Differential Testing of Machine Translators Based on Compositional Semantics

Shuang Liu, *Member, IEEE*, Shujie Dou, Junjie Chen, Zhirun Zhang, and Ye Lu

***Abstract*—Powered by the advances of deep neural networks, machine translation software has achieved rapid progresses recently. Machine translators are widely adopted in people's daily lives, e.g., for information consumption, medical consumption and online shopping. However, machine translators are far from robust, and may produce wrong translations, which could potentially cause misunderstandings or even serious consequences. It is thus critical to detect errors in machine translators, and provide informative feedback for developers. In this work, we adopt the differential testing method to test machine translators. In particular, we use mature commercial translators as reference machine translation engines. Based on the principle of compositionality, which specifies that the meaning of a complex expression is determined by the meanings of its constituent expressions and the syntactic rules used to combine them, we design the oracle which conducts similarity comparison guided by syntactic structure and semantic encoding. In particular, we employ the constituency parsing to obtain the part-whole structure relation between a sentence and one of its component. Then we compute the semantic similarity of each sentence part with pre-trained language model and expert knowledge. We implement our approach into a tool named DCS, conduct experiments on three popular machine translators, i.e., Google translate, Baidu translate and Microsoft Bing translate, and compare DCS with two state-of-the-art approaches, i.e., CIT and CAT. The experiment results show that DCS achieves 8.6% and 35.4% higher precision, respectively. Moreover, the errors reported by DCS have the lowest redundancy in terms of the duplicated error locations in the source sentence. DCS can be used in complement with existing approaches and achieve higher detection precision. It also shows comparable efficiency with state-of-the-art approaches.**

***Index Terms*—Software testing, software quality, machine translation.**

Manuscript received 31 January 2023; revised 17 September 2023; accepted 2 October 2023. Date of publication 13 October 2023; date of current version 12 December 2023. This work was supported in part by the National Natural Science Foundation of China under Projects 61972403, 61802275, and 62372253, in part by the CCF-AFSG Research Fund under Grant RF20210031, in part by the CCF-Huawei Populus Grove Fund under Grant TC2022005, in part by the Key Laboratory of Safety-Critical Software (Nanjing University of Aeronautics and Astronautics), Ministry of Industry and Information Technology under Open Project NJ2022027, and in part by the Natural Science Foundation of Tianjin Fund under Grant 3JCYBJC00010. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Shiva Nejati. *(Corresponding author: Ye Lu.)*

Shuang Liu is with the School of Information, Renmin University of China, Beijing 100872, P.R. China (e-mail: liushuangcs@gmail.com).

Shujie Dou and Junjie Chen are with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: doushujie@tju.edu.cn; junjiechen@tju.edu.cn).

Zhirun Zhang is with the School of New Media and Communication, Tianjin University, Tianjin 300072, China (e-mail: zzr2022245010@tju.edu.cn).

Ye Lu is with the College of Cyber Science, Nankai University, Tianjin 300350, China. He is also with Tianjin Key Laboratory of Network and Data Security Technology, Tianjin 300350, China; State Key Laboratory of Processors, ICT, CAS, Beijing 100190, China; and the Key Laboratory of Data and Intelligent System Security (DISSec), Ministry of Education, Tianjin 300350, China (e-mail: luye@nankai.edu.cn).

Digital Object Identifier 10.1109/TSE.2023.3323969

I. Introduction

MACHINE translators, e.g., Google Translate [1] and Microsoft Bing Translate [2], are widely adopted for international communications in various domains, including academic, tourism and commercial activities. It is reported that Google Translate was adopted by more than 500 million people and 101 languages as in 2016 [3]. The current commercial machine translation software, e.g., Google Translate, are reported to provide translations with quality at human parity [4]. The wide adoptions of machine translators greatly improve the communication efficiency between people from different areas of the world. However, like traditional software, there have been reports on translation errors, which cause serious results, including financial losses [5], risks to personal lives [6], and even Diplomatic accident [7], and become the source of international tension and conflict [8]. Table I shows the translation error types defined in [9], [10].[1]

Given the wide adoption of machine translators and the losses caused by translation errors, it is enormously important to ensure the quality of machine translators. Various approaches [11], [12], [13], [14], [15] are proposed to test machine translators and a number of bugs are detected. Those approaches mainly conduct mutations on a single word of a given type to obtain a mutate sentence [11], [13], [14], [15], and employ certain metamorphic relations, e.g., structure invariance [12], pathological invariance [13], and referential transparency [14], to build the test oracle. These approaches are limited by the type of words that can be replaced, and the mutated sentences may introduce errors themselves, resulting in false positives. Moreover, Those approaches rely on text-level comparison metrics, e.g., BLEU [16] and edit distance [17], or syntactic structures to measure the similarity of translated sentences. Yet those metrics cannot reflect semantic similarity of sentences, and thus cannot not fully capture the quality of a translation [18]. CIT [12] employs constituency invariance relation for test case generation and oracle construction. Different from approaches which replace a single word, CIT generates test cases that share

[1]The meaning of the translated sentences are labelled by five volunteers who have more than 10 years of professional English writing experiences.

TABLE I
TRANSLATION ERROR TYPES AND EXAMPLE ILLUSTRATION

Error Type	Description	Source sentence	Translated sentence	Meaning of Translated Sentence
Wrong Choice of Words (WCW)	The original word has a corresponding translation, but the translation is wrong or inaccurate	The entrepreneur showed an amiable demeanor as he engaged clients who were **sampling** different models of Tecno brand of mobile phones.	这位企业家表现出和蔼可亲的举止，因为他的客户正在品尝不同型号的Tecno品牌手机。(By Bing)	The entrepreneur showed an amiable demeanor as he engaged clients who were **tasting** different models of Tecno brand of mobile phones.
Under Translation (UT)	Word of the source sentence is omitted or copied.	As always with **such ventures**, Hollywood's agents and content makers were happy to take money from a new buyer.	与往常一样，好莱坞的经纪人和内容制作者很乐意从新买家那里拿钱。(By Bing)	As always, Hollywood's agents and content makers were happy to take money from a new buyer.
Unclear Logic (UL)	The words in the source sentence are translated in the wrong order.	**He points out everything done wrong in the original egg fried rice video**, a response that has gathered more than 17 million views so far.	他指出，在最初的蛋炒饭视频中做错了一切，到目前为止,这一反应已经收集了超过1700万的观看次数。(By Google)	**He pointed out that he did everything wrong in the original egg fried rice video**, a response that has gathered more than 17 million views so far.
Opposite Meaning (OM)	The semantics of the target sentence is the opposite of the source sentence.	President Donald Trump participated in **head-to-head** town halls Thursday night.	周四晚上，唐纳德·特朗普总统参加了迎头痛击的市政厅活动。(By Baidu)	President Donald Trump participated in a **head-on attack** at the town halls Thursday night.
Named Entity Problem (NEP)	The named entity in the source sentence is incorrectly translated or omitted.	He was expected to miss two months but returned two weeks early and has played in the last three **Scarlets** matches.	他本应缺席两个月的比赛，但提前两周复出，并参加了最近三场红牌比赛。(By Baidu)	He should have missed the game for two months, but returned two weeks early and participated in the last three **red card** matches.
Quantifier/Time Problem (Q/T-P)	Incorrect or no translation of quantity or time words in the source sentence.	He **is getting used to** the new lifestyle of spending more time with his family and dogs as well as exercising every day.	他已经习惯了新的生活方式，每天花更多的时间与家人和狗在一起，并锻炼身体。(By Baidu)	He **has gotten used to** his new lifestyle, spending more time with his family and dogs and exercising every day.
Others (OTS)	Errors that are not of the above types.	He says the pandemic has been a shot of adrenaline for a technology that to date had not yet really arrived.	他说，这场大流行给迄今为止尚未真正到来的一项技术注入了肾上腺素。(By Google)	He says the pandemic **has injected** adrenaline into a technology that to date had not yet really arrived.

the same main part and differ by adjunct parts. CIT shows better performance compared with single-word mutation based approaches, yet the different pairs generated by this approach may still refer to the same issue in the original sentence (refer to discussions in Section IV.C.1), and thus provides redundant information which requires more manual analysis efforts.

To summarize, existing approaches all rely on a single translate engine to generate translated sentences, and are not able to detect the errors that exist in the non-mutated parts of the sentence. They employ simple distance-based or syntax-level metrics to compare the similarity of the translated sentences, which cannot reflect semantic similarity and thus cause high false positive and false negative rate. Moreover, multiple mutated sentences usually uncover redundant errors, this redundancy requires more manual analysis efforts.

In this work, we propose a differential testing based method named DCS. Given an input sentence in English, DCS obtains two translated sentences (in Chinese) from two different machine translators, e.g., Google and Baidu. Based on the principle of compositionality [19], which specifies that the meaning of a complex expression is determined by the meanings of its constituent expressions and the rules used to combine them, DCS designs the oracle which conducts similarity comparison guided by syntactic structure and semantic encoding. In particular, we employ the constituency parsing [20] to obtain the part-whole structure relation between a sentence and one of its component. Then we compute the semantic similarity of each sentence part with pre-trained language model SimCSE [21] and expert knowledge. Lastly, we detect the translation errors based on the similarity scores and report the mapping from the target error translation to the source English sentence to help locate bugs.

We conduct experiments with three datasets, one self-constructed dataset consisting of 800 sentences of 8 different categories and the other two datasets released by existing approaches [11], [12] The comparison results with CIT [12] and CAT [22] show that DCS achieves 8.6% and 35.4% higher average precision, and 42.8% and 7.7% higher average recall than CIT and CAT, respectively, on the three machine translators we tested. Moreover, the errors reported by DCS have the lowest redundancy in terms of the duplicated error locations in the source sentence. DCS also shows comparable efficiency with state-of-the-art approaches.

