Revisit Input Perturbation Problems for LLMs: A Unified Robustness Evaluation Framework for Noisy Slot Filling Task

Guanting Dong[1], Jinxu Zhao[1], Tingfeng Hui[1], Daichi Guo[1], Wenlong Wang[1], Boqi Feng[1], Yueyan Qiu[1], Zhuoma Gongque[1], Keqing He[2], Zechen Wang[1], and Weiran Xu[1(✉)]

[1] Beijing University of Posts and Telecommunications, Beijing, China
{dongguanting,zhaojinxu,huitingfeng,guodaichi,zechen_wang,fbq,yuanqiu,wanwenlong,xuweiran}@bupt.edu.cn
[2] Meituan Group, Beijing, China
hekeqing@meituan.com

Abstract. With the increasing capabilities of large language models (LLMs), these high-performance models have achieved state-of-the-art results on a wide range of natural language processing (NLP) tasks. However, the models' performance on commonly-used benchmark datasets often fails to accurately reflect their reliability and robustness when applied to real-world noisy data. To address these challenges, we propose a unified robustness evaluation framework based on the slot-filling task to systematically evaluate the dialogue understanding capability of LLMs in diverse input perturbation scenarios. Specifically, we construct a input perturbation evaluation dataset, Noise-LLM, which contains five types of single perturbation and four types of mixed perturbation data. Furthermore, we utilize a multi-level data augmentation method (character, word, and sentence levels) to construct a candidate data pool, and carefully design two ways of automatic task demonstration construction strategies (instance-level and entity-level) with various prompt templates. Our aim is to assess how well various robustness methods of LLMs perform in real-world noisy scenarios. The experiments have demonstrated that the current open-source LLMs generally achieve limited perturbation robustness performance. Based on these experimental observations, we make some forward-looking suggestions to fuel the research in this direction (The code is available at https://github.com/ZhaoJin-xu/A-Unified-Robustness-Evaluation-Framework-for-Noisy-Slot-Filling-Task).

Keywords: Large language models · Robustness evaluation · Slot filling · Input perturbation

G. Dong and J. Zhao—The first two authors contribute equally.

F. Liu et al. (Eds.): NLPCC 2023, LNAI 14302, pp. 682–694, 2023.
https://doi.org/10.1007/978-3-031-44693-1_53

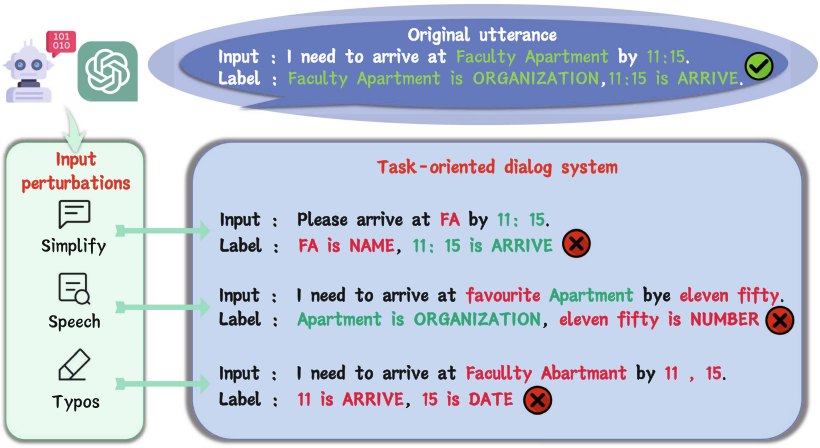


Fig. 1. The impact of various types of input perturbations on the slot filling system in real-word scenarios

1 Introduction

The slot filling (SF) task in the goal-oriented dialog system aims to identify task-related slot types in certain domains for understanding user utterances. Recently, Large-scale language models (LLMs) [2,16,23] have shown an impressive ability for in-context learning with only a few task-specific examples as demonstrations. Under the framework of in-context learning, LLMs have achieved promising results in a variety of NLP tasks, including machine translation (MT) [20], question answering (QA) [9] and named entity extraction (NEE) [2]. However, despite its powerful abilities, the high performance of these models depend heavily on the distribution consistency between training data and test data. As the data distribution of dialogues in real scenarios is unknown [25], there are still many challenges in applying these methods to real dialogue scenarios.

In real dialogue systems, due to the diverse language expression and input errors of humans, models often need to deal with a variety of input perturbations. As shown in Fig. 1, due to different expression habits, users may not interact with the dialogue system following the standard input format, but may simplify their queries to express the same intent. What's more, errors in the upstream input system may also introduce disturbances to the downstream model (e.g. typos from keyboard input, speech errors from ASR systems). LLMs are usually pre-trained and fine-tuned on perturbation-free datasets, resulting in poorer performance than supervised small language models with robust settings on noisy slot filling tasks. However, the accuracy of slot filling tasks directly reflects the model's understanding of user queries, which would impact the performance of the model in other downstream tasks. Therefore, exploring the robustness and generalization of LLMs under various input perturbations is crucial for the application of task-oriented dialogue systems in real scenarios.

To address the above challenges, in this paper, we aim to investigate how well various robustness methods of large language models perform in real-world noisy scenarios, and then provide empirical guidance for the research of robust LLMs. Based on the slot filling task, we propose a unified robustness evaluation framework to evaluate the dialogue understanding capability of large models in diverse input perturbation scenarios. Firstly, we construct a input perturbation evaluation dataset, Noise-LLM, to investigate the robustness of large language models in two different input perturbation settings. 1) **Single perturbation setting:** Based on the DST dataset Raddle [18] which contains various types of real-world noisy texts, we transformed them into slot filling data through manual annotation which consists of 5 types of single perturbation at the character, word, and sentence levels. 2) **Mixed perturbation setting:** We utilized the widely used Slot Filling dataset SNIPS [3] and data augmentation tools [6] to construct 4 types of mixed perturbation that fit real-world dialog scenarios. Furthermore, we utilize a multi-level data augmentation method (character, word, and sentence levels) to construct a candidate data pool, and carefully designed two ways of automatic task demonstration construction strategies (Instance-level and Entity-level) with various prompt templates. Our goal is to provide empirical guidance for research on robust LLMs based on our evaluation framework. Based on this framework, we conduct extensive experiments on Noise-LLM and analyze the results from the perspectives of input perturbation types and different demonstrations, respectively.

Our main contributions are concluded as follows:

(1) To our best knowledge, we are the first to comprehensively investigate the effects of input perturbations at different levels on LLMs, and construct an input perturbation evaluation dataset, Noise-LLM, which includes single perturbation and mixed perturbation settings that fit real-world scenarios.
(2) We propose a unified robustness evaluation framework based on the SF task, which includes a multi-level data augmentation method, diverse prompt templates, and various automatic task demonstration construction strategies.
(3) Experiments result demonstrate that the current open-source LLMs generally have limited ability to counter input perturbations. The extensive analysis also provides empirical guidance for further research.

2 Related Work

2.1 Input Perturbation Problem

Owing to the notable performance gaps between benchmarks and real-world scenarios, the robustness of NLP systems in the face of input perturbations has garnered significant attention recently. [10,14] conduct empirical evaluations of the robustness of various NLP systems against input perturbations using synthetic benchmarks that we generated. [4,7] focus on the robustness of the sequence labeling framework against optical input perturbations (e.g. misspellings, OOV).

[5] investigate the robustness of dialogue systems on ASR noise. [21] mainly focus on the ASR noise-robustness SLU models in dialogue systems. In this paper, we investigate the abilities of ChatGPT against input perturbation problems for slot filling tasks.

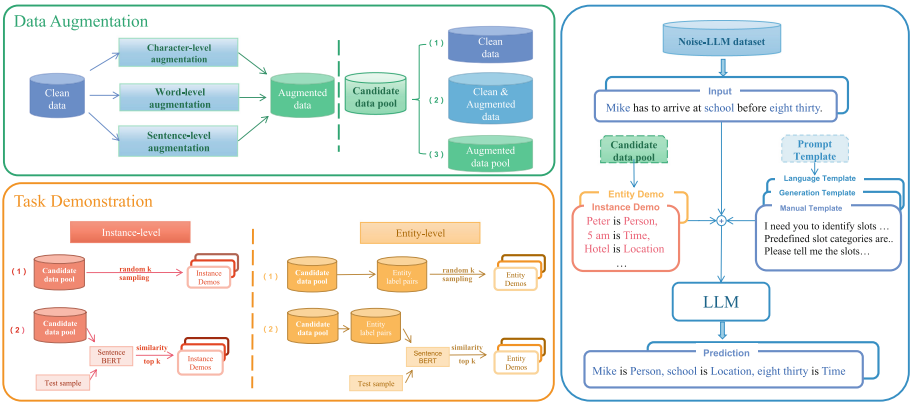


Fig. 2. The overall architecture of our unified robustness evaluation framework based on slot filling task

2.2 Large Language Models

The emergence of large-scale language models (LLMs) have brought revolutionary transformation to the field of natural language processing (NLP) [22]. LLMs, such as GPT-3 [2], LLaMA [23], ChatGPT, and GPT-4 [16], have demonstrated impressive abilities on various tasks and in generating fluent responses due to the large-scale training corpora, as well as the use of external techniques such as instruction tuning and reinforcement learning from human feedback (RLHF) [17]. LLMs based on generative framework reformulate the information extraction task [11], task-oriented dialog systems [19,26], even from multi modal perspective [12,27,28]. More recently, the NLP community has been exploring various application directions for LLMs. For instance, chain-of-thought prompting [24] enables LLMs to generate problem-solving processes step-by-step, significantly enhancing the model's reasoning ability. Some researchers have utilized the powerful interactive capabilities of LLMs to generate commands that invoke external tools for better handling of various downstream tasks [22]. Other researchers have proposed parameter-efficient fine-tuining (PEFT) methods to address the issue of excessive computational resource usage during fine-tuning of LLMs, which significantly reduces resource consumption without sacrificing performance [8]. In this paper, we aim to investigate the robustness of LLMs against input perturbations. To be specific, we explore several prompt and demonstration strategies and conduct a large number of experiments to advance our research.

3 Method

3.1 Problem Definition

Given an input utterance $X = \{x_1, x_2, ..., x_N\}$, where N represents the length of X, we adopt a triple $y_i = \{l, r, t\} \in Y$ to represent the $i-th$ entity that appears in X, where Y represents all the entity triplets in X, and l, r denote the entity boundaries, while t denotes the entity type. We use D_{clean}, $D_{augment}$, and D_{test} to denote the conventional labeled slot filling datasets, including the clean dataset, the data-augmented dataset, and the test set, respectively, where $D_{test} = \{X_1 : Y_1, X_2 : Y_2, ...X_N : Y_N\}$. We define $P_i = X_i : Y_i$, and then $D_{test} = P_1, P_2, ..., P_N$, and we formulate the spoken language perturbation process in the real scenario as $\widetilde{P} = \Gamma(P)$, such that $\widetilde{X} \neq X$ but $\widetilde{Y}$ may or may not be identical to Y. The form of the perturbation transformation Γ is flexible, which could be a specific type of perturbation (e.g., Typos) or mixed perturbation.Based on the above process, we can obtain a perturbed test set $\widetilde{D}_{test}$.

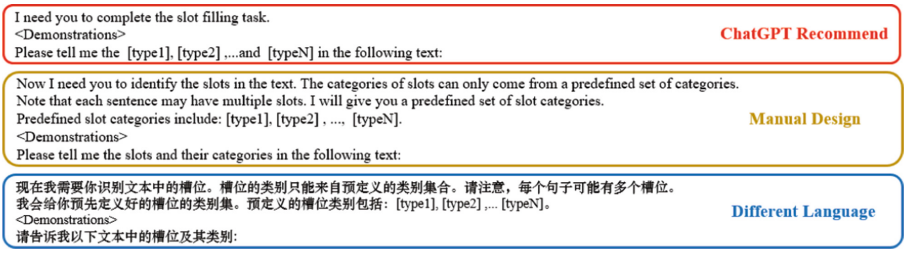


Fig. 3. The different prompt templates for slot filling task.

3.2 Data Augmentation

As shown in the top left corner of Fig. 2, we utilize the data augmentation tool NLPAug [13] to transform the clean training set into an augmented dataset. The specific data augmentation methods employed in this study can be categorized into the following three types: **Character-level augmentation:** randomly add, delete and replace characters in one token with the probability p. **Word-level augmentation:** randomly delete and insert words and replace words with homophones in one sentence with probability p. **Sentence-level augmentation:** replace sentences with synonymous ones.

