Noisy Text Sequences Aggregation as a Summarization Subtask

Sergey Pletenev[1]

[1]National Research University Higher School of Economics (HSE University), Moscow, Russian Federation

Abstract

Most speech-driven systems on the first step convert audio to text through an automatic speech recognition (ASR) model and then pass the text to any downstream natural language processing (NLP) modules. However, these ASR models can lead to system failure or undesirable output when being exposed to natural language perturbation or variation in practice. In this paper, we introduce a simple yet efficient model for improving the understanding of the semantics of the input speeches and error correction by processing multi-hypothesis ASR systems.

Keywords

ASR n-best hypotheses integration, ASR, Seq2Seq, NLP, Spoken language understanding,

1. Introduction

Speech-enabled systems have found increasing use in recent years, especially in conversational dialogue and spoken language translation systems. Voice assistants such as Alexa (Amazon), Siri (Apple) or Alica (Yandex) are widely used in smartphones to retrieve information and control devices. Typically, a pipeline process is used to create a speech-enabled system. First, speech is converted into text by an automatic speech recognition (ASR) system. The text is then put into downstream modules to perform various tasks. Errors in the ASR system at the first stage spread to subsequent modules and reduce their performance. A small disturbance in the ASR system can corrupt the full pipeline. A simple but effective way to deal with ASR errors is to train follow-up tasks on samples containing ASR noise. The second way is to combine several audio transcriptions into one good transcription.

In this paper we propose a simple method to generate clean texts from a corpus containing ASR errors. This work is inspired by the success of Seq2Seq models on natural language generation tasks such as paraphrasing[1], text summarization[2] and translation[3]. We finetuned several Seq2Seq models on ASR text hypotheses to obtain a clean text.

Our contributions are two-fold as follows.

1. We propose an effective approach to ASR error correction by using Seq2Seq models.
2. We release the code of our model for future research.[1]

VLDB 2021 Crowd Science Workshop: Trust, Ethics, and Excellence in Crowdsourced Data Management at Scale, August 20, 2021, Copenhagen, Denmark

alex010rey@gmail.com (S. Pletenev)

[1]https://github.com/A1exRey/VLDB2021_workshop_t5

Table 1
Statistics of the datasets

Dataset	lines	WAcc	PAcc	Exact match(%)
VLDB 2021	67900	0.7703	0.88	29.7
DeBert	778	0.578	0.636	25.1
DSTC2/3	54648	0.517	0.706	8.1

2. Preprocessing

We experimented on the three datasets.

- **VLDB 2021**[4]: Dataset contains 9500 unique lines, 7 hypotheses for each example.
- **DSTC2/3** [5]: Dataset consists of human-computer dialogues in a restaurant domain collected with Amazon Mechanical Turk. It contains reference texts and ASR hypotheses. It has around 10 hypotheses for each text.
- **Stacked DeBert** [6]: Dataset generated by using freely available TTS(text-to-speech) and STT(speech-to-text) systems. From 6 to 7 hypothesis for each unique line.

All datasets are shown in the table 1 .

We use JiWER toolkit[2] to clean up our datasets and calculate WER (in this case WAcc)[7] metric for each line. WER is de facto standard metric for ASR system assessment. It is calculated by the total error count normalized by the reference length. In our work, we use an additional scoring metric called Phone Edit Rate (PER) [8] to evaluate the phoneme-level noisiness of the generated samples:

$$PER(ref, pred) = \frac{Levenshtein(Phone(ref), Phone(pred))}{Len(Phone(ref))} \tag{1}$$

$$PAcc(ref, pred) = 1 - PER(ref, pred) \tag{2}$$

Where ref is original text and $pred$ is text with ASR noise. $Phone$ is a function to transform text to phoneme. We use CMU Pronouncing Dictionary[3] to transform our texts.

