Зборник радова Факултета техничких наука, Нови Сад

УДК: 007:004
ДОИ: https://doi.org/10.24867/35OI03Fejes

Употреба великих језичких модела у тестирању софтвера

Use of Large Language Models in Software Testing

Давид Фејеш, *Факултет техничких наука, Нови Сад*

Студијски програм – ИНЖЕЊЕРСТВО ИНФОРМАЦИОНИХ СИСТЕМА

Кратак садржај – *Развој генеративне вештачке интелигенције омогућио је широку примену великих језичких модела у различитим доменима развоја софтвера. Велики језички модели се све чешће користе као подршка у фазама аутоматизације, анализе и евалуације тестова, повећавајући ефикасност тестирања. Овај рад се бави улогом и потенцијалом великих језичких модела у савременом тестирању софтвера, са посебним освртом на његове предности, ограничења и факторе који утичу на поузданост. Посебна пажња посвећена је питању поузданости и стабилности модела, што представља кључни изазов у практичној примени. Резултати указују да велики језички модели тренутно имају највећу вредност као подршка постојећим алатима и методама, док њихова пуна аутономија захтева даља истраживања.*

Кључне речи (три до пет): *велики језички модели, тестирање софтвера, аутоматизација тестирања, вештачка интелигенција*

Abstract – *The advancement of artificial intelligence has enabled the broad adaptation of large language models across various domains of software development. These models are increasingly used to support automation, analysis, and evaluation of tests, thereby improving the overall efficiency of software testing. This paper explores the role and potential of large language models in modern software testing, with particular emphasis on their advantages, limitations, and factors that affect reliability. Special attention is given to the issues of consistency and stability, which represent major challenges in real-world problems. This finding suggests that large language models currently provide the greatest value as intelligent support tools that complement existing methods, while their full autonomy in testing remains a subject of ongoing research.*

Keywords: (three to five): *large language models, software testing, test automation, artificial intelligence*

НАПОМЕНА: Овај рад проистекао је из мастер рада чији ментор је био др Владимир Мандић, ванр. проф.

1. УВОД

Напредак у области генеративне вештачке интелигенције последњих година значајно је утицао на домен развоја и тестирања софтвера. Велики језички модели (енгл. *Large Language Models* - LLMs) представљају највидљивији резултат тог напретка, јер поседују способност разумевања природног језика и генерисања кода који одговара логичким и контекстуалним захтевима. Њихова примена у тестирању софтвера отвара нове могућности за аутоматизацију и оптимизацију процеса који су традиционално били временски и ресурсно захтевни, доприносећи ефикаснијем откривању грешака и побољшању квалитета кода. Употребом LLM-ова могуће је аутоматски генерисати тестне случајеве, препознати недовољну покривеност кода, као и идентификовати грешке у раним фазама развоја. Ипак, поред потенцијала, јављају се и изазови који се односе на поузданост, стабилност резултата и ограничену контролу над процесом генерисања. Разумевање ових аспеката постаје кључно за безбедну и ефикасну примену LLM-ова у пракси. У овом раду је спроведен систематски преглед литературе (енгл. *Systematic Literature Review* – SLR) са циљем да се прикаже улога великих језичких модела у савременом тестирању софтвера, да се анализирају њихове најчешће области примене, користи и ограничења, као и фактори који утичу на квалитет и поузданост генерисаних тестова. Посебна пажња посвећена је параметрима конфигурације модела, попут температуре, који имају значајан утицај на стабилност и разноврсност излазних резултата. На основу идентификованих налаза, рад указује на потенцијал LLM-ова као интелигентне подршке постојећим техникама тестирања и дефинише смернице за будућа истраживања у овој области. Досадашња истраживања се углавном фокусирају на техничке аспекте примене LLM-ова при генерисању софтвера, док је систематско сагледавање њихове поузданости и стабилности резултата ретко заступљено. Управо због тога је потребно разматрање постојећих студија, како би се идентификовали трендови, уочене празнине и отворена истраживачка питања која и даље ограничавају њихову практичну примену.

2. ПРЕГЛЕД СТАЊА У ОБЛАСТИ

Истраживање примене великих језичких модела у тестирању софтвера последњих година бележи

значајан раст, нарочито након појаве GPT модела. Све већи број студија тежи ка испитивању потенцијала ових и других модела у области аутоматизације тестирања софтвера. У овом поглављу представљен је протокол који је примењен током прегледа литературе. Обухвата дефинисање циљева истраживања, истраживачких питања, формулисање стратегије претраге и критеријума прихватљивости и коначно процес селекције радова како би се добио коначан скуп анализираних студија.