3.3 Task Demonstration

Task demonstration D is constructed by retrieving instance examples from a dataset. The role of constructing task demonstration is to select appropriate

examples that can effectively demonstrate how the LLM should solve the task. We categorize the demonstration into two types: (1) Entity-oriented demonstration, and (2) Instance-oriented demonstration.

Entity-Oriented demonstration. Given a set of entity type labels $L = l_1, l_2, ..., l_N$, where N is the number of label L, we select one entity example e per label l from D_{train} and modify it into the form $[e_i \text{ is } l_i]$. For each label l, we adopt two strategies for selecting entity e: (1) Randomly selecting entity e from the clean, augmented, or mixed data pool. (2) Retrieving the k most relevant entity examples e from the clean, augmented, or mixed data pool using SentenceBERT [20]. Specifically, the SentenceBERT algorithm generates independent CLS embeddings for the input x and e, and calculates the cosine similarity between them to rank the entity e.

Instance-Oriented Demonstration. Given an example S and its label Y, we modify all of its entities and their corresponding labels into the form of "e_i is l_i", and concatenate them with S to form S'. We also adopt two strategies to select example S: (1) Randomly selecting example S from the clean, augmented, or mixed data pool. (2) Retrieving the k most relevant examples S from the clean, augmented, or mixed data pool using SentenceBERT.

Prompt Template. ChatGPT is a large language model that relies on prompts to guide its output generation. The quality of the output can be influenced by the style of the prompts used. Since slot filling is essentially a sequence labeling task rather than a generation task, it is not entirely consistent with the paradigm of contextual learning. Therefore, we have designed three templates specifically for slot filling. To design these prompts, we initially sought inspiration from ChatGPT by requesting its advice. However, since the templates provided by ChatGPT were not specifically tailored to our task, we felt that this could potentially impact its performance. Hence, we also designed prompts based on the requirements of slot filling. Considering that ChatGPT is primarily pre-trained on English language data, its understanding of prompts in other languages may vary. Therefore, we also tested the English templates translated into Chinese using ChatGPT. The three prompts used in our experiments are shown in Fig. 3.

3.4 In-Context Inference

In-context learning (ICL) is popularized as a way of learning for large language models in the original GPT-3 paper [1]. When using in-context learning, we give the large language model a task description and a set of demonstration examples (input-label pairs). A test input is then appended to the end of the demonstration examples, and the LM is allowed to make predictions only based on the conditioned demonstration examples. In order to answer the question correctly, the model needs to understand the demonstration examples of ICL to determine the input distribution, output distribution, input-output mapping, and format. We concatenate the prompt, demonstration, and test input for in-context learning and use it as model input for inference.

4 Experiments

4.1 Datasets

Based on RADDLE [18] and SNIPS [3], we constructed the Noise-LLM dataset, which includes two settings: single perturbation and mixed perturbation. For the single perturbation setting, RADDLE is a crowdsourced diagnostic evaluation dataset covering a broad range of real-world noisy texts for dialog systems. We extract 50 utterances for clean data, and each type of noisy utterance (Typos, Speech, Simplify, Verbose, and Paraphrase) from RADDLE to construct the evaluation set. **Typos** are caused by non-standard abbreviations, while **Speech** arises from recognition and synthesis errors from ASR systems. **Simplification** refers to users using concise words to express their intentions, while **Verbose** refers to users using redundant words to express the same intention. **Paraphrase** is also common among users who use different words or restate the text based on their language habits. For mixed perturbation setting, based on SNIPS, we used textflint [6] to introduce character-level perturbation **Typos**, Word-level perturbation **Speech**, sentence-level perturbation **AppendIrr**, and mixed perturbation to the test set and construct a multi-perturbation evaluation set.

Table 1. The performance (F1 score) of the finetuned SOTA (NAT, PSSAT) and LLMs (Text-davinci-003, ChatGPT) with best entity-oriented and instance-oriented demonstrations on clean and noisy test sets.

