Programming Language Representation with Semantic-level Structure

Anonymous Author(s)

ABSTRACT

Natural language processing (NLP) technique becomes one of the core techniques for developing text analytics applications. For developing an NLP application, the application is required to achieve high reliability before it goes to market. The trustworthiness of the prevalent NLP applications is obtained by measuring the accuracy of the applications on hold-out dataset. However, evaluating NLP on testset does with hold-out accuracy is limited to show its quality because the held-out datasets are often not comprehensive.

While the behavioral testing over multiple general linguistic capabilities are employed, the testing relies on manually created test cases, and is still limited to measure its comprehensive performance for each linguistic capability. In this work, we introduce Auto-CHECKLIST, an NLP model testing methodology. Given a linguistic capability, the Auto-CHECKLIST finds relevant testcases to test the linguistic capability from existing datasets as seed inputs, generates sufficient number of new test cases by fuzzing the seed inputs based on their context-free grammar (CFG). We illustrate the usefulness of the Auto-CHECKLIST by showing input diversity and identifying critical failures in state-of-the-art models for NLP task. In our experiment, we show that the Auto-CHECKLIST generates more test cases with higher diversity, and finds more bugs.

ACM Reference Format:

Anonymous Author(s). 2022. Programming Language Representation with Semantic-level Structure. In *Proceedings of ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2022).* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Software testing is the cruicial process when developing software. It evaluates an attribute or capability of the software and determines that it meets the requirements by examining the behavior of the software under test. Software testing in the early stage of the development finds bugs, and fixing them saves amount of costs. In addition, reliable software testing methodology ensures software quality to users in that the software meets requirements by verification and validation. Regarding that, NLP application is a branch of artifical intelligence software, and testing NLP application also becomes important process as well.

The prevalent models of NLP are evaluated via train-validation-test splits. train and validation set is used to train the NLP model and the hold-out set is used for testing by measuring accuracy. The accuracy is a indicator of the performance of the models. Despite its usefulness, the main limitation of the testing paradigm is that the hold-out set often overestimates the performances. Each dataset comes with specific biases, and the biases increase the discrepancy of distribution between dataset and real-world [11]. The aforementioned accuracy on hold-out set does not consider the discrepancy and it is limited to achieve comprehensive performrance of the NLP model. As a consequence, it is difficult to analyze weekness of the model [21].

On the subject of the limitation of traditional testing paradaigm, a number of methods have been proposed. First, multiple diagnostic datasets for evaluating NLP model were introduced for obtaning generalized evaluation of the NLP model [20]. Not only that, model is evaluated on different aspects such as robustness of the model on adversarial sets [2, 5, 14, 17], fairness [10, 16], logical consistancy [12], prediction interpretations [13] and interactive error analysis [21]. Especially, CHECKLIST implements behavioral testing methodolgy for evaluating multiple linguistic capabilities of NLP model [15]. CHECKLIST introduces input-output behaviors of linguistic capabilities and generates behavior-guided inputs for validating the behaviors. It provides comprehensive behavioral testing of NLP models through a number of generated inputs. However, the approach only relies on manually generated input templates, thus the template generation becomes expensive and time consuming. In addition, the generated templates are selective and often too simple, and it is limited to provide restricted evaluation of linguistic capabilities. Thus, it does not garauntee the comprehensive evaluation.

In this paper, we present Auto-CHECKLIST, an automated NLP model evaluation method for comprehensive behavioral testing of NLP models on sentiment analysis task. For each behavior of linguistic capability, Auto-CHECKLIST does not rely on the manual input generation. Instead, it establishes input requirement for evaluating a linguistic capability and finds suitable inputs that meet the requirement from existing public dataset. Therefore, Auto-CHECKLIST increases input diversity and generality. Further, Auto-CHECKLIST applies the fuzzing testing principle to generate inputs by mutating the selected inputs as seed inputs. Fuzzer in Auto-CHECKLIST first expands seed input grammar structures and determines its available part-of-speech to maintain structural naturalness. After that, to hold contextual naturalness of the mutated inputs, the fuzzer completes the expanded new structures via data-driven context-aware word suggestion. Additionally, sentiment-independent words in the inputs are replaced with rule-based word suggestion.

We demonstrate its generality and utility as a NLP model evaluation tool by evaluating well-known sentiment analysis models: BERT-base [4], RoBERTa-base [8] and DistilBERT-base [18]. We show that

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ISSTA 2022, 18-22 July, 2022, Daejeon, South Korea

© 2022 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 BACKGROUND

3 RELATED WORK

4 TECHNIQUE AND IMPLEMENTATION

Auto-CHECKLIST generates input sentences with the following phases illustrated in 1: 1. search phase searches seed sentences according to its requirement of linguistic capability, 2. seed parsing phase parses the found seed sentences and extract their context-free grammar, 3. reference phase collects large corpus, 4. syntax expansion identification, and 5.sentence expansion and generation. In this section, we provide more details on each phase.

4.1 Search phase

The search phase in Auto-CHECKLIST searches and samples subset of sentences in the dataset that meets the requirement extracted from each linguistic capability. In order to evaluate NLP models on linguistic capability precisely, there are two conditions in our interest. First, inputs in the distrbution characterized by each linguistic capability is of importance. Each linguistic capability represents performance of NLP model on specific input distribution. Therefore, the linguistic capability introduces the constraints of the input distribution, and input sentences from the constrained distribution are only qualified to be used for evaluating the NLP model on the linguistic capability. In addition, diversity in inputs is important to evaluate NLP models on the linguistic capability. The idea is that inputs that differ are more likely to cover the NLP model behavior, and more coverage increases trustworthiness of the evaluation. Based on them, we approach

5 RESEARCH QUESTIONS FOR EVALUATION

6 EXPERIMENT

In this section, we present experiments to evaluate the effectiveness of our proposed evaluation methodology. In particular, we address the following research questions:

- **RQ1** : How effective is our proposed evaluation model for finding failures given a linguistic capability?
- **RQ2** : How effective is our proposed model for generating diverse test cases?
- **RQ3** : How effective is test cases generated from our proposed model for detecting diverse type of errors? acc score
- **RQ4** : How effective is our new test case generation using context-free grammar expansion?

For answering **RQ1** and **RQ2**, we generate test cases and use them for evaluating model on linguistic capabilities. In this experiment, We assess the ability to find failures by anlyzing model's performance on the generated test cases. We also measure the diversity among the generated test cases using similarites among them. Next, we answer **RQ3** by retraining sentiment analysis model with generated test cases and measuring performances. The idea behind this is that more comprehensive inputs becomes closer to real-world distribution and addresses more type of errors. Therefore, it leads to improve the model performance. In this experiment, We retrain the model and compare performances of the retrained model. Not only that, we conduct ablation study of context-free grammar expansion to understand the its impact in our approach.

6.1 Experiment Setup

Seed Input Selection. For each linguistic capability, we first search all sentences that meet its requirement. Among found sentences, we randomly select 10 sentences due to memory constraint.

Word Sentiment. we extract sentiments of words using the SentiWordNet [1]. The SentiWordNet is a publicly available lexical resource of words on Wordnet with three numerical scores of objectivity, positivity and negativity. Sentiment word labels from the scores are classified from the algorithm from Mihaela et al. [3].

Context-free grammar Expansion. We build a reference Context-free grammar of natural language from the English Penn Treebank corpora [9, 19]. The corpus is sampled from 2,499 stories from a tree year Wall Street Journal collection The Treebank provides a parsed text corpus with annotation of syntactic and semantic structure. In this experiment We implement the treebank corpora available through NLTK, which is a suite of libraries and programs for Natural language processing for English. In addition, we parse the seed input using into its CFG using the Berkeley Neural Parser [6, 7], a high-accuracy parser with models for 11 languages. The input is a raw text in natural language and the output is the string representation of parse tree. Next after comparing CFGs between reference and seed input, we randomly select 10 expansions for generating templates due to memory constraint.

Synonyms. Auto-CHECKLIST searches synonyms of each token from synonym sets extracted from WordNet using Spacy open-source library for NLP.

Models. We evaluate the following sentiment analysis models via Auto-CHECKLIST: BERT-base [4], RoBERTa-base [8] and DistilBERT-base [18]. These models are fine-tuned on SST-2 and their accuracies are 92.43%, 94.04% and 91.3%.

Retraining. We retrain sentiment analysis models. we split Auto-CHECKLIST generated test cases into train/validation/test sets with the ratio of 8:1:1. The number of epochs and batch size for retraining are 1 and 16 respectively.

7 RESULT

REFERENCES

[1] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*. European Language Resources Association (ELRA), Valletta, Malta. http://www.lrec-conf.org/proceedings/lrec2010/pdf/769_Paper.pdf

[2] Yonatan Belinkov and Yonatan Bisk. 2017. Synthetic and Natural Noise Both Break Neural Machine Translation. *CoRR* abs/1711.02173 (2017). arXiv:1711.02173 http://arxiv.org/abs/1711.02173

[3] Mihaela Colhon, Åđtefan VlĂČduÅčescu, and Xenia Negrea. 2017. How Objective a Neutral Word Is? A Neutrosophic Approach for the Objectivity Degrees of Neutral Words. *Symmetry* 9, 11 (2017). https://doi.org/10.3390/sym9110280

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[5] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial Example Generation with Syntactically Controlled Paraphrase Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1875–1885. https://doi.org/10.18653/v1/N18-1170

[6] Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual Constituency Parsing with Self-Attention and Pre-Training. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 3499–3505. https://doi.org/10.18653/v1/P19-1340

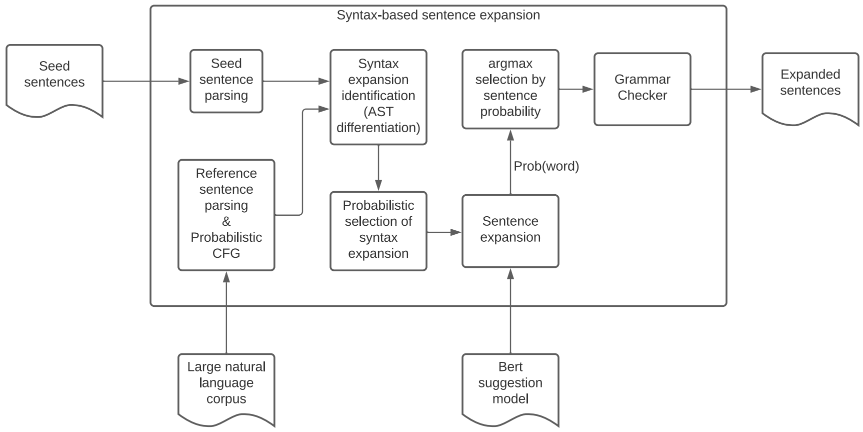

Figure 1: Overall diagram of Auto-CHECKLIST.

[7] Nikita Kitaev and Dan Klein. 2018. Constituency Parsing with a Self-Attentive Encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 2676–2686. https://doi.org/10.18653/v1/P18-1249

[8] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[9] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Comput. Linguist.* 19, 2 (jun 1993), 313âĂŞ330.

[10] Vinodkumar Prabhakaran, Ben Hutchinson, and Margaret Mitchell. 2019. Perturbation Sensitivity Analysis to Detect Unintended Model Biases. *CoRR* abs/1910.04210 (2019). arXiv:1910.04210 http://arxiv.org/abs/1910.04210

[11] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. 2019. Do ImageNet Classifiers Generalize to ImageNet?. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 5389–5400. https://proceedings.mlr.press/v97/recht19a.html

[12] Marco Tulio Ribeiro, Carlos Guestrin, and Sameer Singh. 2019. Are Red Roses Red? Evaluating Consistency of Question-Answering Models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 6174–6184. https://doi.org/10.18653/v1/P19-1621

[13] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. *CoRR* abs/1602.04938 (2016). arXiv:1602.04938 http://arxiv.org/abs/1602.04938

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically Equivalent Adversarial Rules for Debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 856–865. https://doi.org/10.18653/v1/P18-1079

[15] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond Accuracy: Behavioral Testing of NLP models with CheckList. In *Association for Computational Linguistics (ACL)*.

[16] Paul Röttger, Bertram Vidgen, Dong Nguyen, Zeerak Waseem, Helen Z. Margetts, and Janet B. Pierrehumbert. 2020. HateCheck: Functional Tests for Hate Speech Detection Models. *CoRR* abs/2012.15606 (2020). arXiv:2012.15606 https://arxiv.org/abs/2012.15606

[17] Barbara Rychalska, Dominika Basaj, Alicja Gosiewska, and Przemysław Biecek. 2019. Models in the Wild: On Corruption Robustness of Neural NLP Systems. In *Neural Information Processing*, Tom Gedeon, Kok Wai Wong, and Minho Lee (Eds.). Springer International Publishing, Cham, 235–247.

[18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv* abs/1910.01108 (2019).

[19] Sphinx and NLTK Theme. 2021. *NLTK Documentation*. https://www.nltk.org/howto/corpus.html.

[20] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. *CoRR* abs/1804.07461 (2018). arXiv:1804.07461 http://arxiv.org/abs/1804.07461

[21] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2019. Errudite: Scalable, Reproducible, and Testable Error Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 747–763. https://doi.org/10.18653/v1/P19-1073