2.1. Циљеви истраживања

Циљ овог истраживања је сагледавање улоге великих језичких модела у савременом тестирању софтвера.

2.2. Истраживачка питања

Истраживачка питања која су разматрана током прегледа литературе:

ИП1: Како и који велики језички модели се користе у тестирању софтвера?

ИП2: Које су предности и ограничења примене великих језичких модела у тестирању софтвера?

ИП3: Како различите конфигурације великих језичких модела утичу на квалитет и поузданост генерисаних резултата?

2.3. Стратегија претраге

У циљу проналажења релевантних студија, коришћене су две научне базе података, а то су *IEEE Explore* и *Scopus*. Над обе базе је спроведен напредни упит са кључним терминима као што су *software testing, large language models, test generation* као и њихови синоними.

2.4. Критеријуми селекције радова

У оквиру критеријума укључивања, обухваћени су радови који испуњавају следеће:
Рад мора бити објављен у периоду од 2022. до 2025. године, како би се обухватио скуп радова који рефлектују тренутни ниво развоја LLM-а:

- Рад је написан на енглеском језику.
- Рад мора бити објављен као емпиријска студија, експеримент, студија случаја или евалуација.
- Рад се бави применом великих језичких модела у тестирању софтвера, при чему анализира његове перформансе, користи и ограничења.

Критеријуми искључења примењени су како би се одстраниле студије које нису релевантне за истраживачки фокус:

- Рад који се бави искључиво развојем или тренирањем LLM-а без њихове примене у тестирању софтвера.
- Рад истражује употребу LLM-а у другим доменима, попут анализе кода, верификације или природне обраде језика, без директне везе са генерисањем тестова.
- Теоријски радови или прегледи без емпиријских доказа и евалуација.

2.5. Процес селекције радова

На основу дефинисаног протокола спроведен је процес селекције литературе у две фазе. Најпре је извршена иницијална претрага база научних радова где су идентификоване 44 студије. Након уклањања 6 дупликата, спроведен је преглед наслова и сажетака, након чега је елиминисано још 11 радова. Приступ 4 рада није био могућ, те су из тог разлога искључени из даље студије. Укупно је 23 рада ушло у другу фазу. Друга фаза обухвата детаљан преглед самих радова. Како у овој фази ниједан рад није одбачен, прихваћено је 23 рада, који задовољавају све критеријуме и који ће служити за даљу анализу.

2.6. Процена квалитета студија

Процена квалитета спроведена је како би се осигурала поузданост и релевантност анализираних радова. Свака студија је оцењена на основу шест критеријума који се односе на: усмереност истраживања на примену великих језичких модела у тестирању софтвера, јасноћу и релевантност циљева, прикладност примењеног истраживачког приступа, јасноћу и анализу резултата, препознавање ограничења и њихових импликација, као и допринос практичним увидима у домену тестирања.

2.7. Екстракција података

Процес екстракције података дефинисан је тако да обезбеди структуриран и упоредив скуп информација за све анализиране радове. За сваку студију издвојено је осам категорија података: тип емпиријске студије (експеримент, студија случаја или евалуација), начин примене великог језичког модела у тестирању, конкретни модели коришћени у истраживању, идентификоване предности и ограничења, вредности параметра температуре, као и практичне препоруке, закључци и предлози за будућа истраживања. Ови подаци су послужили као основа за синтезу налаза и анализу представљену у наредном поглављу.

3. РЕЗУЛТАТИ И ДИСКУСИЈА

У овом поглављу детаљно су приказани резултати анализе у односу на раније дефинисана истраживачка питања.

3.1. ИП1: Како и који велики језички модели се користе у тестирању софтвера?

Велики језички модели све више привлаче пажњу као алат за аутоматизацију и унапређење различитих фаза развоја софтвера, укључујући генерисање јединичних тестова применом приступа учења из малог броја примера (енгл. *Few-shot learning*) или додатном обуком модела [1,2]. Досадашња истраживања указују да LLM-ови могу да постигну обећавајуће резултате у домену генерисања јединичних тестова [3]. Идентификован је широк спектар коришћених LLM-ова у студијама. Резултати показују јасну доминацију модела из породице GPT [1-14], посебно верзија GPT-

3.5, GPT-3.5 Turbo и GPT-4. Велику заступљеност имају специјализовани модели за разумевање кода, Codex [2,13,15-18] и StarCoder [2,13], док су модели отвореног кода, LlaMA 2 [3,4,19] и Mistral-7B [19,20], заступљени у мањем броју студија. У две студије модели нису јасно дефинисани [21,22]. У литератури се још помињу и BLOOM [13], CodeGen2 [13], CodeGPT [23], DeepSeek [11], InCoder [13], LlaMA [15], PALM-2 [4], StarCoder [13] и StarCoderBase [13].

Резултати показују да се велики језички модели најчешће користе за генерисање јединичних тестова, док су примене у домен-специфичном тестирању и репродукцији грешака у порасту. У домен-специфичним студијама модели се прилагођавају одређеним индустриским системима, попут контролних [7] и телекомуникационих [19] апликација, како би генерисали тестове за специфичан домен, док у развоју вођеним понашањем (енгл. *Behavior-Driven Development* – BDD) тестирање прихватљивости показује потенцијал LLM-ова у генерисању сценарија блиских корисничким захтевима. Табела 1 приказује дистрибуцију студија по начину употребе великих језичких модела у тестирању софтвера.

Табела 1. Дистрибуција радова по употреби

Категорија	**Број студија**
Јединични тестови	9
Домен-специфично тестирање	5
Репродукција грешака	4
Побољшање и исправка тестова	3
БДД тестови	1
АПИ тестирање	1

Велики језички модели се у тестирању софтвера углавном користе као помоћни алати за аутоматизацију постојећих процеса, при чему доминирају модели из GPT породице, док се отворена решења углавном примењују у експерименталним радовима, што указује да је област у фази технолошког сазревања. Ова доминација комерцијалних модела указује на ограничену доступност ресурса за истраживање и на зависност тренутних и будућих пројеката од затворених платформи, што представља методолошко ограничење за репродуктивност резултата.

3.2. ИП2: Које су предности и ограничења примене великих језичких модела у тестирању софтвера?

На основу налаза из свих анализираних студија, предности и ограничења примене великих језичких модела у тестирању софтвера могу се посматрати као две комплементарне димензије, а то су ефикасност и поузданост. Истраживања показују да су најчешће користи повезане са повећањем покривености кода [2,3,8-12,14-17,21], убрзањем тестних циклуса [1,18-20] и смањењем мануелног напора [4-7,13,23], чиме се значајно унапређује продуктивност процеса тестирања. Једна студија истиче корист у разумљивости и одрживости тестног кода [22], јер LLM-ови могу да анализирају постојеће тестове и предложе семантичке исправке. Насупрот томе, најчешће пријављена ограничења односе се на халуцинације и нетачне излазе [3-6,10,11,17,21], што доводи у питање репродуктивност и захтева људску валидацију резултата. Уочена је и ограничена генерализација [8,23] и зависност од формулације упита (енгл. *prompt dependency*) [18,19], што утиче на стабилност резултата и отежава поновљивост експеримената. Додатно, неколико студија наводи ограничења у сложеним сценаријима [1,18] и захтевност трошкова [2,20] као практичну баријеру за примену у индустрији. Велики језички модели тренутно највише доприносе унапређењу ефикасности и аутоматизације тестирања, али њихова поузданост остаје ограничена због нестабилности, халуцинација и недовољне генерализације. Њихова највећа вредност тренутно лежи у улози интелигентне подршке тестерима.

3.3. ИП3: Како различите конфигурације великих језичких модела утичу на квалитет и поузданост генерисаних резултата?

Различите конфигурације LLM-ова показују приметне разлике у квалитету и поузданости генерисаних тестова. Параметар температуре, као један од кључних параметара у процесу генерисања, контролише однос између детерминистичког понашања модела и разноврсности излаза, чиме директно утиче на стабилност, креативност и поузданост резултата.

Табела 2. Дистрибуција радова по температури

Категорија	**Број студија**
Без параметра температуре	16
Експериментална конфигурација	7
Подразумевана конфигурација	3

У највећем броју студија [1,3-9,12,14,17-20,22,23] тај параметар није наведен, што указује да истраживачи најчешће користе подразумеване поставке модела без експлицитне контроле овог параметра. Мањи број радова [13,16,21] експлицитно наводи подразумевану конфигурацију, али без детаљне анализе њиховог утицаја на квалитет излаза. Тиме се потврђује тренд ослањања на фабричке поставке комерцијалних модела. Неколико студија [2,10,11,13,15,16,21] спровело је експерименте са различитим вредностима, што представља изузетак у односу на целокупан корпус. Њихови резултати указују да ниже вредности доводе до веће конзистентности, мање халуцинација и тачнијих тестова, док више вредности повећавају разноврсност и потенцијал за откривање нових сценарија, али истовремено указују на већу стохастичност и појаву синтаксних или логичких грешака. Свеукупно, налази показују да ефекат температуре остаје недовољно емпиријски испитан, а заједница још није успоставила јасне смернице за оптимално подешавање конфигурационих параметара.

3.4. Препоруке за будућа истраживања

Резултати спроведеног прегледа указују да велики језички модели имају значајан потенцијал за унапређење процеса тестирања софтвера, али њихова примена захтева контролу и стандардизацију. Модели се показују најделотворнијим када се користе као

помоћни алати, уз људску валидацију генерисаних тестова. Посебну пажњу потребно је усмерити на систематско испитивање конфигурационих параметара, где параметар температуре контролише однос између детерминистичког и стохастичког понашања модела. Препоручује се експериментисање са нижим вредностима температуре у задацима који захтевају стабилност и тачност, док се више вредности могу користити у контексту проналажења разноврсних излаза. Даља истраживања треба да укључе поређење између комерцијалних и отворених модела. Коначно, неопходно је посветити више пажње квалитативним аспектима тестова, разумљивости, прецизности, одрживости у практичној употреби у реалним окружењима. Развој стандардизованих метрика представља предуслов за поуздано поређење резултата различитих студија.

4. ЗАКЉУЧАК

Налази овог истраживања указују да велики језички модели представљају значајан корак ка интелигентној аутоматизацији тестирања софтвера, али још увек нису достигли ниво поузданости неопходан за потпуну самосталност у овом процесу. Њихова највећа вредност тренутно лежи у улози подршке, где доприносе повећању ефикасности, покривености и брзини тестирања, док људска интервенција остаје неопходна ради валидације и контроле резултата. Доминација комерцијалних модела, попут GPT породице, потврђује њихову супериорност у квалитету генерисаних тестова, али истовремено указује на ограничену репродуктивност и зависност од затворених платформи. Отворени модели попут LlaMA и Mistral за сада показују потенцијал углавном у експерименталним сценаријима. Иако су предности LLM-ова најизраженије у домену повећања ефикасности, идентификована ограничења, халуцинације, недовољна генерализација и осетљивост на формулацију упита представљају кључне изазове за њихову ширу примену. Подаци показују да је параметар температуре ретко експлицитно анализиран, што упућује на недостатак систематских истраживања. Управо овај аспект може бити пресудан за развој поузданих и контролисаних LLM решења у тестирању. Свеукупно посматрано, велики језички модели данас представљају важан покретач промена у начину на који се софтвер тестира. Уместо да замене постојеће методе, они их надопуњују, убрзавају процес тестирања, смањују потребу за ручним радом и омогућавају брже откривање грешака. Њихов тренутни значај лежи у подршци стручњацима, а не у потпуној аутономији. Да би се тај потенцијал у потпуности искористио, потребна су даља емпиријска истраживања која ће се бавити стабилношћу, подешавањем параметара и поузданошћу резултата. Развој стандардизованих поступака и метрика представља неопходан корак ка томе да LLM-ови постану поуздан и интегрисан део процеса тестирања софтвера.

5. ЛИТЕРАТУРА

[1] R. F. de L. Junior, L. F. P. de B. Presta, L. S. Borborema, V. N. da Silva, M. L. de M. Dahia, and A. C. S. e Santos, “A Case Study on Test Case Construction with Large Language Models: Unveiling Practical Insights and Challenges,” Dec. 2023, [Online]. Available: http://arxiv.org/abs/2312.12598

[2] M. Schäfer, S. Nadi, A. Eghbali, and F. Tip, “An Empirical Evaluation of Using Large Language Models for Automated Unit Test Generation,” Dec. 2023, [Online]. Available: http://arxiv.org/abs/2302.06527

[3] Y. Chen, Z. Hu, C. Zhi, J. Han, S. Deng, and J. Yin, “ChatUniTest: A Framework for LLM-Based Test Generation,” in *FSE Companion - Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, Association for Computing Machinery, Inc, Jul. 2024, pp. 572–576. doi: 10.1145/3663529.3663801.

[4] S. Karpurapu *et al.*, “Comprehensive Evaluation and Insights Into the Use of Large Language Models in the Automation of Behavior-Driven Development Acceptance Test Formulation,” *IEEE Access*, vol. 12, pp. 58715–58721, 2024, doi: 10.1109/ACCESS.2024.3391815.

[5] G. Yi, Z. Chen, Z. Chen, W. E. Wong, and N. Chau, “Exploring the Capability of ChatGPT in Test Generation,” in *Proceedings - 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion, QRS-C 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 72–80. doi: 10.1109/QRS-C60940.2023.00013.

[6] A. Lops, F. Narducci, A. Ragone, and M. