.2	29.8
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 3: AED results on *main setups* using errors from a different ASR model. Format is similar to table 2.

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 showed significant performance gains when using RED-ACE, compared to using the transcription text or the confidence scores alone, indicating the effectiveness of RED-ACE in constructing richer representation for error detection. Our results also demonstrated the robustness of RED-ACE to changes in the audio quality, which can be crucial for real-world applications.

In future work, we would like to explore the benefits of ASR confidence scores for *error correction* models. We also hope that our work will inspire AED researchers to integrate RED-ACE in their models, in order to potentially benefit from its complementary effect.

[7] The difference in F1 Δ% is not statistically significant.

[8] The difference in F1 Δ% is statistically significant.

[9] Also using Google Cloud API, this time with *video* instead of *default* model, more details in §A.2.

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.

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.

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.

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, a weight decay of 0.01, and a learning rate of 3e-6. 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. The best checkpoint was found at epochs 100-150 after approximately 8 hours of training time. All models were trained on TPUs (4x4). The confidence embedding matrix is randomly initialized with truncated normal distribution.[10]. 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, we treat a word as an error if one of it's tokens was tagged as error by the model. Bert base has 110 million parameters, the inclusion of confidences embeddings for RED-ACE added 10k additional parameters.

Binning Table 4 contains 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,[11] 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.

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

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

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 4: Effect on different binning strategies (*other*).

Pool	Split	# Examples	# Words	# Errors
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 5: Our AED dataset statistics when using a different ASR model (*video* instead of *default*).

A.2 ASR Models

We use Google Cloud Speech-to-Text API as our candidate ASR model.[12] In our main experiments we select the *default* ASR model[13] and enable the word-level confidence.[14] In our experiment with additional ASR model (table 3) we selected the *video* model.

Additional details about the *video* model We use the *video* model to make sure RED-ACE is effective for multiple ASR models (as discussed in §5). For completeness we provide additional details about the setup with *video*. Table 5 contains the AED dataset statistics when using the *video* model instead of *default*. A notable difference from table 1 is a significantly lower error rate on both pools. In table 3 we reported results for *video* only on the *main setups*, for completeness we add here the results for the *robustness setups* as well. Table 6 contains full results on *video*, including the *robustness setups*.

A.3 C-O Plot for the *clean* Corpus

In fig. 3 we illustrate the possible precision and recall values when using Confidence Only (C-O) on the *other* pool. For completeness we provide

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

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

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

	Main setups						Robustness setups					
	Train *clean* -> Eval *clean*			Train *other* -> Eval *other*			Train *other* -> Eval *clean*			Train *clean* -> Eval *other*		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
C-O	28.7	22.4	25.2	34.5	26.2	29.8	35.4	18.3	24.1	27.4	30.5	28.9
BERT	54.9	**77.2**	64.2	52.7	78.8	63.2	61.2	73.5	66.8	42.9	**82.2**	56.4
RED-ACE	**58.6***	75.4	**65.9***	**55.2***	**80.7***	**65.6***	**62.8***	**75.8***	**68.7***	**47.7***	79.8*	**59.7***
F1 Δ%			+2.6%			+3.8%			+2.8%			+5.9%

Table 6: AED results using a different ASR model (*video* instead of *default*). Format is similar to table 2.

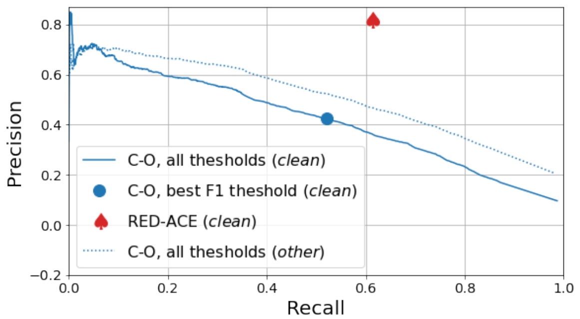

Figure 4: Comparison of RED-ACE to using the confidence scores alone on a threshold basis. Each threshold leads to a different precision recall balance. This plot is equivalent to fig. 3 but evaluated on the *clean* pool. In addition we added a dotted line representing the precision recall curve on the *clean* pool from fig. 3.

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.

the plot for the *clean* pool as well in fig. 4. We also added a dotted line representing the precision recall curve on the *other* pool from fig. 3. The higher precision recall values on the *other* pool are additional evidence that *clean* can be more challenging for error detection, due to lower error rate, as discussed in §5.

A.4 Published AED Dataset

As described in §3, we generate our own AED dataset. 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. In addition, while Google Cloud is a publicly available service, a paid subscription is required in order to transcribe significant amounts of data. Thus, we hope that our transcriptions will make AED more accessible. Finally, the underlying ASR model in Google Cloud can change over time, publishing the exact transcriptions that we obtained during our experiments, will ensure the full reproducibility of our results.


```
{
  "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.

The LibriSpeech Corpus Details We provide here additional details abut the LibriSpeech corpus.[15] The corpus contains approximately 1000 hours of English speech from read audio books. 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 an-

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

notation 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[16].

A.5 Limitations and Risks

Limitations Whilst we evaluated RED-ACE on multiple datasets with multiple ASR models, all experiments were run on English data. As such the benefits of RED-ACE on other languages has not been shown. Additionally, in this paper we focused on substitution and deletion errors of ASR systems, as such our approach does not account for ASR errors where the system simply deletes output 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.

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