In summary, our approach has the following contributions.

- We propose the first differential testing method for machine translators, and propose a novel oracle based on the principle of composition. Our approach is orthogonal to existing approaches and can be combined with existing approaches.
- We conduct experiments on three mainstream machine translators, i.e., Google Translate, Baidu Translate and Bing Translate, on three datasets and compare DCS with two state-of-the-art approaches CIT and CAT. Experiment results show DCS outperforms both compared tools on precision, recall as well as the redundancy level, and achieves comparable efficiency with the compared approaches.
- We implement the proposed approach into a tool named DCS and the source code is publicly available[2].

[2]https://github.com/tjudoubi/toolDCS

Fig. 1. The motivation example.

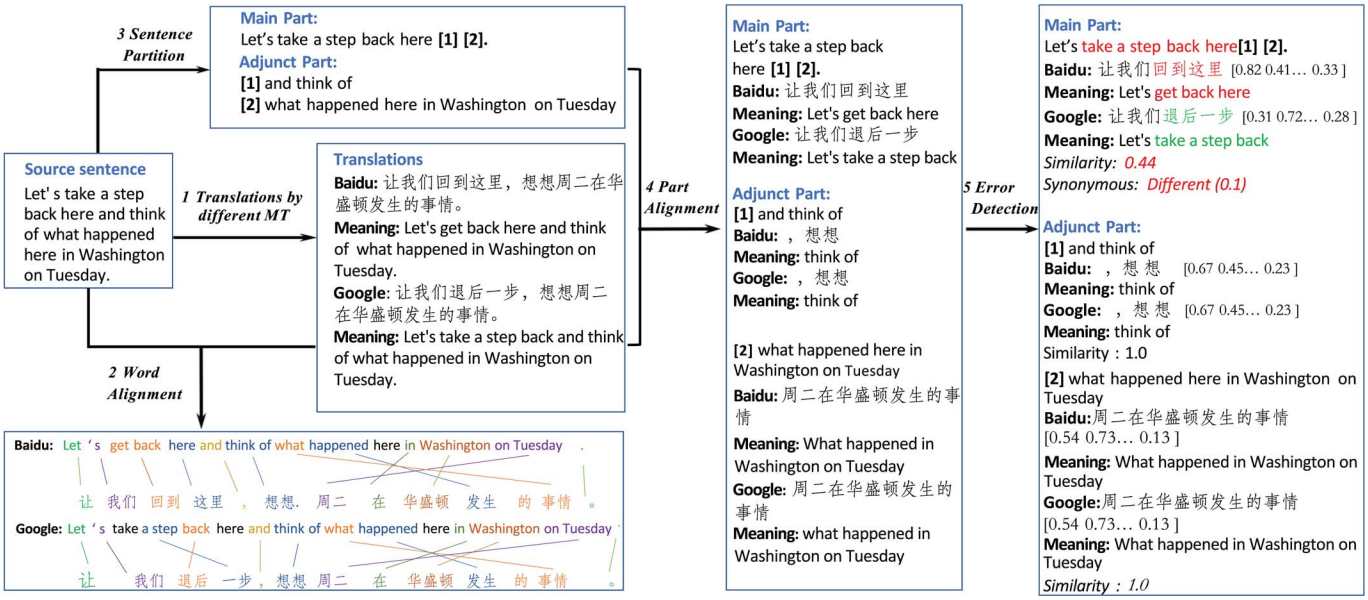

Fig. 2. Overview of our approach.

The paper is organized as follows. We provide a motivation example in Section II. In Section III we introduce the details of our method. Section IV reports the experiment result and analysis on it. We discuss related work in Section V and Section VI concludes the paper.

II. Motivation Example

We provide a motivation example, shown in Fig. 1, to more clearly describe the motivation of our approach. In Fig. 1, the seed sentences and the mutated sentences, the Chinese translations as well as the meaning of the Chinese translations are provided. The underlined words are mutated words and the bold shadowed words are words that are incorrectly translated.

We can observe in the motivation example of Fig. 1(a) that, both the seed sentence and the mutated sentences contain the same error, i.e., the phrase "flown to" is translated to "被派往" , which means "been dispatched to". Existing approaches based on different metamorphic relations are not able to detect this kind of error, since only one translate engine is used to generate translations, and the same phrase tends have identical translations.

Fig. 1(b) shows the example of redundant errors. Although multiple mutated sentences are obtained from the seed sentence, yet the mutated sentences uncover the same error, i.e., the wrong translation of word "hitting". Existing approaches [12], [22] generate a large number of mutated sentences from one source sentence, and thus result in high redundancy in detected errors, which require heavy manual efforts to analyze those errors.

III. Our Approach

The overview of our approach is shown in Fig. 2, which consists five main steps. Given an unlabeled monolingual English sentence, DCS first obtains two different Chinese translations from two machine translators, e.g., Google translate and Baidu translate in our running example. Then DCS conducts word alignment from the English sentence to the two corresponding Chinese translations. Afterwards, DCS conducts sentence compression and constituency parsing to obtain the main part and the adjunct parts of the sentence, respectively. Based on the word-level alignment information and the sentence partitions, we can obtain the aligned translations for the sentence main part and the adjunct parts. The last part is error detection based on the computed semantic similarity. Recall that the main intuition of our approach is that the translations from different translators should be semantically consistent. Based on the principle of

compositionality, which specifies that the meaning of a complex expression is determined by the meanings of its constituent expressions and the syntactic rules used to combine them, we compute the semantic similarity of the corresponding parts of the translated sentences. In particular, we adopt the pre-trained machine learning model SimCSE [23], and assisted with synonymous checking, to measure the semantic similarity. Note that our method is able to accurately locate the erroneous phrases based on the word-level alignment. Moreover, the translation of the reference engine can also provide a good reference for ground truth translations, and thus reduce the fixing efforts.

A. Translate by Different Machine Translation Software

As the first step, we automatically invoke the official APIs provided by the testing targets, i.e., Google Cloud Translation [24], Azure Microsoft Translator API [25] and Baidu Translate API [26], to obtain translated sentences (in Chinese). For each source sentence, we invoke the three APIs to obtain three translated sentences and then pair the translated sentences to obtain three translated sentence pairs, i.e., ⟨Google, Bing⟩, ⟨Bing, Baidu⟩ and ⟨Baidu, Google⟩. Note that the orders of the translated sentences in the pairs are not important. Fig. 2 shows a running example of the translated sentence pair of Google and Baidu.

B. Word Alignment

After obtaining the translated sentences from each machine translation software, we conduct word alignment, which aligns each English word in the source sentence to its semantic equal Chinese word in each translated sentence. The running example in Fig. 2 illustrates the result of word alignment, we use different colors to represent different words and use the solid line to show the correspondence relation.

Word alignment is a natural language processing technique that establishes a connection between two words if there is a translation relationship between them. We adopt the state-of-the-art word alignment model AWESOME [27], which achieves the best performance on word alignment in five languages including English-Chinese alignment. AWESOME adopts the pre-trained language models, fine-tunes them on parallel text with various objectives designed to improve alignment quality. The input of AWESOME are tokenized source sentence and its corresponding translated sentence, and the output is a list of numeric pairs i-j, which indicates that the i_{th} word of the source sentence is aligned with the j_{th} word of the translated sentence.

Note that, in English-Chinese translations, there are cases where named entities are not translated into Chinese. The reasons could be that the named entity does not have a corresponding Chinese word, or the Chinese translation is a transliteration such that there are more than one word/phrase translations in Chinese. As a result, named entity translations could introduce false positives, where both translations are correct. Fig. 3(a) shows an example of named entity translation, where "Capital Economics" is translated to "凯投宏观" (Capital Economics) by Google translate, yet Baidu translate does not translate it. The sentences translated by Google Translate and Baidu Translate are semantically equal, yet the verbal difference between the translations of "Capital Economics" will result in a false positive. To mitigate this problem, we conduct named entity replacement on the translated sentences, in which we replace the translated named entity (i.e., in the target language) with the named entity in the source language based on the word alignment information. We first search the translated sentence (in Chinese) in one translation software (e.g., Baidu) for English phrases. When an English phrase is identified, we check the word alignment information to locate its correspondence in the source (English) sentence. Then the located English phrase in the source sentence is used to align for the Chinese translation in another translation software (e.g., Google). In this way, we find the aligned named entity in the two translated sentences. Then we replace the Chinese translation with its aligned English Phrase. For instance in Fig. 3(b), we replace "凯投宏观" (Capital Economics) with "Capital Economics" in the sentence translated by Google Translate. In this way, we eliminate the potential false positives caused by named entities. In particular, name entity replacement reduces 29% false positives.

Fig. 3. Example of named entity replacement.

C. Sentence Partition

As the third step, we conduct sentence partition, where sentence compression and constituency parsing [28] are conducted to obtain the structural compositions of the given source sentence.

We adopt the state-of-the-art sentence compression model SLAHAN [29], which is a Seq2Seq model that generates simple sentences that remove redundant information from the sentence and retain the grammatical structure and important content. The output of SLAHAN constitutes the main part of the source sentence. For the remained sentence, we use constituency structure to search for every adjunct part. For instance, in Fig. 4, the example "Let' s take a step back here." is identified as main part by the SLAHAN model, and the remaining sentence is divided into two adjunct parts, i.e., "and think of" and "what happened here in Washington on Tuesday".