The PER metric allows us to more accurately measure the accuracy of our models. Table 2 shows an example of estimating results using PER and WER metrics. We can see that in some cases the WER metric shows no change in quality: the last two rows show the same WER result, while PER shows the difference between these rows of text. In other cases WER metric shows worse result than it actually is. In the first two rows of table 2 the difference between the predicted result and the correct answer is one apostroph. WER shows one whole word error, while PER shows only one phoneme error, which is much more accurate.

[2]https://github.com/jitsi/jiwer/
[3]http://www.speech.cs.cmu.edu/cgi-bin/cmudict

Table 2
Example of texts in datasets. In each cell, the first line is text, the second line is the phoneme sequence of the text

text/phoneme	$WAcc$	$PAcc$
normally i'm the brutally honest one N AO R M AH L IY AY M DH AH B R UW T AH L IY AA N AH S T W AH N	1	1
normally am the brutally honest one N AO R M AH L IY AE M DH AH B R UW T AH L IY AA N AH S T W AH N	0.833	0.985
know me am the brittle honest one N OW M IY AE M DH AH B R IH T AH L AA N AH S T W AH N	0.333	0.794
newly individually honest one N UW L IY IH N D IH V IH JH UW AH L IY AA N AH S T W AH N	0.333	0.676

3. Methods

3.1. Models

In this work we use several models and baseline.

- **Baseline.** As a simple baseline we use majority vote: If some text occurs N times in a corpus, that text is considered correct, otherwise a random text is selected.
- **Advanced baseline.** For better baseline we use two algorithms: ROVER[9] and HRRASA[10].
- **T5.**[11] The T5 model is trained on several datasets for 18 different tasks which majorly fall into 8 categories: text summarization, question answering, translation etc. In experiments we use 3 different sizes: t5-small, t5-base, t5-large.
- **PEGASUS.**[12] PEGASUS model pretraining task is intentionally similar to summarization: important sentences are removed/masked from an input document and are generated together as one output sequence from the remaining sentences, similar to an extractive summary. We use PEGASUS model trained on Xsum dataset [13]

We use HuggingFace Transformers [4] for model training and prediction. Each model is trained with following parameteres: encoder length 512, decoder length 64, batch size 3, 8 epochs, learning rate 5e-05, after each 1000 steps we evaluate our models with beam size 12.

3.2. Data

We use a pipeline to clean-up and prepare our datasets:

1. Remove punctuation marks (except apostrophes) and numbers
2. Convert texts to lowercase
3. Remove unnecessary spaces in the sentence
4. Limit the number of hypotheses for each of the unique texts to 7
5. Concatenate hypotheses to single text with token "|" for T5 and with token "." for PEGASUS.

Test set contains 1400 example for 200 unique texts, and was taken from VLDB 2021 only.

[4] https://huggingface.co/transformers/

Table 3
Self-evaluation test results

model	finetuned	WAcc	PAcc
baseline (N>1)		0.9092	0.9522
baseline (N>2)		0.9039	0.9412
HRRASA		0.9103	-
ROVER		0.9225	-
T5-small	-	0.7859	0.8774
T5-small	+	0.9429	0.9752
T5-base	-	0.8152	0.8985
T5-base	+	0.9520	0.9813
T5-large	-	0.85	0.9322
T5-large	+	**0.9683**	**0.9884**
PEGASUS-xsum	-	0.397	0.618
PEGASUS-xsum	+	0.939	0.9739

Table 4
Blind test results

Model	WAcc	PAcc	$\triangle WAcc$	$\triangle PAcc$
T5-large	0.9554	0.9803	-	-
with DeBert	**0.9565**	**0.9852**	+0.0011	+0.0049
with DeBert (>Q1)	0.9559	0.9831	+0.0005	+0.0028
with DSTC 2/3	0.9103	0.9311	-0.0451	-0.0492
with DSTC 2/3 (>Q1)	0.9238	0.9567	-0.0316	-0.0236

4. Results

Table 3 shows the results of the self-evaluation, where we use only VLDB2021 dataset for train and test. On this table we can see that the largest T5 model gives best results. It is also interesting that the T5 can perform well even without training, unlike PEGASUS model.

