SR-MT: A Metamorphic Method to Test the Robustness of Speech Recognition Software

Feifei Wang
Naval University of Engineering
Wuhan, Hubei, China
wfeifei75336@163.com

Kerong Ben
Naval University of Engineering
Wuhan, Hubei, China
benkerong08@163.com

Xian Zhang
Naval University of Engineering
Wuhan, Hubei, China
919453887@qq.com

ABSTRACT

Speech recognition (SR) systems are systems that convert speech signals into text and are widely used in mobile devices, wearable devices, and intelligent living room devices. When SR software recognizes speech in the laboratory or another quiet environment, the recognition accuracy is high. However, when SR software is applied in a complex real-world environment, there may be background noise from birds, machines, and so on, the influence of other speakers, or other adverse factors, which will reduce the recognition accuracy of the SR software. In this work, we propose SR-MT, which is a metamorphic testing (MT) approach to test the robustness of SR software. In SR-MT, we adopt four criteria to analyze the MT results and rank the robustness of the target SR software for speech affected by noise interference and speech variations from level 1 to level 5. SR-MT was evaluated on three real industrial applications: iFLYTEK speech-to-text[1], Baidu speech-to-text[2], and Google speech-to-text[3]. We found that on average, 13.5% of the words in the speech could not be recognized correctly when the signal-to-noise ratio reached 10 dB. Similarly, changes in speech speed and tone will also reduce the recognition accuracy of SR software.

CCS CONCEPTS

• **Software and its engineering → Empirical software validation**.

KEYWORDS

metamorphic testing, metamorphic relations, speech recognition, robustness testing

ACM Reference Format:
Feifei Wang, Kerong Ben, and Xian Zhang. 2022. SR-MT: A Metamorphic Method to Test the Robustness of Speech Recognition Software. In *7th International Workshop on Metamorphic Testing (MET'22), May 9, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3524846.3527339

[1]https://www.xfyun.cn/services/lfasr
[2]https://ai.baidu.com/tech/speech/aasr
[3]https://cloud.google.com/speech-to-text

1 INTRODUCTION

Speech recognition (SR) software is becoming more widely used, and its validation is becoming more critical. Current research on the testing of SR software mainly focuses on the diversity of words and languages covered, ignoring the ability of the software to handle input speech subjected to interference from noise or unexpected forms of speech, such as fluctuations in pitch from high to low. For example, when a person gives the instruction "Turn on the light" to an intelligent living assistant, the assistant may instead turn on the refrigerator under the influence of the sound from a TV program. This example illustrates the instability of SR software in the face of small disturbances. On the other hand, testing SR software is challenging. A tester must acquire a large number of real-life inputs and check that the outputs meet the expectation, with the tester acting as a human oracle. Such validation-based testing is very expensive in terms of time and cost. Hence, the lack of a test oracle [2] is an important problem for SR software.

In the study reported in this paper, we used the metamorphic robustness testing approach [23] to test the robustness of SR software. Metamorphic testing (MT) [5] is an effective approach for alleviating the oracle problem. A central element of MT is a set of metamorphic relations (MRs), which are necessary properties of the target function or algorithm in relation to multiple inputs and their expected outputs. MT involves verifying both source and follow-up test cases as well as their outputs against corresponding MRs. If the actual outputs of the source and follow-up test cases violate a certain MR, then it can be concluded that the software under test is faulty with respect to the property associated with that MR. This kind of test result verification is strict, and it can judge only whether bugs are present in software; it cannot evaluate the severity of those bugs. We construct MR-1 based on the assumption that the recognition result of added-noise speech is the same as that of clean speech. For example, we input a clean speech with the content "The reasons for this dive seemed foolish" into Baidu SR software, and assume that the recognition result in text is "The reasons for this dive seemed foolish". Then we added chopping noise from the kitchen to the clean speech. The recognition result of the added-noise speech may be "A reasons for this dive seemed foolish", or "The reasons for the style seemed foolish", or "A reasons for the Steve seed fully". These three results all violate MR-1, but their degrees of violation are different, reflecting different levels of system robustness. For example, the first result for the follow-up speech is wrong for only one word, and the error has no effect on the sentence semantics. The second is wrong for two words, but the sentence semantics have changed. The third result has a high word error rate of 71%, and the sentence semantics are completely different.

Therefore, we use four evaluation metrics, addressing the four dimensions of words, phrases, sentences, and semantics, to analyze the degree of violation of MRs. According to the scores of the evaluation metrics, we divided the robustness of SR software into five levels, which gives a quantitative evaluation of the robustness of SR systems. The main contributions of this article are in three aspects:

- We develop four MRs for the verification of SR systems.
- We use four evaluation metrics addressing four dimensions, words, phrases, sentences, and semantics, to analyze the MT results and give a grade evaluation for the robustness of SR software.
- We assess the effectiveness of our proposed SR-MT approach in two languages (Chinese and English) by testing three real industrial SR applications. All of our data, the results, and the code are open source[4].

2 RELATED WORK

The effectiveness of MT technology has been continuously verified in a series of studies and has been widely applied in research fields such as Web services [19] and bioinformatics [15]. In development practice, MT has successfully found defects in several real programs, such as the Google and Bing search engines [24], NASA systems [12], and the YouTube and Spotify Web application programming interfaces(APIs) [17]. Brown et al. [3] also applied MT to navigation software and successfully found real defects in Google Maps.