Algorithm 1 shows the detailed process. It takes the source sentence as input, and produces a part list, which contains the main part and a list of adjunct parts. The algorithm first conducts sentence compression with SLAHAN to obtain the main part (line 2), and constituency parsing with Stanford parser [28] to obtain the constituency parsing tree (line 3). Then it removes the main part from the constituency parsing tree (line 4) and cuts the remaining of constituency parsing tree to obtain adjunct parts (line 5). Function *cutAdjunct* recursively visit each node of the remaining constituency parsing tree, and collects the

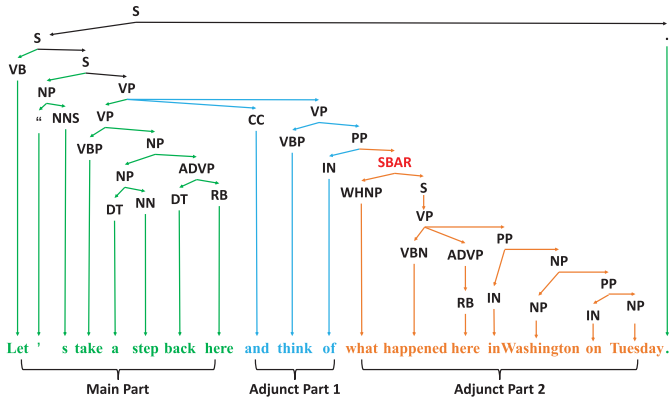

Fig. 4. Example of sentence partition.

Algorithm 1: Sentence Partition

Input : *sourceSentence*
Output: *PartsList*: Including Main part and adjunct parts

1 Initialize $PartsList$, $leafList$
2 $mainPart$ = *SentenceCompression* ($sourceSentence$)
3 $Tree$ = *ConstituencyTree* ($sourceSentence$)
4 $adjunctTree$ = *remove* ($Tree$, $mainPart$)
5 *cutAdjunct*($adjunctTree.root$,$leafList$, $PartsList$)
6 **if** $leafList$ *is not* $Null$ **then**
7 ADD $leafList$ TO $PartsList$
8 **end**
9 ADD $mainPart$ TO $PartsList$
10 **Function** `cutAdjunct` ($node$, $leafList$, $PartsList$) **:**
11 **if** $node$ *is leaf* **then**
12 ADD $node$ TO $leafList$
13 **end**
14 **else**
15 **if** $node.attribute \in Clause$ **then**
16 Initialize $newLeafList$
17 **foreach** $child$ *in* $node$ **do**
18 *cutAdjunct*($child$,$newLeafList$,$PartsList$)
19 **end**
20 **if** $newLeafList$ *is not Null* **then**
21 ADD $newLeafList$ TO $PartsList$
22 **end**
23 **end**
24 **else**
25 **foreach** $child$ *in* $node$ **do**
26 *cutAdjunct*($child$,$leafList$,$PartsList$)
27 **end**
28 **end**
29 **end**

leaf nodes, i.e., the textual information. When the clause node attribute, e.g., SBAR which indicates the root of a clause, is encountered, a new part is found and is added to *PartList* (lines 15–22).

D. Part Alignment

After obtaining the structural compositions of sentences and the word-level alignment, we conduct part alignment, in which we align each part obtained in sentence partition, of the source sentence with its corresponding translated part. This step is quite intuitive. Given a part of the source sentence, we simply collect the corresponding Chinese words in the translated sentence based on the word alignment results, and order them consistently with their positions in the translated sentence. The parts in the translated sentences, which align with the same part of the source sentence will form a part pair.

The running example in Fig. 2 provides a clear illustration on this, in which "what happened here in Washington on Tuesday" is aligned with "周二在华盛顿发生的事情" (which means "what happened in Washington on Tuesday") for both Baidu translate and Google translate. Therefore, we obtain a part pair of the translated sentence, i.e., ⟨周二在华盛顿发生的事情, 周二在华盛顿发生的事情⟩ (which means ⟨what happened in Washington on Tuesday⟩, ⟨what happened in Washington on Tuesday⟩).

E. Error Detection

The traditional process of difference testing calculates the similarity of the translated sentences by directly applying metrics like BLEU [16], ROUGE [30], Edit Distance [17]. However, those metrics cannot reflect semantic similarity of sentences, and thus cannot fully capture the quality of a translation [18]. Recently, pre-trained language models, e.g., BERT [31] and its derivatives, have shown impressive performance on semantic representation tasks and thus are good candidates for semantic similarity comparison. However, the most popular pre-trained language model BERT [31] is shown to have performance decrease on long text for classification tasks [32]. Our preliminary experiment, which conducts full sentence semantic representation with BERT also confirms this conclusion. Moreover, it has been shown that that BERT tends to encode all sentences into a small spatial region (referred to as "collapse"), which makes most of the sentence pairs have high similarity scores, even for sentences that are semantically completely unrelated [33].

To solve the issue of "collapse" on long sentences, we employ the principle of compositionality [19], which was first proposed by Frege and deeply affects the improvement of natural language processing [34]. The principle says that the meaning of a complex expression is determined by the meanings of its constituent expressions and the rules used to combine them.[3] Based on the principle of compositionality, we conduct error detection on each part of the translated sentence. If any part is wrongly translated, then the entire sentence is wrongly translated. In this way, we transform the semantic comparison on the sentence level to comparison on the sentence part level, which dramatically eliminate the issue of "collapse" on long sentences. As we adopt the differential testing methodology, the error detection is conducted by checking the semantic similarity of the part pairs obtained through part alignment.

To compare the semantic similarity sentence parts, we combine semantic similarity analysis with synonym checking, which employs the power of both pre-trained language models and the expert knowledge.

Semantic Similarity Analysis We adopt the SimCSE framework [21], which enhances the pre-trained language models,

[3]Note that there had been arguments on principle of compositionality and Pelletier [35] had provided a wonderful discussion.

e.g., BERT [31] and RoBERTa [36], with contrastive objective functions for sentence embedding. The model can improve the sentence representation ability by using a self-supervised training method, which comprehensively surpasses the previous methods on the semantic text similarity task. In particular, we chose the SimCSE model with fine-tuning using Chinese semantic similarity training data as our representation model to obtain sentence vectors. Then cosine similarity on the sentence vectors is calculated to obtain the similarity score.

Like existing approaches which manually set threshold values [11], [13], we need to take a trade off between false positives and false negatives. In our approach, high similarity score means semantic consistency between the two parts. However, setting an extremely high score may result in a large number of false positives, i.e., semantic similar pairs are reported as issue pairs, while setting a low similarity score may result in high numbers of false negatives, i.e., semantic inconsistent pairs are missed. Through analysis on the false positives, we observe that most of them are due to synonym words, which are not properly embedded by the language model. Based on this observation, we set the similarity threshold to be 0.75, which is a relatively low threshold to control the number of false negatives. We further conduct synonym checking to eliminate potential false negatives.

Synonym Checking We employ the power of both expert knowledge and pre-trained language models for synonym checking. In particular, we adopt the authoritative Chinese synonym dictionary, the HIT IR-Lab Tongyici Cilin [37], as resources of expert knowledge. Limited by the small number (77,343 words) of synonyms contained in the HIT IR-Lab Tongyici Cilin, we further conduct a word-level synonym analysis using the bag of word model. The detailed checking procedure is shown in Algorithm 2, where the input is a pair of sentences to be checked and a threshold value for the bag-of-word model. We first obtain a *wList*, a token sequence cut from a given sentence (lines 1–2). Then we use synonym dictionary to check the similarity of the two token sequences (line 3). We further conduct a word-level synonym analysis with word to vector model (line 4). The two sentences pass the synonym checking if they pass either the synonym dictionary check or the word-level synonym analysis (line 5).

Function `synoDictCheck` (lines 6–11) checks the synonym similarity of two token sequences. It queries Tongyici Cilin (the authoritative Chinese synonym dictionary) to obtain a synonym list dictionary for each given token sequence (lines 7–8), i.e., one synonym list for each token in the sequence. Then function `synonymCheck` is called to cross check the synonyms of the paired sentences (lines 9–10), i.e., $sent_1$ is checked against the synonym list dictionary of $sent_2$, and vice versa. `synonymCheck` simply removes all stop words and words which are in the given synonym list dictionary from the given sentence (lines 13–18). If there are no words left in the given sentence, then a true flag is returned (line 19). Note that we conduct the checking for both sentences (lines 9–10), only when both checking returns `True`, then we report `True` (line 11), indicating semantic similar.