Methods	Clean	Character	Word	Sentence			Overall
		Typos	Speech	Paraphrase	Simplification	Verbose	
NAT($\mathcal{L}_{aug}$)	96.01	67.47	85.23	87.73	87.32	85.41	87.21
NAT($\mathcal{L}_{stabil}$)	96.04	67.54	85.16	87.42	87.33	85.29	87.27
PSSAT	96.42	68.34	85.65	91.54	89.73	85.82	88.16
Text-davinci-003	43.09	34.26	39.34	38.42	40.12	37.18	38.54
ChatGPT	71.43	40.65	60.00	55.56	65.54	55.56	57.21
ChatGPT+Instance level	68.21 (−3.2)	65.04 (+24.3)	70.56 (+10.5)	58.82 (+2.2)	73.02 (+7.4)	61.77 (+6.2)	68.34 (+11.1)
ChatGPT+Entity level	74.07 (+2.6)	62.18 (+21.5)	55.39 (+4.6)	75.59 (+18.9)	70.96 (+5.4)	71.75 (+16.1)	71.55 (+14.3)

4.2 Implementation Details

We use the default settings to invoke the OpenAI API for text-davinci-003. For ChatGPT, we manually evaluate the results using the corresponding platform and demo websites. For PSSAT [4], we contact the authors and obtain the source code. We follow PSSAT's experiment settings for their upstream work and downstream work. For all the experiments of PSSAT, we conduct the training and testing stages on the A6000 GPU.

4.3 Baselines and Large Language Models

NAT [15] provides two perturbation-aware training methods(data augmentation & stability training) to improve the robustness of the model against perturbations.

PSSAT [4] introduces two MLM-based training strategies to better learn contextual knowledge from perturbed corpus, and utilizes a consistency processing method to filter the generated data.

Text-davinci-003 [2] is the most advanced GPT-3.5 model with 175B parameters.

Table 2. The performance of ChatGPT with different example selection strategies on clean and noisy test sets. The two best methods for each perturbation are marked.

Demonstration	Strategy	Setup	Clean	Character	Word	Sentence			Overall
				Typos	Speech	Paraphrase	Simplification	Verbose	
Instance-oriented level	Random	Clean	**76.92**	58.73	**68.26**	58.61	**74.19**	57.78	65.36
		Augment	68.21	**65.04**	**70.56**	58.82	**73.02**	61.77	**68.34**
		Clean+Augment	71.31	61.01	57.81	53.43	65.03	63.71	62.32
	Retrieve	Clean	56.49	54.84	60.00	45.53	53.84	52.55	54.37
		Augment	61.42	51.67	64.66	51.97	64.00	61.04	60.25
		Clean+Augment	60.32	52.44	60.93	55.82	64.06	44.96	58.24
Entity-oriented level	Random	Clean	74.07	**62.18**	55.39	**75.59**	70.96	**71.75**	**71.55**
		Augment	**78.39**	58.54	58.91	**66.14**	70.40	**64.75**	65.61
		Clean+Augment	70.09	60.19	61.88	64.12	65.50	57.37	63.22
	Retrieve	Clean	53.66	52.89	54.55	54.13	57.37	42.96	53.42
		Augment	70.49	51.72	68.21	55.82	63.86	47.83	61.38
		Clean+Augment	72.58	61.16	65.40	42.46	71.64	46.97	62.84

ChatGPT is created by fine-tuning a GPT-3.5 series model via instruction tuning and reinforcement learning from human feedback (RLHF).

4.4 Main Result

Perturbation Level. Our experimental results are shown in Table 1. LLM performance (ChatGPT, Text-davinci-003) is far behind fintune SOTA (NAT, PSSAT) regardless of whether it is on a clean test set or with perturbations at various levels. This phenomenon may be due to the fact that large models are usually pre-trained on large-scale general training data, which makes it difficult to perform well on specific domain data in zero-shot situations. Further comparison of ChatGPT's performance on clean data and various levels of perturbations shows that large models have the most significant drop in character-level perturbation **Typos** (71.43->40.65) and sentence-level perturbation **Verbose** (71.43->55.56), with no significant changes in **Simplification**. Based on the main experimental table and specific case analysis, we have three explanations:

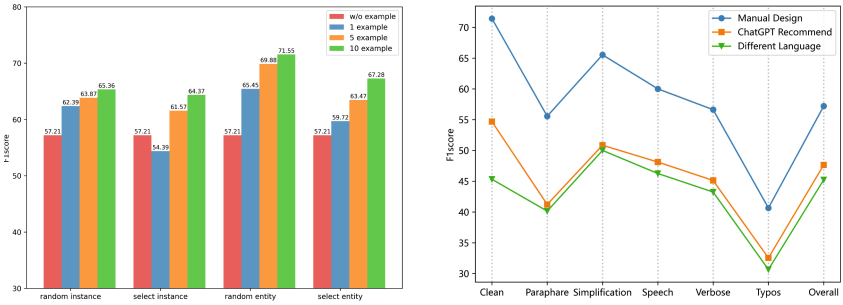
1. Fine-grained perturbation like **Typos** affect the semantics of the entity itself. Although large language models can infer from the context of the sample and have some robustness in recognizing slot entities, the drastic changes in entity semantics can make it difficult to predict the correct entity label.
2. **Verbose** language tends to use redundant expressions to convey the intended meaning, often without explicitly mentioning specific entities. ChatGPT will over-predict entities due to knowledge confusion, which greatly affects its performance in the verbose domain.
3. Despite affecting the semantic information of the original sentence, **simplification** perturbation mitigates the chances of incorrect slot position prediction in ChatGPT by reducing the complexity of input content. Consequently, the model performs relatively better in the Simplification domain.

Demonstration Level. As is shown in Table 2, incorporating entity-oriented and instance-oriented demonstration strategies into ChatGPT indicates a substantial improvement in the overall results (14.34 for Entity and 11.13 for Instance), which demonstrates the effectiveness of our approach. Our evaluation primarily involved analyzing 1) the impact of different types of demonstration, 2) the selection of demonstration, and 3) the distribution of demonstration on the results.

Table 3. The performance of Text-davinci-003 and ChatGPT with best entity-oriented and instance-oriented demonstrations on mixed perturbation.

Methods	Clean	Char	Word	Sen	Char+Word	Word+Sen	Char+Sen	Char+Word+Sen	Overall
		Typos	Speech	AppendIrr	Spe+Typ	Spe+App	Ent+App	Spe+App+Typ	
Text-davince-003	31.24	27.18	23.41	27.48	19.32	19.78	20.73	18.84	24.64
ChatGPT	59.65	42.11	34.83	45.61	27.58	31.03	26.38	26.11	38.18
ChatGPT+Instance level	67.18	48.94	42.25	52.61	34.26	38.79	38.64	30.67	46.58
ChatGPT+Entity level	65.71	47.36	40.37	53.42	36.55	37.35	34.21	29.06	44.27

The Type of Demonstration. Firstly, the entity-level strategy performs better than the instance-level strategy overall, and both strategies show a significant improvement on character-level perturbation (Typos). Specifically, the entity-level strategy exhibits a remarkable improvement on coarse-grained perturbation such as verbose and paraphrase. This is because entity demonstrations provide clear entity boundary information and corresponding labeling, aiding in distinguishing entities from non-entities and greatly reducing entity boundary ambiguity. The instance-level strategy, compared to the entity-level strategy, significantly improves the model's ability to handle Simplify and Speech perturbations. This is due to the diverse context examples provided in instance demonstrations which enhance the model's understanding of domain-specific context information and the distribution of slots, assisting in better reasoning to test samples.



(a) (b)

Fig. 4. (a) The influence of the number of examples in demonstrations. (b) The performance of different types of prompt templates.

The Selection of Demonstration. In most cases, randomly selecting samples proves to be a more effective strategy for both entity-level and instance-level methods, compared to selecting samples based on similarity. The latter may even further reduce the model's performance. One plausible explanation for this finding is that in noisy scenarios, semantic similarity is generally low (¡0.4), making it challenging to ensure consistency with the test sample distribution. By contrast, randomly retrieved samples offer a diverse set of noisy samples that can positively stimulate the context learning ability of large models.