In addition to traditional software applications, MT has also been widely validated in the field of intelligent software. A construction method for MRs based on data augmentation has been applied for various tasks in the field of image recognition, such as image classification [7], face recognition [25], and automatic driving [20]. In the field of machine translation, the typical testing and evaluation methods based on human recognition and reference translation rely on high-quality annotations with high acquisition costs. Therefore, a large number of scholars have adopted the method of MT to test machine translation software and have found errors in real translation software such as Google Translate [11].

Compared with image classification and machine translation, there has been relatively little testing research for SR systems.

Du et al. proposed DeepCruiser [6]. It is a white-box testing method that performs coverage-guided fuzzing by applying several metamorphic transformations to audio samples, such as changing the audio speed and volume, with the goal of generating audio that will be incorrectly transcribed by SR systems. DeepCruiser can produce a large number of test cases that SR software cannot recognize correctly, but it requires access to the SR model as well as existing audio files and their corresponding transcribed texts as input.

Asyrofi et al. proposed CrossASR [1], a black-box approach that capitalizes on existing text-to-speech (TTS) systems to automatically generate test cases for SR systems. CrossASR is a differential testing solution that compares the outputs of multiple SR systems to uncover erroneous behaviors among those systems. However, whether speech synthesized by TTS systems is suitable for testing SR systems is also a challenging question to answer.

[4]https://github.com/SIYUAN9446/SR-MT

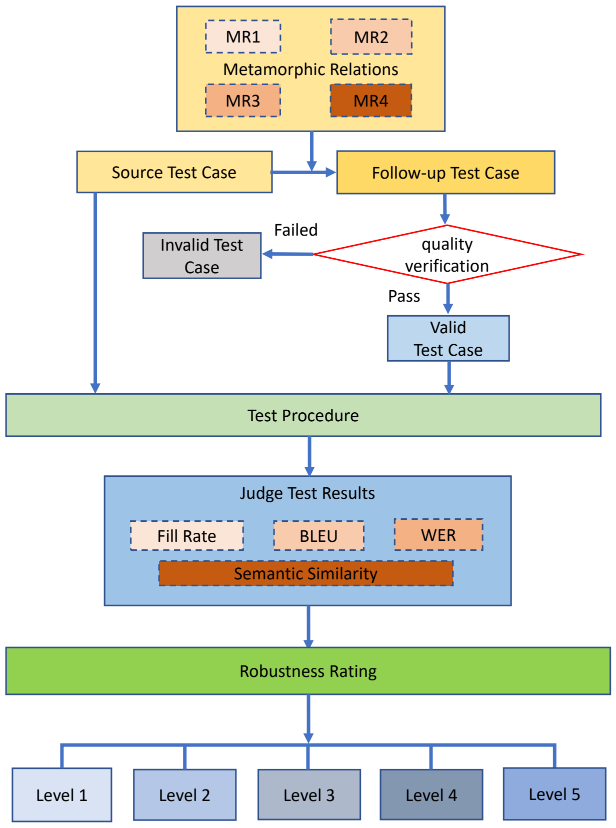


Figure 1: Workflow of SR-MT

There are several works in which adversarial attacks targeting SR systems have been performed (also known as adversarial test case generation) [16, 22]. These adversarial test methods focus on generating imperceptible adversarial audio examples, with the audio signal remaining barely distinguishable from the original signal, which rarely appear in real life. Additionally, similar to DeepCruiser, these approaches require existing audio files and their corresponding transcribed texts as input.

3 SR-MT METHOD

3.1 SR-MT workflow

Figure 1 shows the workflow of SR-MT. SR-MT contains three components:

1. Construction of metamorphic relations (MRs) to effectively assess the robustness of SR applications. SR systems perform general transformations from audio speech samples to the corresponding natural-language texts and are often expected to work properly on speech samples with various volume, speed and voice characteristics. In this work, we constructed four MRs by adding noise, varying the speech speed, varying the tone, and combining these variations.

2. Generation of follow-up test cases corresponding to the MRs and assessment of voice quality using a subjective evaluation method. After generating the follow-up test cases, we evaluated the voice

quality of the generated test cases using a subjective evaluation method. When the average subjective score exceeded a certain threshold, we considered the generated test case to be valid for use in subsequent tests.

3. Testing of public SR applications and performance of oracle evaluation. For this purpose, we adopted four metrics to analyze the MT results and ranked the robustness of the SR software from level 1 to level 5.

3.2 Metamorphic relations

The process of SR relies on extracting acoustic features such as mel frequency cepstral coefficients (MFCCs) or perceptual linear predictive coefficients from the original speech. As a primary consideration, we alter the acoustic characteristics of the speech data and construct corresponding MRs in accordance with the requirements for the robustness of SR applications. In addition, speech collected in a specific domain will have certain unique background sounds, for example, sounds from a bathroom, cars, an airport, and so on.

Considering the influence of background sounds and the acoustic characteristics of the speech itself, we propose four MRs:

(1) MR-1: Add noise to the source speech

(2) MR-2: Change the speed of the source speech

(3) MR-3: Change the tone of the source speech

(4) MR-4: Add noise to speech changed to a lower speed (combination of MR-1 and MR-2)

3.2.1 MR-1: Add noise to the source speech. MR-1 is based on the assumption that background noise introduced into source speech should not severely affect the recognition results. This MR is built on observations of real application scenarios. For example, in a clean environment, a home control assistant should be able to correctly recognize the speech command "turn on the light", and in a kitchen environment, the home control assistant should not instead recognize the speech command as "turn off the light" due to disturbance caused by the sound of chopping.