On the table 4 we show our experiments with datasets. A study was conducted to identify significant contributions to the proposed model performance. For all additional texts, the WAcc and PAcc scores on the blind test set is reported. The pretraining on DeBert dataset has a significant impact on both the $WAcc$ and $PAcc$ scores, also giving a more stable training. But training with DSTC 2/3 dataset gives much worse scores. In addition, we have tried to clean up some bad examples in our datasets: we calculate quantiles on $WAcc$ and remove the 1-quantile (the worst ones) from datasets. But clearing the data from bad examples did not significantly improve the quality of models.

5. Error Analysis

The first problem we had with models for summarization was the limitation on the output of text. We can partly control text generation. All models have been pretrained on the tasks of generating from a paragraph to few sentences, while our task requires only one sentence as the output. Therefore, in some cases, the model generated multiple sentences, which had a negative

impact on quality. We tried to counter this by replacing the "." token with the "|" token in the T5 model.

The second problem is that almost any additional data gives worse scores. This is probably because the original data has very good quality (due to been human crowd-sourced) at the same time DSTC2/3 and DeBert were computer partitioned.

6. Conclusion

This paper presents our approach to noisy text sequence aggregation, which is ranked second place in the VLDB 2021 Crowd Science Challenge. Our paper shows the effectiveness of the method. The error analysis also shows that the proposed approach can perform better with additional datasets.

In the future, we plan to adopt our model to the speech in other domains. We also plan to train the model to generate texts with ASR-noise.

References

[1] E. Egonmwan, Y. Chali, Transformer and seq2seq model for paraphrase generation, in: Proceedings of the 3rd Workshop on Neural Generation and Translation, Association for Computational Linguistics, Hong Kong, 2019, pp. 249–255. URL: https://www.aclweb.org/anthology/D19-5627. doi:10.18653/v1/D19-5627.

[2] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019. arXiv:1910.13461.

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2017. arXiv:1706.03762.

[4] D. Ustalov, N. Pavlichenko, I. Stelmakh, D. Kuznetsov, VLDB 2021 Crowd Science Challenge on Aggregating Crowdsourced Audio Transcriptions, in: Proceedings of the 2nd Crowd Science Workshop: Trust, Ethics, and Excellence in Crowdsourced Data Management at Scale, Copenhagen, Denmark, 2021.

[5] M. Henderson, B. Thomson, J. D. Williams, The second dialog state tracking challenge, in: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), Association for Computational Linguistics, Philadelphia, PA, U.S.A., 2014, pp. 263–272. URL: https://www.aclweb.org/anthology/W14-4337. doi:10.3115/v1/W14-4337.

[6] G. Cunha Sergio, M. Lee, Stacked debert: All attention in incomplete data for text classification, Neural Networks 136 (2021) 87–96. URL: http://dx.doi.org/10.1016/j.neunet.2020.12.018. doi:10.1016/j.neunet.2020.12.018.

[7] A. C. Morris, V. Maier, P. D. Green, From wer and ril to mer and wil: improved evaluation measures for connected speech recognition., in: INTERSPEECH, ISCA, 2004. URL: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2004.html#MorrisMG04.

[8] T. Cui, J. Xiao, L. Li, X. Jiang, Q. Liu, An approach to improve robustness of nlp systems against asr errors, 2021. arXiv:2103.13610.

[9] J. Fiscus, A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover), in: 1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings, 1997, pp. 347–354. doi:10.1109/ASRU.1997.659110.

[10] J. Li, Crowdsourced Text Sequence Aggregation Based on Hybrid Reliability and Representation, Association for Computing Machinery, New York, NY, USA, 2020, p. 1761–1764. URL: https://doi.org/10.1145/3397271.3401239.

[11] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research 21 (2020) 1–67. URL: http://jmlr.org/papers/v21/20-074.html.

[12] J. Zhang, Y. Zhao, M. Saleh, P. J. Liu, Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, 2020. arXiv:1912.08777.

[13] S. Narayan, S. B. Cohen, M. Lapata, Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 2018.