The Distribution of Demonstration. Our experimental results show that, for instance-level models, selecting 10 examples exclusively from the augmented data pool leads to the highest performance on the noisy test dataset. In contrast, for entity-level models, selecting 10 examples from the clean data pool results in the best overall performance on the noisy test dataset. We believe that instance augmented demonstrations enable LLMs to learn the slot information and the correspondence between slots and labels under input perturbations, thereby enhancing the LLMs' ability to understand semantic information in noisy input and increasing their robustness to noise. In contrast, entity demonstrations only provide the correspondence between entities and labels, and selecting examples from the augmented data pool may mislead LLMs to incorrectly identify and classify entities. Furthermore, entity clean demonstrations enable LLMs to learn the correct entity boundaries and the correspondence between entities and labels, thus achieving better performance. However, when we select 10 examples from a mixed data pool of clean and augmented examples, ChatGPT's performance is not as good as when selecting examples exclusively from the clean or augmented data pool. In some cases, it even performs worse than ChatGPT without any examples. This also demonstrates that consistent distribution of inputs in the demonstrations contributes to performance gains during in-context learning.

Mixed Perturbation Experiment. Table 3 shows the performance of different models in facing single and mixed perturbation. Obviously, ChatGPT's performance will drop significantly when facing mixed perturbation, and is lower than all single perturbation results. Although ChatGPT performs better than Text-Davince-003, this improvement becomes very limited in the multi-perturbation scenario. We found that adding entity or instance examples to ChatGPT can effectively improve the model's perturbation robustness. Specifically, the method of adding entity examples improves the overall performance by 22%, while adding instance examples improves it by 21%. Even with the joint interference of three perturbations, our method of selecting examples appropriately can still maintain a 17% improvement in performance, which demonstrates the effectiveness and stability of our proposed approach.

Number of Demonstrations. The Fig. 4(a) shows the impact of the number of examples in demonstrations on the overall performance of chatgpt in single perturbation scenarios. As the number of examples increases, chatgpt's performance improves in both clean and noisy scenarios. Further analysis shows that demonstration selection based on similarity improves with an increase in the number of demos, as the overall semantic similarity between demos and test examples increases and their distribution consistency improves. However, increasing the number of demonstrations through random selection does not improve performance beyond a certain point. This is because as the number of demonstrations increases, their diversity reaches a peak, and additional demonstrations do not provide new semantic information, resulting in stable performance.

Type of Prompts. The Fig. 4(b) shows the impact of different types of prompts on the results. For both the clean and noisy test sets, manually designed prompts perform better than the model-recommended templates. Obviously, when we input detailed descriptions of slot filling task paradigms into LLMs, they can better understand the slot filling task and achieve better performance. When translating the manually designed templates into Chinese, the model's performance also dropped significantly, which verifies our hypothesis that ChatGPT has different levels of understanding of prompts in different languages. Compared to manually constructed templates, ChatGPT's recommended templates show a significant difference, highlighting both the sensitivity of LLMs to templates and the challenge of using LLMs to automate template construction.

5 Conclusion

In this paper, we are the first to comprehensively investigate the effects of input perturbations at different levels on LLMs. We further propose a unified robustness evaluation framework for noisy slot filling to systematically evaluate the dialogue understanding capability of LLMs in diverse input perturbation scenarios. Specifically, we construct a noise evaluation dataset, Noise-LLM, which contains

five types of single noise and four types of mixed noise data. Moreover, we utilized a multi-level data augmentation method to construct a candidate data pool, and carefully designed two ways of automatic task demonstration construction strategies with various prompt templates. Experiments result demonstrate that the current open-source LLMs generally have limited ability to counter input perturbations. Our analysis provides empirical guidance for future research.