Thus, we find it reasonable to assume that added noise should not affect the results by a significant factor. Samples of the noise to be modeled were collected in the real world, including 15 kinds of noise, such as high-frequency channel noise, speech babble, and factory floor noise. When generating speech with noise, the pure speech and noise are synthesized in accordance with a certain signal-to-noise ratio (SNR), and this synthesis method yields speech close to that encountered in the real world. The SNR formula is as follows:

$$SNR(S(t), n(t)) = 10 \cdot \log \frac{\sum_t s^2(t)}{\sum_t n^2(t)} \quad (1)$$

Here, the SNR is expressed in dB, $\sum_t s^2(t)$ is the pure speech energy, and $\sum_t n^2(t)$ is the noise energy. In the experiment, we set the SNR to Synthetic speech with noise. In an experiment, we set the SNR for synthetic speech with noise. Figure 2 illustrates examples of the waveforms of pure speech and noisy speech.

3.2.2 MR-2: Change the speed of the source speech. MR-2 specifies that if we change the speed of the input speech, the recognition

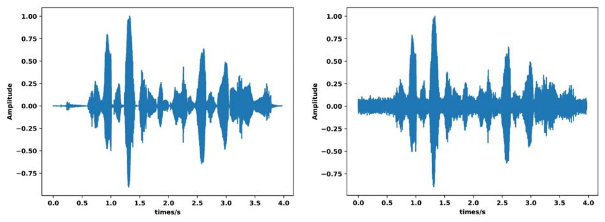


Figure 2: Waveforms of Speech Samples Satisfying MR-1

results should not change. To test this MR, we use the waveform-similarity-based overlap-add (WSOLA) [9] procedure, which is a digital signal processing method for stretching or compressing the duration of a given audio signal. After conversion, the time-scale-modified signal will sound as if the original signal content had been performed at a different tempo while preserving properties such as pitch and timbre. WSOLA uses a method of waveform superposition to achieve variable speed without modulation, with no effect on the fundamental frequency. It mainly includes the following steps: (1) divide the audio time domain into frames, (2) add a Hamming window to the first frame, (3) take an interval Ha and remove the second frame, (4) add a Hamming window to the second frame, and (5) superimpose with the previous frame by a step of Hs. The formulae are as follows:

$$\begin{cases} L_{out} = L_{in}/rate \\ rate = H_a/H_s \end{cases} \quad (2)$$

Here, L_{out} is the length of the converted speech, L_{in} is the length of the source speech, and rate is the rate by which the speech speed is changed. In an experiment, H_s was fixed to half of the frame length. Accordingly, the speed of the speech was adjusted by setting H_a. Figure 3 shows an example of two corresponding speech samples at different speed rates (1.0 and 0.75).

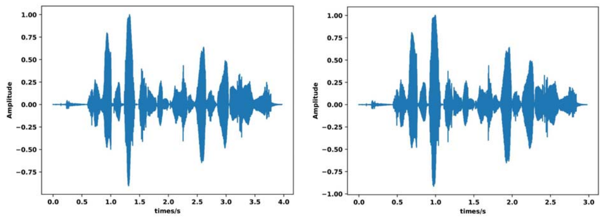


Figure 3: Waveforms of Speech Samples Satisfying MR-2

3.2.3 MR-3: Change the tone of the source speech. MR-3 specifies that if we change the tone of the input speech, the recognition results should not change. The tone can be changed through simple resampling, but in this case, the length of the speech sample will be changed at the same time. Therefore, based on the resampling results, the speech speed should also be changed by a corresponding factor. Accordingly, we use a combination of resampling and WSOLA to change the speech tone. After resampling, if the fundamental frequency is decreased, some of the high-frequency parts will be lost. To avoid this issue, we adopt the method proposed by Kobayashi K et al. [10] to repair high frequencies. Figure 4 shows that the period and frequency of the converted speech are different from those of the source speech.

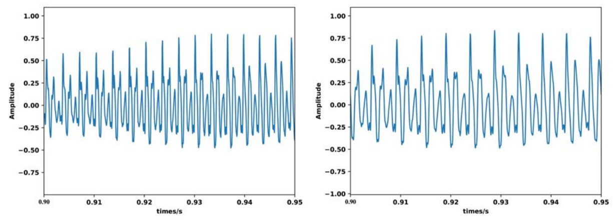


Figure 4: Waveforms of Speech Samples Satisfying MR-3

3.2.4 MR-4: Add noise to speech changed to a lower speed (combination of MR-1 and MR-2). Through experiments, we have found that SR applications not only may recognize words wrongly in speech with noise but also may identify the noise itself as additional words. For speech at a normal speed, SR applications tend to recognize such extra content before the beginning or after the end of the speech. However, when speech recorded at a slower speed is also affected by noise, might noise in the middle of the speech also be incorrectly recognized as words, thus affecting the recognition result? To answer this question, we formulate MR-4, which specifies that the recognition result should remain unchanged when the input speech is changed to low speed and noise is added simultaneously (a combination of MR-1 and MR-2).