Algorithm 2: Synonym Checking

```
   Input : sent1, sent2 // the sentence pair to be compared.
           θbow //threshold for bag-of-word checking.
   Output: True/False
1  wList1 ← cutWord(sent1)
2  wList2 ← cutWord(sent2)
3  Flagd ← synoDictCheck(wList1,wList2)
4  Flagb ← bowCheck(wList1,wList2)
5  return Flagd || Flagb
6  Function synoDictCheck(wList1,wList2):
7      synoListDict1 ← getSynonym(wList1)
8      synoListDict2 ← getSynonym(wList2)
9      flagA ← synonymCheck(synoListDict1,sent2)
10     flagB ← synonymCheck(synoListDict2,sent1)
11     return flagA ∧ flagB
12 Function synonymCheck(synoListDict, sent):
13     foreach synoList in synoListDict do
14         foreach word in synoList do
15             remove word from sent
16         end
17     end
18     remove all stop words from sent
19     return len(sent) == 0
20 Function bowCheck(wList1,wList2):
21     Remove identical words and stop words from wList1
         and wList2
22     vBag1=word2vec(wList1)
23     vBag2=word2vec(wList2)
24     score =
         min(sim(vBag1, vBag2),sim(vBag2, vBag1))
25     return score > θbow
26 Function sim(vBag1,vBag2):
27     s=1
28     foreach xi in vBag1 do
29         s = min(max({cos(xi,yj) | yj in vBag2 }),s)
30     end
31 return s
```

Limited by the small number (77,343 words) of synonyms contained in the HIT IR-Lab Tongyici Cilin, we may still have false positives. Function `bowCheck` (lines 20–25) is called to conduct check the word-level similarity of two given token sequences. After deleting the identical words and stop words respectively, we using the Chinese-Word-Vector model [38] to obtain corresponding word vector bags (lines 22–23). Then function `sim` (lines 26–31) is called to calculate the similarity value of two given word vector bags. Function `sim` calculates the maximum cosine similarity of every word in $vBag_1$ with all words in $vBag_2$, and then returns the minimum value among them. The intuitive explanation is for every word in $vBag_1$, find a word in $vBag_2$ that has the largest cosine similarity (highest semantic similarity), and then pick the smallest cosine value among all values calculated for words in $vBag_1$. If the smallest cosine value is still larger than a given threshold, it means that all the words in $vBag_1$ is able to find a semantic similar word in $vBag_2$. Note that function sim has directions and thus we need to calculate for both directions to choose the smaller ones to get the similarity score (line 24). If the score is bigger than θ_{bow}, we consider that both vector bags are similar and thus return true (line 25). The similarity threshold of our bag-of-words model is set to 0.45 empirically.

TABLE II
STATISTICS OF THE OUR DATASET

Corpus	# Words/ Sentence	Average Words	Words	
			Total	Distinct
Business	13-37	23	2,349	1079
Culture	15-38	24	2,412	1,096
Entertainment	14-36	23	2,354	1,061
Health	15-35	22	2,232	850
Political	15-35	26	2,597	1,199
Sports	15-35	24	2,392	1,110
Technology	16-41	25	2,553	1,159
Travel	15-36	23	2,362	1,115

IV. EVALUATIONS

A. Experiment Setting

We implement our approach in Python. The evaluation is conducted on a laptop with Intel(R) Corei7 CPU, 16GB memory. The deep learning models, i.e., Word Alignment model AWESOME and the sentence compression model SLAHAN, run on a server with 512GB memory and an Nvidia 2080RTX GPU.

We adopt an open-source implementation of the sentence representation model SimCSE [23], which is fine-tuned for Chinese sentence representations. For word alignment model, we adopt awesome-align [27] which is fine-tuned by ZH-EN dataset. Note that since the tested machine translators produce consistent translations for multiple runs of the same test input, therefore, we invoke each machine translator once for one source sentence.

Dataset We collect unlabelled sentences from CNN articles to form our dataset. In particular, we crawled 60 articles of 8 different categories, i.e., business, culture, entertainment, health, political, sports, technology and travel, from the CNN official website [39]. Then we select grammatically and semantically correct sentences from each category to form a sentence list of the corresponding category. After that, we randomly select 100 sentences from the sentence list of each category to form our dataset, which consist 800 sentences. We tentatively select 8 categories to evaluate whether DCS consistently performs well in different semantic context. Note that our dataset does not have any intersection with the datasets used in existing approaches [11], [12], [15], [22], which ensures the fairness comparison with existing approaches from the data perspective. The statistics of the sentences in our dataset are shown in Table II.

We also adopt the datasets released by CIT [12] and SIT [11] to evaluate our approach and the compared approaches. The dataset of CIT consists of 600 grammatically and semantically correct source sentences collected from China Daily, BBC and CNN news. The dataset of SIT consists of 200 source sentences extracted from CNN news.

Compared Method We compare our approach DCS with two state-of-the-art approaches CIT [12] and CAT [22]. CIT employs the constituency invariance relation and construct sentence pairs by adding one adjunct part to the base sentence. We also compare with CAT, which is a parallel work of CIT and conducts single-word replacement. We follow the links provided in their papers and adopt the open source model released. We use the hyper-parameters reported in the corresponding papers of the compared methods. Please note that, CIT [12] and CAT [22] have separately shown to outperform existing approaches, including SIT [11], Purity [14] and TransRepair [15]. Therefore, we only compare with CIT and CAT, which represent state-of-the-art results.

Testing Targets We select three most widely used translation software, i.e., Google Translate, Microsoft Bing Translate and Baidu Translate, and target the English-Chinese translation engine as our testing target systems. The three translators are all popular commercial translation software and have large international and domestic user base. They all provide Web interface as well as APIs to enable users and developers to acquire instant translation results. In our work, we directly call the provided APIs by each translation software, i.e., Google Cloud Translation API for Google [24], Azure Microsoft Translator API for Bing [25] and Baidu Translate API [26] for Baidu, to obtain translated sentences. Note that we take the English-Chinese translation engine as the testing target due to data labelling and evaluation convenience.

Ground Truth Labelling We invited 5 graduate students, who are native Chinese speakers and have more than 10 years experience of leaning and writing in English. To avoid the bias that one may have on particular subjects, we require three volunteers to label each translated sentence individually. To simplify the labelling task, we automatically generate translated sentence pairs (in Chinese) and ask the volunteers to label "True" or "False", where "True" means the given two sentences have different semantics, and "False" means they have the same semantics. A tutorial is provided to educate on the labeling process. The rule of "majority win" is adopted to merge the labels. If all three volunteers produce consistent labels, we directly adopt the label. If there are discrepancies, we invite all three volunteers to discuss and decide an agreed label. Due to the large number of mutated sentences generated by the compared approaches [12], [22], we are not able to label all of them. Therefore, we randomly select 800 sentence pairs generated by each approach on each machine translator. In total, 7200 pairs (3 tested machine translators and 3 compared approaches) of sentences are manually labelled.

Evaluation Metrics We adopt the precision (P), recall (R) and F1-score (F) as the evaluation metrics for effectiveness following the experiment settings of existing work [22]. The formulas used to calculate the metrics are shown in Formulas (1) $\sim$ Formulas (3), where TP represents the number of sentence pairs correctly identified as positive, FN represents the number of sentence pairs incorrectly identified as negative, FP represents the number of sentence pairs incorrectly identified as positive.

$$P = \frac{TP}{TP + FP} \tag{1}$$

$$R = \frac{TP}{TP + FN} \tag{2}$$

$$F = \frac{2 * P * R}{P + R} \tag{3}$$

TABLE III
THE EFFECTIVENESS COMPARISON OF DCS, CIT AND CAT

Approach			Business	Culture	Entertainment	Health	Political	Sports	Technology	Travel	Average
Baidu	DCS	P	**0.79**	**0.77**	**0.81**	**0.76**	**0.85**	**0.83**	**0.83**	**0.82**	**0.81**
		R	**0.82**	**0.75**	**0.9**	**0.74**	**0.86**	**0.86**	**0.94**	**0.8**	**0.84**
		F	**0.8**	**0.76**	**0.86**	**0.75**	**0.85**	**0.86**	**0.89**	**0.81**	**0.83**
	CIT	P	0.61	0.75	0.67	0.64	0.73	0.77	0.75	0.62	0.7
		R	0.39	0.5	0.25	0.31	0.51	0.55	0.26	0.67	0.41
		F	0.48	0.6	0.36	0.42	0.6	0.64	0.39	0.64	0.51
	CAT	P	0.42	0.6	0.58	0.58	0.57	0.47	0.43	0.4	0.51
		R	0.75	0.7	0.68	0.72	0.77	0.8	0.64	0.67	0.7
		F	0.55	0.65	0.63	0.64	0.65	0.59	0.52	0.5	0.6
Google	DCS	P	**0.79**	**0.77**	**0.79**	**0.75**	**0.78**	**0.88**	0.84	**0.85**	**0.82**
		R	0.82	0.75	**0.85**	**0.79**	**0.78**	**0.85**	**0.87**	**0.85**	**0.82**
		F	**0.8**	**0.76**	**0.82**	**0.77**	**0.78**	**0.87**	**0.86**	**0.85**	**0.82**
	CIT	P	0.61	0.68	0.5	0.64	0.73	0.58	**0.93**	0.56	0.65
		R	0.61	0.41	0.6	0.33	0.62	0.54	0.34	0.56	0.47
		F	0.61	0.52	0.55	0.44	0.67	0.56	0.5	0.56	0.55
	CAT	P	0.33	0.55	0.58	0.45	0.49	0.48	0.35	0.43	0.46
		R	**0.83**	**0.79**	0.81	0.69	0.74	0.84	0.75	0.8	0.76
		F	0.48	0.66	0.68	0.55	0.59	0.61	0.48	0.56	0.58
Bing	DCS	P	**0.78**	**0.81**	**0.79**	**0.75**	**0.75**	**0.86**	0.81	**0.82**	**0.8**
		R	0.79	0.75	**0.85**	**0.79**	**0.82**	**0.84**	**0.87**	**0.82**	**0.83**
		F	**0.78**	**0.78**	**0.82**	**0.77**	**0.78**	**0.85**	**0.84**	**0.82**	**0.82**
	CIT	P	0.64	0.76	0.79	0.73	0.53	0.71	**0.82**	0.79	0.72
		R	0.39	0.57	0.2	0.35	0.53	0.44	0.5	0.61	0.41
		F	0.48	0.65	0.32	0.47	0.53	0.54	0.62	0.69	0.52
	CAT	P	0.63	0.51	0.66	0.51	0.59	0.55	0.62	0.4	0.55
		R	**0.86**	**0.77**	0.62	0.79	0.69	0.83	0.74	0.73	0.75
		F	0.73	0.61	0.64	0.63	0.63	0.67	0.69	0.52	0.64

Existing approaches only measure the inconsistent sentence pairs detected, and fail to notice that there could be multiple inconsistent pairs pinpointing to the same wrongly translated phrase. To quantitatively measure how the detected issues help in identifying unique errors, which can assist localization and fixing of bugs, we define a new metric named the redundancy level. Formula (4) provides the formal definition of redundancy level, which is defined as the average number of detected issues that point to a unique error in the source sentence. In Formula (4), |R| is the number of inconsistent sentence pairs reported by the testing method, |U| is the unique number of errors in R.

$$Redundancy\_level = \frac{|R|}{|U|} \tag{4}$$

For instance, in the example shown in Fig. 1(b), |R| is 2 and the number of unique errors is 1. Therefore, the redundancy level is 2.