References

1. Brown, T.B., et al.: Language models are few-shot learners. ArXiv abs/2005.14165 (2020)
2. Brown, T.B., et al.: Language models are few-shot learners (2020)
3. Coucke, A., et al.: Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces (2018)
4. Dong, G., et al.: PSSAT: a perturbed semantic structure awareness transferring method for perturbation-robust slot filling. In: Proceedings of the 29th International Conference on Computational Linguistics, pp. 5327–5334. International Committee on Computational Linguistics, Gyeongju (2022). https://aclanthology.org/2022.coling-1.473
5. Gopalakrishnan, K., Hedayatnia, B., Wang, L., Liu, Y., Hakkani-Tur, D.: Are neural open-domain dialog systems robust to speech recognition errors in the dialog history? An empirical study (2020)
6. Gui, T., et al.: TextFlint: unified multilingual robustness evaluation toolkit for natural language processing (2021)
7. Guo, D., et al.: Revisit out-of-vocabulary problem for slot filling: a unified contrastive frameword with multi-level data augmentations. ArXiv abs/2302.13584 (2023)
8. Hu, E.J., et al.: LoRA: low-rank adaptation of large language models (2021)
9. Lazaridou, A., Gribovskaya, E., Stokowiec, W., Grigorev, N.: Internet-augmented language models through few-shot prompting for open-domain question answering. ArXiv abs/2203.05115 (2022)
10. Li, X., et al.: A robust contrastive alignment method for multi-domain text classification. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7827–7831 (2022). https://doi.org/10.1109/ICASSP43922.2022.9747192
11. Li, X., et al.: Generative zero-shot prompt learning for cross-domain slot filling with inverse prompting. In: Findings of the Association for Computational Linguistics: ACL 2023, pp. 825–834. Association for Computational Linguistics, Toronto (2023). https://aclanthology.org/2023.findings-acl.52
12. Lin, J., et al.: M6: multi-modality-to-multi-modality multitask mega-transformer for unified pretraining. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, KDD 2021, pp. 3251–3261. Association for Computing Machinery, New York, NY (2021). https://doi.org/10.1145/3447548.3467206
13. Ma, E.: NLP augmentation (2019). https://github.com/makcedward/nlpaug
14. Moradi, M., Samwald, M.: Evaluating the robustness of neural language models to input perturbations. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 1558–1570. Association for Computational Linguistics, Online and Punta Cana (2021). https://doi.org/10.18653/v1/2021.emnlp-main.117, https://aclanthology.org/2021.emnlp-main.117

15. Namysl, M., Behnke, S., Köhler, J.: NAT: noise-aware training for robust neural sequence labeling. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1501–1517. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.acl-main.138, https://aclanthology.org/2020.acl-main.138
16. OpenAI: GPT-4 technical report (2023)
17. Ouyang, L., et al.: Training language models to follow instructions with human feedback (2022)
18. Peng, B., Li, C., Zhang, Z., Zhu, C., Li, J., Gao, J.: RADDLE: an evaluation benchmark and analysis platform for robust task-oriented dialog systems. arXiv preprint arXiv:2012.14666 (2020)
19. Qixiang, G., et al.: Exploiting domain-slot related keywords description for few-shot cross-domain dialogue state tracking. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pp. 2460–2465. Association for Computational Linguistics, Abu Dhabi (2022). https://aclanthology.org/2022.emnlp-main.157
20. Reimers, N., Gurevych, I.: Sentence-BERT: sentence embeddings using Siamese BERT-networks. arXiv preprint arXiv:1908.10084 (2019)
21. Ruan, W., Nechaev, Y., Chen, L., Su, C., Kiss, I.: Towards an ASR error robust spoken language understanding system. In: Interspeech 2020 (2020). https://www.amazon.science/publications/towards-an-asr-error-robust-spoken-language-understanding-system
22. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., Zhuang, Y.: HuggingGPT: solving AI tasks with ChatGPT and its friends in HuggingFace (2023)
23. Touvron, H., et al.: LLaMA: open and efficient foundation language models (2023)
24. Wei, J., et al.: Chain-of-thought prompting elicits reasoning in large language models (2023)
25. Wu, D., Chen, Y., Ding, L., Tao, D.: Bridging the gap between clean data training and real-world inference for spoken language understanding. arXiv preprint arXiv:2104.06393 (2021)
26. Zeng, W., et al.: Semi-supervised knowledge-grounded pre-training for task-oriented dialog systems. In: Proceedings of the Towards Semi-Supervised and Reinforced Task-Oriented Dialog Systems (SereTOD), pp. 39–47. Association for Computational Linguistics, Abu Dhabi (2022). https://aclanthology.org/2022.seretod-1.6
27. Zhang, Y., et al.: Pay attention to implicit attribute values: a multi-modal generative framework for AVE task. In: Findings of the Association for Computational Linguistics: ACL 2023, pp. 13139–13151. Association for Computational Linguistics, Toronto (2023). https://aclanthology.org/2023.findings-acl.831
28. Zhao, G., Dong, G., Shi, Y., Yan, H., Xu, W., Li, S.: Entity-level interaction via heterogeneous graph for multimodal named entity recognition. In: Findings of the Association for Computational Linguistics: EMNLP 2022, pp. 6345–6350. Association for Computational Linguistics, Abu Dhabi (2022). https://aclanthology.org/2022.findings-emnlp.473