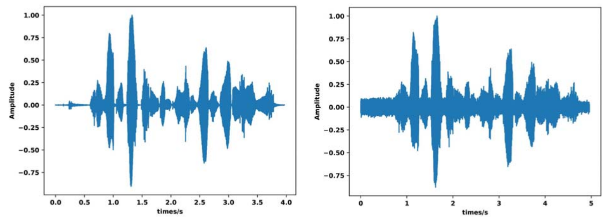


Figure 5: Waveforms of Speech Samples Satisfying MR-4

3.3 Oracle evaluation

3.3.1 Evaluation metrics. To verify the test case execution outcome against the MRs, we implement four evaluation metrics.

The Fill rate is defined as the ratio of the number of test cases (n) satisfying the MRs to the total number of test cases (N). The fill rate is formally expressed as follows:

$$Fill\,Rate = \frac{n}{N} \tag{3}$$

The fill rate reflects the satisfaction of the MRs at the sentence level. A lower fill rate for a test dataset implies higher fault detection capabilities.

Bilingual evaluation understudy (BLEU) [14] is used to evaluate the difference between a model-generated sentence (candidate) and the source sentence (reference). The value of the BLEU score ranges from 0.0 to 1.0; if the two sentences perfectly match, the BLEU score is 1.0, and if they perfectly mismatch, the BLEU score is 0.0. The BLEU method is realized by calculating n-gram models of the candidate sentence and the reference sentence separately and then calculating the number of matches. This comparison method is an assessment at the level of granularity of words and phrases. The n-gram probability calculation proceeds as follows. Suppose that there is a sentence consisting of N words, where each word is influenced by a set of preceding words, from the first word to the word immediately before it, then, the calculation method is as follows.

If the occurrence of a word depends only on the occurrence of the word before it, we call the corresponding probability the 2-gram probability:

$$p(S) = p(\omega_1\omega_2\cdots\omega_n) = p(\omega_1)p(\omega_2|\omega_1)\cdots p(\omega_n|\omega_{n-1}) \tag{4}$$

If the occurrence of a word depends only on the two words that precede it, we call this the 3-gram case:

$$p(S) = p(\omega_1\omega_2\cdots\omega_n) = p(\omega_1)p(\omega_2|\omega_1)\cdots p(\omega_n|\omega_{n-1}\omega_{n-2}) \tag{5}$$

To comprehensively measure speech similarity, we calculate BLEU-1 to BLEU-4, which reflect the satisfaction of the MRs at the phrase level.

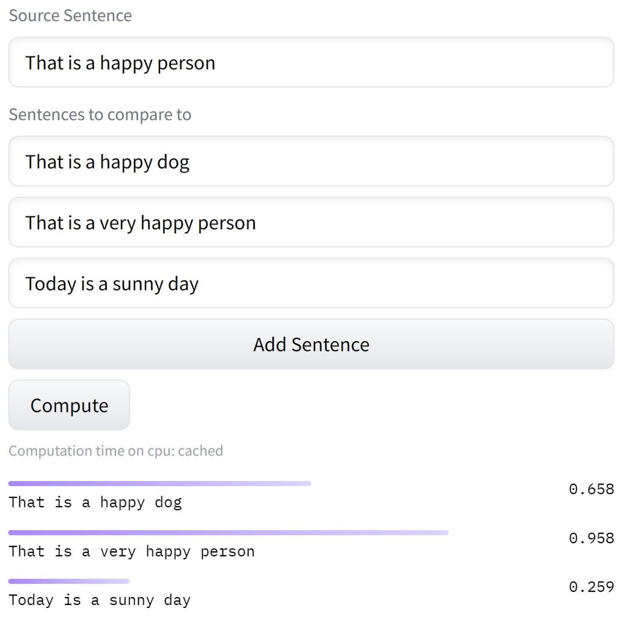
The word error rate (WER) [13] is a direct index of the SR rate. Its definition is as follows: to identify the consistency between a given word sequence and a standard sequence of words, we need to replace, delete, or insert some words such that the sequence to be evaluated becomes identical to the standard sequence. The total number of these insertions, replacements, or deletions of words divided by the number of words in the standard sentence is referred to as the WER, and the relevant formulae are as follows:

$$\begin{cases} WER{=}100\frac{S+D+I}{N}\% \\ Accuracy = 100 - WER\% \end{cases} \tag{6}$$

Here, S refers to substitutions, D refers to deletions, I refers to insertions, and N denotes the number of words. The WER reflects the violation of the MRs at the word level.

Semantic similarity [4] is a metric used to judge the similarity between natural language sentences while accounting for the variability of natural language expressions. The fill rate is a coarse-grained measure that can determine only whether the recognition results for test pairs are identical. The BLEU score and the WER measure the similarity of the recognition results at a word-level granularity. For instance, suppose that the source recognition result is "That is a happy person" and the follow-up recognition results are "That is a very happy person" and "That is a happy dog". These two pairs have the same scores in terms of the BLEU and WER metrics but have completely different meanings. Therefore, we use a sentence similarity model[5] based on Transformers to judge the semantic similarity between the recognition results for source test cases and corresponding follow-up test cases. An example is shown in figure 6.

[5]https://huggingface.co/sentence-transformers/multi-qa-MiniLM-L6-cos-v1



That is a happy person

That is a happy dog

That is a very happy person

Today is a sunny day

```
That is a happy dog              0.658
That is a very happy person      0.958
Today is a sunny day             0.259
```

Figure 6: An Example of Semantic Similarity Evaluation

3.3.2 Robustness rating. We proposed a rating method for the robustness of SR software. The specific implementation method is only applicable to SR-MT, but the idea of quantitative evaluation can be applied to other domains. The robustness is divided into five levels, with 1 representing very poor and 5 representing excellent. When the scores of $1 - WER$ and BLEU-4 are both in 0.8-1.0, the robustness level is 5, the highest, with 0.6-0.8 representing level 4, 0.4-0.6 representing level 3, 0.2-0.4 representing level 2, and 0.0-0.2 representing level 1. By analyzing the results of evaluation metrics, we can have a quantitative evaluation of the robustness of SR software.

4 EXPERIMENTS

In this section, we report an experiment conducted to demonstrate the validity of the proposed testing method.

4.1 Design of The Experiment

Applications under test: We selected three real SR applications available from the industry, namely, iFLYTEK speech-to-text, Baidu speech-to-text, and Google speech-to-text, which are online services invoked through APIs written in Python. These applications were used to evaluate the validity of the proposed testing method in mining SR applications for robustness bugs. Applying our testing method to multiple applications developed by independent vendors enables us to demonstrate the reusability and generalizability of the test strategies.

Datasets: We evaluated SR-MT on three datasets: an English speech dataset, a Chinese speech dataset, and a noisy speech dataset.

TIMIT [8] contains broadband recordings of 630 speakers of 8 major dialects of American English. TIMIT offers 52 phonemes, 6 closures, and 5 identifiers, fully considering articulation diversity, sentence type, and phoneme text diversity for the assessment of automatic SR systems.

THCHS-30[6] is an open Chinese speech database, which was acquired in a quiet office environment with a single carbon microphone for a total of over 30 hours. The text of THCHS-30 was selected from large-capacity news to improve vocabulary coverage.

Noise_92[7] contains 15 kinds of noise: white noise, pink noise, high-frequency (HF) channel noise, speech babble, two types of factory floor noise, two types of jet cockpit noise, destroyer engine room noise, destroyer operations room noise, F-16 cockpit noise, military vehicle noise, tank noise, machine gun noise, and car interior noise. All noise data files last 235 seconds and have a sampling rate of 19.98 kHz.

4.2 Experimentation Process

The experiment consisted of the following 3 steps.

1. Generation of the follow-up test cases

We randomly selected 100 speech samples each from the TIMIT and THCHS-30 datasets as the source test cases to generate the follow-up test cases. For MR-1, the introduced noise should be controlled to a reasonable degree such that it is lower in volume than the principal speech. Thus, we set the SNR to 0, 10, 30, and 50 dB, representing different noise levels. We synthesized the follow-up test cases using speech recorded in a quiet environment from the TIMIT and THCHS-30 datasets and noise from the Noise_92 dataset. For MR-2 and MR-3, we set the speed rate and the resampling rate to 0.75 and 1.25, respectively, to generate speech at different speeds and pitches. For MR-4, we added the same 15 kinds of noise used in MR-1 to the slowed speech satisfying MR-2, generating another 1500 test cases. All processing for the generation of follow-up test cases was realized through scripts, which can be automated.

2. Voice quality assessment of the follow-up test cases

Table 1: The MOS score

Index		Average score			
Data Set	Index	MR-1	MR-2	MR-3	MR-4
THCH-30	sharpness	4.3	4.4	4.3	4.5
	naturalness	4.2	4.2	4.1	4.1
TIMIT	sharpness	4.2	4.2	4.1	4.3
	naturalness	4.0	4.1	4.1	4.2

To evaluate the voice quality of the follow-up test cases, we adopted a subjective evaluation method in which human participants were asked to give subjective grading opinions on the quality of speech based on certain prespecified principles. Referring to the International Speech Conversion Challenge [21], naturalness and sharpness were used as subjective evaluation indexes. We invited 20 volunteers to assign a mean opinion score (MOS) [18], which has been widely used in listening tests, to quantify the speech quality. In the MOS assessment, the listeners rated the voice quality of the follow-up test cases using a five-point scale, with 1 representing very poor and 5 representing excellent.

[6]http://www.openslr.org/18/
[7]http://http//spib.rice.edu/spib/select_noise.html

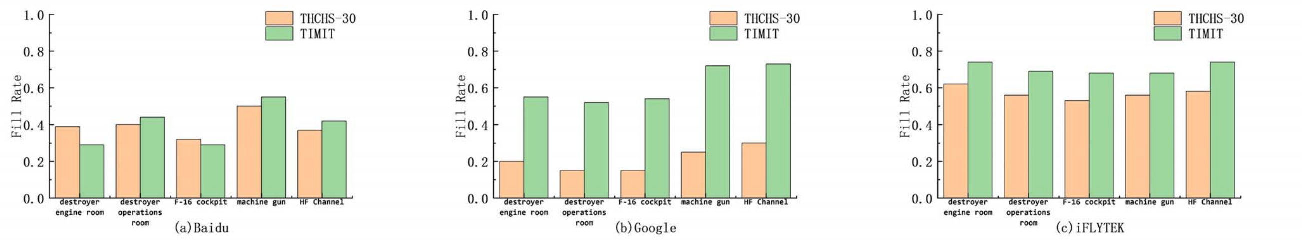


Figure 7: Results of MR-1

Table 1 shows the average scores of the generated test speech samples for each MR, which all surpass threshold 4, indicating that the follow-up test cases are valid.

3. Testing on the generated test cases

We constructed test pairs consisting of a source test case and a corresponding follow-up test case based on one of the MRs. These pairs were input into the three SR applications under test in batches. Then, we called the APIs of the SR platforms and obtained the recognition results. Based on the recognized text, the similarity between the recognition results for the source and follow-up test cases was measured in terms of the four previously introduced metrics, the fill rate, BLEU score, WER, and semantic similarity, which were calculated using scripts. Then, the rank of the robustness of SR software was evaluated from level 1 to level 5 according to the scores of WER and BLEU.

Accordingly, during the experimental process, the generation of the test cases and the analysis of MR violations could both be automated.

4.3 Experimental Results and Analysis

In the testing process, we found that it took almost 23 seconds for iFLYTEK to recognize a 5-second-long English speech sample but only approximately 1 second for Baidu and Google, which may be related to the algorithms and databases adopted by the different software. Therefore, we do not present comparisons between applications but instead analyze only how each application performs in the face of inputs with different types of interference. The experimental results are shown in tables, in which the values are presented as percentages.

4.3.1 Results in terms of the fill rate. Figure 7 shows the results for the fill rate of MR-1 obtained for 5 types of added noise among the 15 noise types in Noise_92 for an SNR of 10 dB. The results show that the violation ratios for MR-1 on the Baidu, Google, and iFLYTEK platforms reach 60%, 62%, and 37%, respectively, revealing the lack of robustness of these three SR applications when dealing with noise interference. We also find that all three platforms perform roughly the same in the face of different noise backgrounds when recognizing a given language. However, the Google speech-to-text platform exhibits poor performance when recognizing Chinese speech, far worse than its recognition performance for English.

For MR-1, we set the signal-to-noise ratio (SNR) to 0, 10, 30 and 50 dB to generate follow-up test cases in order to test the sensitivity of an SR system to noise. Figure 8 shows the MT results based on MR-1. We can find that when the SNR is 0 dB, the word error rate of SR software can be as high as 60%. When the SNR is 10 dB, the recognition accuracy is improved. When the SNR surpasses 30 dB, the noise has little impact on SR software, indicating that within a reasonable range, noise has little effect on the robustness of SR.

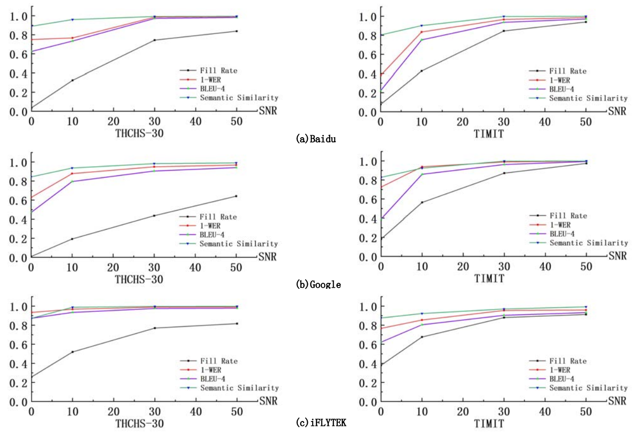


Figure 8: Results of MR-1

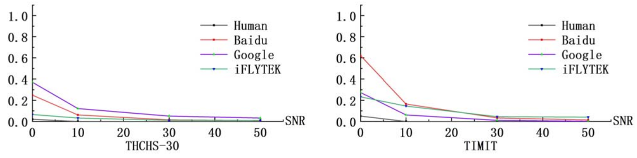


Figure 9: Comparision of Recognition Results of Human and SR Software

In addition, we performed another study to compare these results with those of human recognition. 20 volunteers were asked to recognize clean speech and speech with added noise. Then, we evaluated the recognition results using the same metrics used to analyze the results of the SR systems, as shown in Figure 9. When the SNR was 0 dB, that is, when the noise energy was the same as the speech energy, the study participants could still distinguish the noise well and achieve a high recognition rate. We can conclude that humans can effectively filter out noise in accordance with their own cognition and experience, while machines are more sensitive to noise.

Table 2 shows the fill rates for all four MRs on the two datasets. Among the four MRs, all three software platforms show the best performance for MR-2 and poor performance for MR-3 and MR-4. This indicates that the three SR applications are sensitive to pitch changes and have poor robustness for low-speed speech with added noise. To eliminate the concern of whether the generated speech quality affects the results, we performed additional analysis and found that the performance for MR-4 was poor even though MR-4 is a combination of MR-1 and MR-2, for which all SR applications performed well. Thus, we conclude that the SR applications have poor robustness for speech with pitch variations and for low-speed speech with added noise.

Table 2: Fill Rate

Application	**Dataset**	MR-1 (%)	MR-2 (%)	MR-3 (%)	MR-4 (%)
Baidu	THCHS-30	40.25	42.00	24.00	19.50
	TIMIT	39.25	61.50	30.50	31.75
Google	THCHS-30	18.75	21.50	14.00	10.50
	TIMIT	58.25	61.00	43.00	53.00
iFLYTEK	THCHS-30	56.75	52.50	31.00	50.75
	TIMIT	70.00	75.50	54.50	59.67

4.3.2 Results in terms of the BLEU and WER scores. BLEU and WER are to measure the impacts of noise interference and tone and speed changes on SR from the perspective of word and phrase matching. Figure 10 shows the results for the BLEU and WER scores under different MRs. In Figure 10, the scores of MR-1 are the MR with an SNR of 10 dB. It can be found that although the fill rates are low (between 0.3 and 0.6), the word matching rates are high (between 0.7 and 1.0). BLEU-1 score is similar to the WER, which is consistent with the underlying principles of these two indicators. The BLEU-4 score represents the match rate for four consecutive words, which is less than the WER by approximately 10%. These results indicate that the number of inconsistent words in the recognition results is low, which reflects good performance of the three SR systems at the word level.

Table 3: Semantic Similarity

Application	**Dataset**	MR-1 (%)	MR-2 (%)	MR-3 (%)	MR-4 (%)
Baidu	THCHS-30	94.37	95.13	73.34	95.76
	TIMIT	88.22	95.75	71.99	84.99
Google	THCHS-30	94.23	93.78	91.67	93.21
	TIMIT	93.15	92.04	86.54	92.25
iFLYTEK	THCHS-30	97.45	98.12	93.67	94.45
	TIMIT	92.96	93.33	86.32	89.88

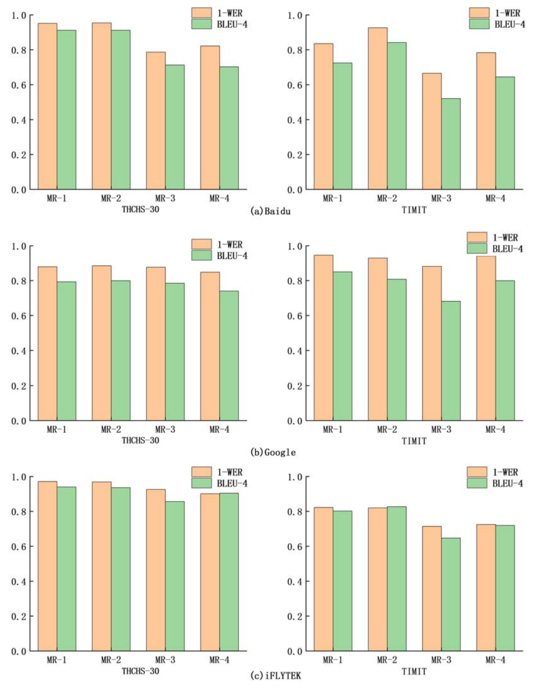


Figure 10: Results in terms of the BLEU and WER Scores

Table 4: Robustness Level of SR Systems

Robustness Level		Noise(SNR)				Speed change	Tone change
		0	10	30	50		
Baidu	THCHS-30	**4**	**4**	5	5	5	**4**
	TIMIT	**2**	4	5	5	5	3
Google	THCHS-30	**3**	4	5	5	5	4
	TIMIT	**3**	5	5	5	5	4
iFLYTEK	THCHS-30	5	5	5	5	5	5
	TIMIT	**4**	5	5	5	5	**4**

4.3.3 Results in terms of semantic similarity. The results of SR are often used as the inputs to command and control systems, so it is important to consider the semantics of the recognized sentences. Different positions and contents of a mistake in a word can lead to different meanings of the whole sentence. Therefore, we investigate the semantic similarity scores to measure the difference between the recognition results for the source and follow-up test cases. As table 3 shows, the semantic similarity scores of the recognition results of the three software platforms range from 0.72 to 0.98, revealing a lack of semantic coherence in recognition of these SR applications.

4.3.4 Results in terms of robustness level. In accordance with the word error rate (WER) and bilingual evaluation understudy at the 4-gram level (BLEU-4) results for the test cases, we performed a graded evaluation of the robustness of the three public SR platforms in the face of different types of environmental interference. The evaluation results are shown in Table 4. As we can see, when Baidu recognizes English speech with an SNR of 0 dB, the robustness level of the software drops severely, falling two grades lower than its robustness for Chinese.

5 CONCLUSION

In this paper, we have proposed a metamorphic testing (MT) approach named SR-MT to assess the robustness of speech recognition (SR) systems.

We assessed the effectiveness of SR-MT by testing three real industrial SR applications: iFLYTEK speech-to-text, Baidu speech-to-text, and Google speech-to-text. We found that on average, 13.5% of the words in speech samples could not be recognized correctly when the signal-to-noise ratio reached 10 dB. Similarly, changes in speech speed and tone will also reduce the recognition accuracy of SR software. The robustness of the three SR applications in the face of noise interference and variations in speech characteristics was rated from level 1 to level 5.

Different application scenarios impose different robustness requirements. For example, in voice control of the operation of a vehicle or a weapon launch, even a small recognition error may cause serious consequences, so high robustness is required. However, in smart home control or daily voice communication, the requirements for robustness are relatively low. Therefore, SR-MT can provide guidance in choosing the most appropriate SR software for different application scenarios.

In future research, we intend to identify additional MRs by considering MR output patterns (MROPs) [17] to cover more diverse scenarios. Moreover, the robustness evaluation method proposed in this paper is relatively coarse, and refining the evaluation index in accordance with the application scenario will also be an important objective in future work.

ACKNOWLEDGMENTS

To volunteers who contributed to speech quality evaluation and speech recognition.

REFERENCES

[1] Muhammad Hilmi Asyrofi, Ferdian Thung, David Lo, and Lingxiao Jiang. 2020. Crossasr: Efficient differential testing of automatic speech recognition via text-to-speech. In *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 640–650.

[2] Earl T. Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. 2015. The Oracle Problem in Software Testing: A Survey. *IEEE Trans. Software Eng.* 41, 5 (2015), 507–525.

[3] Joshua Brown, Zhi Quan Zhou, and Yang-Wai Chow. 2018. Metamorphic Testing of Navigation Software: A Pilot Study with Google Maps. In *51st Hawaii International Conference on System Sciences, HICSS 2018, Hilton Waikoloa Village, Hawaii, USA, January 3-6, 2018*, Tung Bui (Ed.). ScholarSpace / AIS Electronic Library (AISeL), 1–10.

[4] Dhivya Chandrasekaran and Vijay Mago. 2021. Evolution of semantic similarity—A survey. *ACM Computing Surveys (CSUR)* 54, 2 (2021), 1–37.

[5] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T. H. Tse, and Zhi Quan Zhou. 2018. Metamorphic Testing: A Review of Challenges and Opportunities. *ACM Comput. Surv.* 51, 1 (2018), 4:1–4:27.

[6] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Jianjun Zhao, and Yang Liu. 2018. Deepcruiser: Automated guided testing for stateful deep learning systems. *arXiv preprint arXiv:1812.05339* (2018).

[7] Anurag Dwarakanath, Manish Ahuja, Samarth Sikand, Raghotham M. Rao, and R. P. Jagadeesh Chandra Bose and. 2018. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis, ISSTA 2018, Amsterdam, The Netherlands, July 16-21, 2018*, Frank Tip and Eric Bodden (Eds.). ACM, 118–128.

[8] John S Garofolo, Lori F Lamel, William M Fisher, Jonathan G Fiscus, and David S Pallett. 1993. DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1. *NASA STI/Recon technical report n* 93 (1993), 27403.

[9] Shahaf Grofit and Yizhar Lavner. 2007. Time-scale modification of audio signals using enhanced WSOLA with management of transients. *IEEE transactions on audio, speech, and language processing* 16, 1 (2007), 106–115.

[10] Kazuhiro Kobayashi, Tomoki Toda, and Satoshi Nakamura. 2016. Implementation of F0 transformation for statistical singing voice conversion based on direct waveform modification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*. IEEE, 5670–5674.

[11] Dickson T. S. Lee, Zhi Quan Zhou, and T. H. Tse. 2020. Metamorphic Robustness Testing of Google Translate. In *ICSE '20: 42nd International Conference on Software Engineering, Workshops, Seoul, Republic of Korea, 27 June - 19 July, 2020*. ACM, 388–395.

[12] Mikael Lindvall, Dharmalingam Ganesan, Ragnar Árdal, and Robert E Wiegand. 2015. Metamorphic model-based testing applied on NASA DAT–An experience report. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 2. IEEE, 129–138.

[13] Sonja Nießen, Franz Josef Och, Gregor Leusch, and Hermann Ney. 2000. An Evaluation Tool for Machine Translation: Fast Evaluation for MT Research. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association.

[14] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*. ACL, 311–318.

[15] Laura L Pullum and Ozgur Ozmen. 2012. Early results from metamorphic testing of epidemiological models. In *2012 ASE/IEEE International Conference on BioMedical Computing (BioMedCom)*. IEEE, 62–67.

[16] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. 2019. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *International conference on machine learning*. PMLR, 5231–5240.

[17] Sergio Segura, José A Parejo, Javier Troya, and Antonio Ruiz-Cortés. 2017. Metamorphic testing of RESTful web APIs. *IEEE Transactions on Software Engineering* 44, 11 (2017), 1083–1099.

[18] Robert C. Streijl, Stefan Winkler, and David S. Hands. 2016. Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives. *Multim. Syst.* 22, 2 (2016), 213–227.

[19] Chang-Ai Sun, Guan Wang, Baohong Mu, Huai Liu, ZhaoShun Wang, and Tsong Yueh Chen. 2011. Metamorphic Testing for Web Services: Framework and a Case Study. In *IEEE International Conference on Web Services, ICWS 2011, Washington, DC, USA, July 4-9, 2011*. IEEE Computer Society, 283–290.

[20] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. DeepTest: automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27 - June 03, 2018*, Michel Chaudron, Ivica Crnkovic, Marsha Chechik, and Mark Harman (Eds.). ACM, 303–314.

[21] Tomoki Toda, Ling-Hui Chen, Daisuke Saito, Fernando Villavicencio, Mirjam Wester, Zhizheng Wu, and Junichi Yamagishi. 2016. The Voice Conversion Challenge 2016.. In *Interspeech*. 1632–1636.

[22] Piotr Żelasko, Sonal Joshi, Yiwen Shao, Jesus Villalba, Jan Trmal, Najim Dehak, and Sanjeev Khudanpur. 2021. Adversarial attacks and defenses for speech recognition systems. *arXiv preprint arXiv:2103.17122* (2021).

[23] Zhi Quan Zhou, T. H. Tse, and Matt Witheridge. 2021. Metamorphic Robustness Testing: Exposing Hidden Defects in Citation Statistics and Journal Impact Factors. *IEEE Trans. Software Eng.* 47, 6 (2021), 1164–1183.

[24] Zhi Quan Zhou, Shaowen Xiang, and Tsong Yueh Chen. 2015. Metamorphic testing for software quality assessment: A study of search engines. *IEEE Transactions on Software Engineering* 42, 3 (2015), 264–284.

[25] Hong Zhu, Dongmei Liu, Ian Bayley, Rachel Harrison, and Fabio Cuzzolin. 2019. Datamorphic Testing: A Method for Testing Intelligent Applications. In *IEEE International Conference On Artificial Intelligence Testing, AITest 2019, Newark, CA, USA, April 4-9, 2019*. IEEE, 149–156.