B. Research Questions

In the evaluation, we would like to answer the following research questions.

- RQ1: How effective is DCS in detecting translation errors?
- RQ2: How can DCS be used in complement with existing detecting tools.
- RQ3: How does principle of compositionality contribute to the error detecting effectiveness?
- RQ4: What are the False Positives and False Negatives detected by DCS like?
- RQ5: How efficient is DCS in detecting translation errors?

C. Experiment Results

1) RQ1: Effectiveness: To measure the effectiveness of DCS, we conduct experiments with our dataset on 3 different machine translators and compare with 2 state-of-the-art approaches. The detailed evaluation results are shown in Table III. From the table, we can observe that DCS achieves consistently good performance on all three machine translation software and all sentence categories. The average precision of DCS is 81%, which outperforms both CIT (69%) and CAT (51%). The average recall of DCS is 83%, which is also much higher than that of CIT (43%) and CAT (74%).

We manually analyze the issues reported by DCS and report the number of errors that appear in each tested translate software on each error category. The results are shown in Table V. Recall that we have 800 source sentences and 3 translation software under test. Therefore, there are a total of 2,400 translated sentence pairs since there are three combinations from the three translation software. Through our manual inspection, 509 errors reported by DCS are true positives. We can observe that the wrong choice of words (WCW) causes the most number of translation errors, followed by under translation (UT) and unclear logic (UL). Among the three tested translation software, Bing Translate has the lowest number of errors, while Baidu Translate has the highest number of errors. DCS is able to detect all types of translation errors.

We also compare our approach with state-of-the-art approaches on datasets released by existing approaches, i.e., SIT [11] and CIT [12]. The results are shown in Table IV. From the table, we can observe that the results show consistent trend with that reported on our dataset. DCS outperforms both compared approaches on the datasets of both CIT and SIT.

TABLE IV
THE EFFECTIVENESS COMPARISON OF DCS, CIT AND CAT ON EXISTING DATASETS

Approach			CIT Dataset						SIT Dataset		Average
			Business	Culture	Health	Political	Sports	Travel	Business	Politcial	
Baidu	DCS	P	0.81	**0.77**	**0.77**	0.73	**0.80**	**0.72**	**0.74**	**0.77**	**0.76**
		R	**0.88**	**0.77**	**0.87**	0.76	**0.84**	**0.78**	0.74	**0.81**	**0.81**
		F	**0.84**	**0.77**	**0.82**	**0.74**	**0.82**	**0.75**	**0.74**	**0.79**	**0.78**
	CIT	P	**0.86**	0.73	0.76	**0.77**	0.69	0.63	0.68	0.69	0.73
		R	0.43	0.40	0.53	0.38	0.42	0.48	0.45	0.35	0.43
		F	0.57	0.52	0.63	0.51	0.52	0.55	0.55	0.46	0.54
	CAT	P	0.54	0.64	0.51	0.42	0.43	0.48	0.51	0.53	0.51
		R	0.77	0.76	0.70	**0.78**	0.70	**0.70**	**0.77**	0.80	0.75
		F	0.63	0.69	0.59	0.55	0.54	0.57	0.62	0.63	0.60
Google	DCS	P	**0.79**	**0.76**	0.77	**0.78**	0.76	**0.74**	0.71	0.73	**0.76**
		R	0.79	**0.78**	**0.79**	**0.74**	0.80	**0.81**	0.70	0.74	**0.77**
		F	**0.79**	**0.77**	**0.78**	**0.76**	**0.78**	**0.77**	**0.70**	**0.73**	**0.76**
	CIT	P	0.70	0.75	**0.80**	0.71	**0.77**	0.67	**0.75**	**0.78**	0.74
		R	0.41	0.43	0.36	0.33	0.63	0.57	0.48	0.64	0.48
		F	0.52	0.55	0.50	0.45	0.69	0.62	0.59	0.70	0.58
	CAT	P	0.50	0.59	0.48	0.58	0.42	0.39	0.56	0.48	0.50
		R	**0.80**	0.69	0.63	0.74	**0.81**	0.80	**0.74**	**0.81**	0.75
		F	0.62	0.64	0.54	0.65	0.55	0.53	0.64	0.60	0.59
Bing	DCS	P	**0.81**	**0.72**	0.72	**0.69**	**0.78**	**0.71**	0.72	**0.81**	**0.75**
		R	**0.83**	0.78	**0.75**	0.67	**0.79**	**0.71**	**0.77**	**0.78**	**0.77**
		F	**0.82**	**0.75**	**0.73**	**0.68**	**0.78**	**0.71**	**0.74**	**0.79**	**0.76**
	CIT	P	0.77	0.71	**0.75**	0.67	0.78	0.71	**0.75**	0.79	0.74
		R	0.50	0.40	0.43	0.40	0.39	0.44	0.58	0.46	0.45
		F	0.61	0.51	0.55	0.50	0.52	0.54	0.65	0.58	0.56
	CAT	P	0.56	0.64	0.54	0.41	0.52	0.40	0.57	0.45	0.51
		R	0.78	**0.79**	0.72	**0.73**	0.75	0.68	0.77	0.78	0.75
		F	0.65	0.71	0.62	0.52	0.61	0.51	0.65	0.57	0.61

TABLE V
NUMBER OF ERRORS REPORTED IN EACH CATEGORY BY DCS

	WCW	UT	UL	Q/T-P	NEP	OM	OTS	Total
Bing	91	27	7	4	10	2	13	154
Google	98	38	8	1	11	1	6	163
Baidu	85	64	7	4	17	2	13	192
Total	274	129	22	9	38	5	32	509

TABLE VI
REDUNDANCY LEVEL OF THE COMPARED APPROACHES

	DCS	CIT	CAT
Bing	1.09	4.32	1.18
Google	1.04	5.26	1.16
Baidu	1.06	4.25	1.17
Average	1.06	4.64	1.17

Redundancy level provides a quantitative way to measure the information each detected inconsistent sentence pair can provide, it can also be used to measure the manual checking and analysis efforts that testers require. The results of redundancy level of the tree compared approaches are shown in Table VI, CIT's *redundancy_level* is around 4, which means that an average of around 4 sentence pair issues reported by CIT refers to the same error, and thus testers manually check 4 redundant reports and obtain one unique information. And for DCS and CAT, the redundancy levels are 1.06 and 1.17, respectively, which are far smaller than CIT.

Answer to RQ1: DCS outperforms both CIT and CAT on all tested translate engines in terms of detection precision and recall. DCS is capable of detecting all types of translation errors and has the lowest redundancy level.

TABLE VII
THE COMPLEMENT EXPERIMENT RESULT

Corpus	Approach		P	R	F1
CIT_Gen	DCS	Baidu	0.82	0.75	0.78
		Google	0.82	0.67	0.75
		Bing	0.85	0.67	0.74
	CIT	Baidu	0.64	0.4	0.49
		Google	0.69	0.39	0.49
		Bing	0.67	0.43	0.52
CAT_Gen	DCS	Baidu	0.7	0.68	0.69
		Google	0.7	0.66	0.68
		Bing	0.73	0.67	0.7
	CAT	Baidu	0.54	0.63	0.58
		Google	0.48	0.6	0.54
		Bing	0.56	0.62	0.58

2) RQ2: Complement With Existing Work: To demonstrate how DCS can be used in complement with compared approaches, we use the test cases generated by CIT and CAT and then use DCS to detect errors on the generated test cases. In particular, we randomly selected 200 sentence pairs from the sentence pairs generated by CIT and CAT, respectively, and both the source and variant sentences in the sentence pairs were used as our test inputs. In Table VII, CIT_Gen is the dataset containing sentence pairs generated by CIT, and CAT_Gen is the dataset containing sentence pairs generated by CAT. We compare DCS with CIT on the test cases generated by CIT and similarly compare DCS with CAT on the test cases generated by CAT, and report the precision, recall and F1-value on all three tested translate engines.

We can observe that DCS achieves better results on all evaluated metrics on all tested translate engines than both CIT and CAT. The results indicate that DCS is a more effective oracle to

TABLE VIII
THE COMPARISON BETWEEN DCS AND DCS WITHOUT COMPOSITIONALITY

Approach		TN	FN	TP	FP	P	R	F
DCS	Baidu	938	88	466	108	0.81	0.84	0.83
	Google	987	85	427	91	0.82	0.82	0.82
	Bing	971	89	435	105	0.81	0.83	0.82
DCS -comp	Baidu	1005	413	141	41	0.77	0.25	0.39
	Google	1038	400	112	40	0.74	0.22	0.34
	Bing	1035	399	125	41	0.75	0.24	0.37

detect translation errors, and it can be used in complement with existing test case generation methods to achieve better error detection result.

Answer to RQ2: DCS constructs a more effective oracle to detect translation errors, and it can be used in complement with existing test case generation methods to achieve better error detection result.

3) RQ3: Effectiveness of Principle of Compositionality: To evaluate the effect of principle of compositionality on detection accuracy improvement, we conduct an experiment to compare DCS and DCS without the compositional semantics (i.e., DCS-comp), which removes the steps of sentence partition, word alignment and part alignment from DCS. Table VIII shows the evaluation result. We can observe that DCS achieves better precision and recall on all tested translate engines. The increment of recall is around 60% for all tested translate engines. The reason is that DCS greatly reduces the number of false negatives, i.e., by more than 4 times, and increases the number of true positives by more than 3 times on all tested translate engines. This is mainly due to the reason that only relying on semantic models to encode the long sentences may still be affected by the issue of "collapse", which makes it hard to precisely measure the similarity of the encoded vectors. With the principle of compositionality, this issue is very much relieved since encoding on each compositional part of a sentence is more accurate.

Answer to RQ3: DCS outperforms DCS-comp on all tested translation engines on all measured metrics, which shows that the principle of compositionality contributes to accurately identifying translation errors.

4) RQ4: Case Study of False Positives and False Negatives: We carefully analyzed the results reported by all three approaches. In particular, we randomly sampled 150 false positives and 150 false negatives for each tested translator to check, and then manually analyze the reasons of the errors.

False Positives are summarized in Table IX. DCS reports 150 false positive pairs in total, where 105 are related to Microsoft Bing Translate, 91 are related to Google Translate and 104 are related to Baidu Translate.[4] The main reasons for the false positives are summarized as follows.

[4]Recall that one false positive pair is related to two translate engines.

- **Synonym.** The semantics of synonyms (words or phrases) in the translated sentence are not accurately represented and not captured in the dictionary, which results in a low similarity score on the translated sentences and thus causes the false positive. For instance, in the first case study shown in Table IX, the phrase "Begrudgingly" is translated to "勉强" ("begrudgingly") by Baidu Translate, and to "不情愿地" ("begrudgingly") in Google Translate. Both translations are correct, yet this synonym semantic is not correctly captured and thus results in false positives. We detect 65 false positives, which counts for 43% of all false positives by DCS, due to this reason.
- **Word Alignment Error.** The incorrect word alignment between English and Chinese causes 32% of the false positives. For instance, in Table IX, "indescribable" was not correctly aligned with "无法形容" ("indescribable") in the translated sentence by Baidu Translate, and thus the aligned Chinese sub-phrases for "is something indescribable" are "是一件的事情" (which means "it is a thing") (by Baidu Translate) and "是难以形容的事情" (which means "it is an indescribable thing") (by Google Translate), respectively. The similarity of the semantic representation of the two Chinese sub-phrases is lower than the given threshold, which leads to a false positive. We detect 48 false positives due to this reason.
- **Structure Difference.** The translated sentences have different syntactical structures, which leads to different aligned parts, and thus semantic similarity by our metrics. Take the case study in Table IX as an example, the clause "which is gaining increasing global influence in recent years" is translated to an attribute phrase of "12th Beijing International Film Festival" by Bing Translate, yet is translated as clause of "12th Beijing International Film Festival" by Baidu Translate. Both are correct syntactical structures and have the same semantics in Chinese, 'which' is translated to "该电影节" (which means "the film festival") by Baidu as pronoun, yet is reasonably omitted in Bing's translation, resulting in a high semantic difference on the translated Chinese parts. We detect 20 false positives due to this reason.
- **Polysemy Words.** The semantics of the Polysemy words are both reasonably correct without giving more context. For instance, in the fourth case study shown in Table IX, the word "lifting" is translated into "好起来" ("getting better") by Google Translate and to "提升" ("lifting") by Bing Translate, which are both reasonably correct in the given context. We detect 17 false positives due to this reason.

False Negatives are summarized in Table X. DCS reports 135 false negatives in total, among which 92 are related to Microsoft Bing Translate, 87 are related to Google Translate and 91 are related to Baidu Translate. The main reasons for the false positives are summarized as follows.

- **Inaccurate Semantic Representation.** The semantic representation by the language model is inaccurate, and thus the cosine similarity fails to measure the semantic similarity. For instance in the first case study of Table X,

TABLE IX
CASE STUDY OF FALSE POSITIVES OF DCS

Source sentence	Translated sentences	Meaning of translated sentences	Reason
Begrudgingly, Carrie agreed, and made her way to the edge of the rock to sit down next to the stranger.	**Baidu:** 嘉莉勉强同意了，走到岩石边缘，在陌生人旁边坐下。 **Bing:** 嘉莉不情愿地同意了，她走到岩石边，坐在陌生人旁边。	**Baidu:** Carrie begrudgingly agreed, walked to the edge of the rock and sat down next to the stranger. **Bing:** Carrie begrudgingly agreed, and she walked over to the rock and sat next to the stranger.	Synonym 43%
Right now is something **indescribable**, I would say, because there is a parent of one tennis player that died.	**Baidu:** 我想说，现在是一件无法形容的事情，因为有一位网球运动员的父母去世了。 **Google:** 我想说，现在是难以形容的事情，因为有一名网球运动员的父母去世了。	**Baidu:** I would say that it is now an indescribable thing because there was a tennis player's parents who died. **Google:** I would say that it is now an indescribable thing because there was a tennis player's parents who died.	Word Alignment Error 32%
12th Beijing International Film Festival, **which is gaining increasing global influence in recent years**, will be held in Beijing from Aug 13 to 20.	**Baidu:** 第十二届北京国际电影节将于8月13日至20日在北京举行。近年来，该电影节在全球的影响力与日俱增。 **Bing:** 近年来全球影响力日益增强的第12届北京国际电影节将于8月13日至20日在北京举行。	**Baidu:** The 12th Beijing International Film Festival will be held in Beijing from August 13 to 20. In recent years, the festival is gaining increasing global influence.. **Bing:** The 12th Beijing International Film Festival, which is gaining increasing global influence in recent years, will be held in Beijing from August 13 to 20.	Structure Difference 13%
I could feel my mood **lifting**, my mind clearing and my body loosening.	**Google:** 我能感觉到我的心情好起来，我的头脑清醒了，我的身体放松了。 **Bing:** 我能感觉到我的情绪在提升，我的思想在清理，我的身体在放松。	**Google:** I could feel my mood getting better, my mind clearing, my body loosening. **Bing:** I can feel my mood lifting, my mind clearing, my body loosening.	Polysemy 12%

TABLE X
CASE STUDY OF FALSE NEGATIVES OF DCS

Source sentence	Translated sentence	Meaning of translated sentence	Reason	Error Type
From sun and flowers to **scenes of people everywhere**, the paintings mirrored the unique charm and beauty of the lifestyle.	**Baidu:** 从阳光和鲜花到随处可见的人们的场景，这些绘画反映了生活方式的独特魅力和美丽。 **Bing:** 从阳光、鲜花到人迹罕至的场景，这些画作反映了生活方式的独特魅力和美感。	**Baidu:** From sunlight and flowers to scenes of people everywhere, the paintings reflect the unique charm and beauty of the lifestyle. **Bing:** From sunlight and flowers to off-the-beaten-path scenes, these paintings reflect the unique charm and beauty of the lifestyle.	Inaccurate Semantic Representation 60%	Opposite Meaning
"They made me feel small; they made me feel stupid and embarrassed even just asking the question," she said at an April 2021 town meetings, **which she quoted**.	**Baidu:** 她在2021年4月的一次城镇会议上说："他们让我觉得自己很渺小；他们让我感到愚蠢和尴尬，即使只是问这个问题." **Google:** 他们让我感到渺小;他们让我感到愚蠢和尴尬，即使只是问这个问题，"她在2021年4月的城镇会议上说。	**Baidu:** At an April 2021 town meeting, she said: "They made me feel small; they made me feel stupid and embarrassed, even if it was just asking the question." **Google:** They make me feel small; "they made me feel stupid and embarrassed, even if it was just asking the question," she said at an April 2021 town meeting.	Same Error 40%	Under Translation

the sentence translated by Bing Translate has an opposite meaning on the phrase "scenes of people everywhere", which is an opposite meaning error. We encountered 81 false negatives of this kind.

- **Same Error.** Two translated sentences have the same errors, e.g., due to under translation, and thus have high semantic similarity. For instance in the second case study of Table X, both translated sentence by Baidu Translate and Google Translate fails to translate the phrase "which she quoted", and this cannot be detected by difference testing based approach. We have 54 false negatives of this type.

Answer to RQ4: Most false positives reported by DCS are due to missed synonyms and word alignment errors. The false negatives are due to inaccurate semantic representation and same error in the translated sentences.

5) RQ5: Efficiency: To measure the error detection efficiency of DCS and the compared approaches, we collect and calculate the time required to process one pair of sentences. To have a clear view of the offline preparation and online processing time, we report three types of time, i.e., preprocessing time, mutation time, and bug detection time. The preprocessing time refers to those processes which do not require the testing targets, i.e., machine translators, and thus can be conducted offline. For both DCS and CIT, the preprocessing time consists of the time for sentence compression and constituency parsing. No preprocessing is required for CAT. The mutation time refers to the time for mutant generation, i.e., the time of replacing words on the adjunct part using Bert [31] for CIT and the time of replacing words on the whole sentence using Bert for CAT. The bug detection time refers to the time of comparing the translated sentences to detect potential bugs. In particular, the bug detection time includes the word alignment, part alignment as well as

TABLE XI
THE AVERAGE EXECUTION TIME (SECONDS) OF COMPARED METHODS

	DCS	CIT	CAT
Preprocessing	0.49	0.49	-
Mutation	-	0.1	0.01
Bug detection	0.25	0.19	0.03

semantic similarity analysis for DCS; checking the translation's correctness by using constituency structure comparison for CIT; and using LCS to compute the similarity of the sentence pairs for CAT. Note that we do not report the time to invoke the testing targets as it is the same for all three approaches.

The results are shown in Table XI. CAT is the most efficient as it does not invoke heavy models like sentence compression and constituency parsing, which both CIT and DCS adopted. Our approach has a higher bug detection time, due to the reason that we need to load the word alignment and sentence representation model, which are time consuming. However, considering the redundancy level of detected errors, DCS is better than CIT in detecting non-redundant errors.

Answer to RQ5: DCS requires the same preprocessing time as CIT and slightly higher bug detection time than CIT. CAT has the lowest bug detection time. However, considering the redundancy level of detected errors, DCS is comparable with baseline approaches in detecting non-redundant errors.

D. Threats to Validity

Threats to internal validity may come from the implementations of our artifact. To eliminate the threats, we conduct manual checking of the programs we coded and run tests on the programs to detect potential bugs. The labeling of the ground truth sentence pairs may also cause threats to validity. To mitigate the threats, we ask three graduate students to label the same sentence pairs independently, and then merge the labeled results. Discussions will be conducted to solve inconsistencies if any.

Threats to external validity may be due to the adoption of the various models for sentence compression, constituency parsing and word alignment. To mitigate the treats, we adopt the projects that are widely adopted, e.g., Stanford Parser. Moreover, we conduct manual checking on the results produced by each model to ensure the correctness of the adopted models. The accuracy of the adopted models, e.g., the word alignment model, do affect the accuracy of DCS. Our case study shows that there are 32% of false positives caused by word alignment errors. However, even with the errors introduced by the word alignment model, DCS still outperforms state-of-the-art approaches. We can expect that a more accurate word alignment model can further improve the performance of DCS, yet this is out of the scope of this work. For the compared methods, we obtain the source code released by the authors for CAT, and re-implement the algorithm of CIT follow their paper due to unavailability of the source code. For both approaches, we try to reproduce their experiment results reported in the paper first to mitigate potential threats. Another external threat is that we evaluate DCS on the English-Chinese translate engine restricted by the labelling efforts required, and whether the results hold for other target languages remain to be evaluated. However, the overall structure of DCS is general. There are mainly two components, i.e., word alignment and similarity comparison, which need to be configured or customized to apply DCS on other target languages. We adopt AWESOME [27], which is the state-of-the-art word alignment model supporting five language pairs. Word alignment is an actively researched topic in NLP and there are word alignment models available for different source-target language pairs [40], [41]. Therefore, DCS is applicable to those sentence pairs with available word alignment models. For the similarity comparison, we adopt pre-trained language model SimCSE [21], which is trained based on Bert_{base} and supports multiple languages, to obtain the embedding vector for the given sentence. There are also various other pre-trained language models [42], [43] available to conduct sentence representation for multiple languages, which can be adopted by DCS to support other target languages. The sentence partition component applies to the source language. We rely on sentence compression and constituency parsing to parse a sentence into the main part and the constituent parts. Stanford CoreNLP [28] supports six languages of constituency parsing and the sentence compression model can be trained for desired languages [29], which makes this component extensible to other source languages other than English.

Threats to construction validity may lie in the test datasets and hyper-parameters used in the models and the compared methods. Following existing approaches [12], [22], we evaluated our approach on three manually labelled datasets, consisting a total of 1600 source sentences. We carefully select the source sentences from the newest CNN articles, which consists 8 different categories, to increase the sentence diversity in the dataset. Nevertheless, the selected source sentences could have been 'seen' by the translation model as those models may be updated frequently, and thus the effectiveness of DCS on unseen data is yet to be evaluated. The hyper-parameters used follow the settings reported in the papers.

V. RELATED WORK

Machine Translation Testing With the continuous development of neural machine translation, the research work on improving the machine translation performance and testing techniques of machine translation has also been increasing.

Belinkov et al. [44] investigated methods to improve model robustness by using structurally invariant word representations and integrated training on different types of adversarial samples. Recently, many technologies are mostly based on metamorphic testing [11], [12], [13], [14], [15]. SIT [11] judges whether the translation is correct by replacing nouns and adjectives in a sentence with words of the same lexical nature and comparing whether the grammatical structures of the translation results are similar. TransRepair [15] check whether is correct by replacing nouns, adjectives, and quantifiers in a

sentence with semantically identical words and comparing the semantics similarity of their translation results. PatInv [13] replaces nouns and adjectives in a sentence with a word that have different semantics and then checks whether the semantics of the translations are different. CIT [12] checks whether the translation's constituency structure of simple sentence is included in the translation's constituency structure of its derived sentences which added the adjuncts parts. CAT [22] has rewritten the part of word replacement of TransRepair to extend the lexicality of substituted words from nouns, adjectives, and quantifiers to all types. And then, CAT also use BERT MLM to doing the mutating like SIT distinguished from TransRepair.

Differential Testing Differential testing, proposed by McKeeman [45] at first, as a regression testing method for large-scale software systems, and is lately widely adopted to find bugs in various fields, e.g., JavaScript engines [46], [47], Java virtual machine [48], SSL/TLS implementations [49], HTTP protocol [50], Java bytecode programs [51], quantum software stacks [52], and CPU [53]. Brubaker et al. [54] used differential testing with frankencerts finding significant security vulnerabilities on SSL/TLS implementations. Petsios et al. [55] designed NEZHA, an efficient differential testing framework whose input's format is agnostic. DLFuzz [56] constructs a differential fuzzing testing framework to find flaws of Deep Learning systems. We propose the first differential testing based approach to test machine translation software. Unlike the other testing targets, which have structured outputs and syntax or even text level comparison will identify the difference, machine translation software output natural language sentences, which require semantic level comparison. We propose the comparison methods based on pre-trained language models and synonym dictionary.

VI. Conclusion

In this work, we propose a differential testing method named DCS for machine translation software based on compositional semantics. Rather than directly compare the semantics of the translated sentences obtained from the two machine translate software, we employ the principle of compositionality, and compute the semantic similarity of the corresponding composition parts of the translated sentences. Word alignment and constituency parsing are used to obtain the corresponding composition parts. We compare our approach with state-of-the-art approaches CIT and CAT on three datasets. The experiment results show that DCS achieves higher precision, and produces less redundant issue pairs. Moreover, DCS can be used in complement with existing test case generation approaches to achieve better detection precision, and achieves comparable efficiency with baseline approaches.

References

[1] "The Google Translate engine," 2022. Accessed: Mar. 14, 2022. [Online]. Available: https://translate.google.com

[2] "The Bing Microsoft Translator," 2022. Accessed: Feb. 11, 2022. [Online]. Available: https://cn.bing.com/translator

[3] B. Turovsky, "Ten years of Google Translate," 2022. Accessed: Mar. 17, 2022. [Online]. Available: https://blog.google/products/translate/ten-years-of-google-translate/

[4] H. Hassan et al., "Achieving human parity on automatic Chinese to English news translation," 2018, *arXiv:1803.05567*.

[5] "Lost in translation: 13 international marketing fails," 2022. Accessed: Mar. 14, 2022. [Online]. Available: https://www.businessnewsdaily.com/5241-international-marketing-fails.html

[6] "Palestinian man is arrested by police after posting 'good morning' in Arabic on Facebook which was wrongly translated as 'attack them'," 2017. Accessed: Mar. 20, 2022. [Online]. Available: https://www.dailymail.co.uk/news/article-5005489/Good-morning-Facebook-post-leads-arrest-Palestinian.html

[7] "After Google Translate's latest update, BBC Culture finds history's biggest language mistakes—including a US president stating 'I desire the Poles carnally'." 2015. Accessed: May 20, 2022. [Online]. Available: https://www.bbc.com/culture/article/20150202-the-greatest-mistranslations-ever

[8] M. C. Mason, "Strategic insights: Lost in translation," 2017. Accessed: May 21, 2022. [Online]. Available: https://press.armywarcollege.edu/articles_editorials/401/

[9] H. Zhao and Q. Liu, "Common error analysis of machine translation output," in *Proc. 10th China Workshop Mach. Transl.*, Nov. 2014, pp. 129–158.

[10] D. Vilar, J. Xu, L. D'Haro, and H. Ney, "Error analysis of machine translation output," in *Proc. Int. Conf. Lang. Resour. Eval.*,, May 2006, pp. 697–702.

[11] P. He, C. Meister, and Z. Su, "Structure-invariant testing for machine translation," in *Proc. ACM/IEEE 42nd Int. Conf. Softw. Eng. (ICSE)*. New York, NY, USA: ACM, 2020, pp. 961–973. [Online]. Available: https://doi.org/10.1145/3377811.3380339

[12] P. Ji, Y. Feng, J. Liu, Z. Zhao, and B. Xu, "Automated testing for machine translation via constituency invariance," in *Proc. 36th IEEE/ACM Int. Conf. Automat. Softw. Eng. (ASE)*, 2021, pp. 468–479.

[13] S. Gupta, P. He, C. Meister, and Z. Su, *Machine Translation Testing via Pathological Invariance*. New York, NY, USA: ACM, 2020, pp. 863–875. [Online]. Available: https://doi.org/10.1145/3368089.3409756

[14] P. He, C. Meister, and Z. Su, "Testing machine translation via referential transparency," in *Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng. (ICSE)*, 2021, pp. 410–422.

[15] Z. Sun, J. M. Zhang, M. Harman, M. Papadakis, and L. Zhang, "Automatic testing and improvement of machine translation," in *Proc. ACM/IEEE 42nd Int. Conf. Softw. Eng. (ICSE)*. New York, NY, USA: ACM, 2020, pp. 974–985. [Online]. Available: https://doi.org/10.1145/3377811.3380420

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meeting Assoc. Comput. Ling. (ACL)*. Philadelphia, PA, USA: Association for Computational Linguistics, 2002, p. 311–318. [Online]. Available: https://doi.org/10.3115/1073083.1073135

[17] E. Ristad and P. Yianilos, "Learning string-edit distance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 5, pp. 522–532, May 1998.

[18] Y. Wu et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," Sep. 2016, *arXiv:1609.08144*.

[19] J. Hintikka, "A hundred years later: The rise and fall of Frege's influence in language theory," *Synthese*, vol. 59, no. 1, pp. 27–49, 1984. Accessed: Apr. 2, 2022. [Online]. Available: http://www.jstor.org/stable/20115984

[20] R. A. Hudson, "Constituency and dependency," vol. 18, no. 3-4, pp. 179–198, 1980. doi: 10.1515/ling.1980.18.3-4.179.

[21] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 6894–6910. Accessed: Aug. 12, 2022. [Online]. Available: https://aclanthology.org/2021.emnlp-main.552

[22] Z. Sun, M. J. Zhang, Y. Xiong, M. Harman, M. Papadakis, and L. Zhang, "Improving machine translation systems via isotopic replacement," in *Proc. ACM/IEEE 44nd Int. Conf. Softw. Eng. (ICSE)*. New York, NY, USA: ACM, 2022, pp. 1181–1192.

[23] "The Chinese sentence representation model," 2022. Accessed: Apr. 2, 2022. [Online]. Available: https://huggingface.co/cyclone/simcse-chinese-roberta-wwm-ext

[24] "Google Cloud Translation," 2022. Accessed: Jan. 27, 2022. [Online]. Available: https://cloud.google.com/translate

[25] "Azure Microsoft Translator," 2022. Accessed: Jan. 27, 2022. [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/translator/
[26] "Baidu translator API," 2022. Accessed: Jan. 27, 2022. [Online]. Available: https://api.fanyi.baidu.com/
[27] Z.-Y. Dou and G. Neubig, "Word alignment by fine-tuning embeddings on parallel corpora," Jan. 2021, pp. 2112–2128.
[28] "Stanford Parser," 2022. Accessed: June 24, 2022. [Online]. Available: https://nlp.stanford.edu/software/lex-parser.shtml
[29] H. Kamigaito and M. Okumura, "Syntactically look-ahead attention network for sentence compression," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 05, 2020, pp. 8050–8057.
[30] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain: Association for Computational Linguistics, 2004, pp. 74–81.
[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[32] S. Liu, R. Guo, B. Zhao, T. Chen, and M. Zhang, "APPCorp: A corpus for android privacy policy document structure analysis," 2020, *arXiv:abs/2005.06945*.
[33] Y. Yan, R. Li, S. Wang, F. Zhang, W. Wu, and W. Xu, "ConSERT: A contrastive framework for self-supervised sentence representation transfer," in *Proc. 59th Annu. Meet. Assoc. Comput. Ling. 11th Int. Joint Conf. Natural Lang. Process. (vol. 1: Long Papers).* [Online]. Association for Computational Linguistics, Aug. 2021, pp. 5065–5075. Accessed: May 25, 2022. [Online]. Available: https://aclanthology.org/2021.acl-long.393
[34] Z. Liu, Y. Lin, and M. Sun, *Compositional Semantics*. Singapore: Springer, 2020, pp. 43–57, https://doi.org/10.1007/978-981-15-5573-2_3
[35] F. J. Pelletier, "The principle of semantic compositionality," *Topoi*, vol. 13, no. 1, pp. 11–24, 1994.
[36] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.
[37] B. Turovsky, "HIT IR-lab Tongyici Cilin," 2022. Accessed: Apr. 20, 2022. [Online]. Available: http://ir.hit.edu.cn/demo/ltp/Sharing_Plan.html
[38] S. Li, Z. Zhao, R. Hu, W. Li, T. Liu, and X. Du, "Analogical reasoning on Chinese morphological and semantic relations," 2018, *arXiv:1805.06504*.
[39] "Cable news network," 2022. Accessed: Nov. 5, 2022. [Online]. Available: https://edition.cnn.com/
[40] M. J. Sabet, P. Dufter, F. Yvon, and H. Schütze, "SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings," 2020, *arXiv:2004.08728*.
[41] Q. Liu, D. McCarthy, I. Vulić, and A. Korhonen, "Investigating cross-lingual alignment methods for contextualized embeddings with token-level evaluation," in *Proc. 23rd Conf. Comput. Natural Lang. Learn. (CoNLL)*, 2019, pp. 33–43.
[42] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, and W. Wang, "Language-agnostic BERT sentence embedding," 2020, *arXiv:2007.01852*.
[43] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," 2019, *arXiv:1907.11692*.
[44] Y. Belinkov and Y. Bisk, "Synthetic and natural noise both break neural machine translation," 2017, *arXiv:1711.02173*.
[45] W. M. McKeeman, "Differential testing for software," *Digit. Tech. J.*, vol. 10, no. 1, pp. 100–107, 1998. Accessed: June 10, 2022. [Online]. Available: http://www.hpl.hp.com/hpjournal/dtj/vol10num1/vol10num1art9.pdf
[46] G. Ye et al., "Automated conformance testing for JavaScript engines via deep compiler fuzzing," in *Proc. 42nd ACM SIGPLAN Int. Conf. Program. Lang. Des. Implementation*, 2021, pp. 435–450.
[47] J. Park, S. An, D. Youn, G. Kim, and S. Ryu, "JEST: N+1-version differential testing of both JavaScript engines and specification," in *Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng. (ICSE)*. Piscataway, NJ, USA: IEEE, 2021, pp. 13–24.
[48] Y. Chen, T. Su, C. Sun, Z. Su, and J. Zhao, "Coverage-directed differential testing of JVM implementations," in *Proc. 37th ACM SIGPLAN Conf. Program. Lang. Des. Implementation*, 2016, pp. 85–99.
[49] L. Quan et al., "SADT: Syntax-aware differential testing of certificate validation in SSL/TLS implementations," in *Proc. 35th IEEE/ACM Int. Conf. Automat. Softw. Eng. (ASE)*. Piscataway, NJ, USA: IEEE, 2020, pp. 524–535.
[50] B. Jabiyev, S. Sprecher, K. Onarlioglu, and E. Kirda, "T-Reqs: HTTP request smuggling with differential fuzzing," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2021, pp. 1805–1820.
[51] Y. Noller, C. S. Păsăreanu, M. Böhme, Y. Sun, H. L. Nguyen, and L. Grunske, "HyDiff: Hybrid differential software analysis," in *Proc. IEEE/ACM 42nd Int. Conf. Softw. Eng. (ICSE)*. Piscataway, NJ, USA: IEEE, 2020, pp. 1273–1285.
[52] J. Wang, Q. Zhang, G. H. Xu, and M. Kim, "QDiff: Differential testing of quantum software stacks," in *Proc. 36nd IEEE/ACM Int. Conf. Automat. Softw. Eng. (ASE)*, 2021, pp. 692–704.
[53] J. Hur, S. Song, D. Kwon, E. Baek, J. Kim, and B. Lee, "DifuzzRTL: Differential fuzz testing to find CPU bugs," in *Proc. IEEE Symp. Secur. Privacy (SP)*. Piscataway, NJ, USA: IEEE, 2021, pp. 1286–1303.
[54] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and V. Shmatikov, "Using frankencerts for automated adversarial testing of certificate validation in SSL/TLS implementations," in *Proc. IEEE Symp. Security Privacy*, vol. 2014, Nov. 2014, pp. 114–129.
[55] T. Petsios, A. Tang, S. Stolfo, A. D. Keromytis, and S. Jana, "NEZHA: Efficient domain-independent differential testing," in *Proc. IEEE Symp. Secur. Privacy (SP)*, 2017, pp. 615–632.
[56] J. Guo, Y. Jiang, Y. Zhao, Q. Chen, and J. Sun, "DLFuzz: Differential fuzzing testing of deep learning systems," in *Proc. 26th ACM Joint Meet. Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng. (ESEC/FSE)*. New York, NY, USA: ACM, 2018, pp. 739–743. [Online]. Available: https://doi.org/10.1145/3236024.3264835

Shuang Liu (Member, IEEE) received the bachelor's degree from Renmin University of China, in 2010, and the Ph.D. degree from the National University of Singapore, in 2015. She worked as a Research Fellow with SUTD, a lecturer with SiT, and an Associate Professor with Tianjin University, China. Her research interests include software testing, formal methods, and AI for SE.

Shujie Dou received bachelor's and master's degrees from Tianjin University, China, in 2020 and 2023, respectively. During the postgraduate education, he mainly engaged in the research on machine translation and automated testing.

Junjie Chen received the Ph.D. degree from Peking University, China, in 2019. He is an Associate Professor with the College of Intelligence and Computing, Tianjin University. His main research interests include software testing, SE for AI, and AI for SE.

Zhirun Zhang received the B.S. degree from Nankai University, Tianjin, China, in 2022. He is now working toward a graduate degree with the School of New Media and Communication at Tianjin University. His main research interests include natural language processing and Q&A system testing.

Ye Lu received the B.S. and Ph.D. degrees from Nankai University, Tianjin, China, in 2010 and 2015, respectively. He is an Associate Professor with the College of Cyber Science, Nankai University. His main research interests include FPGA accelerator, LLM, blockchain virtual machine, and embedded system.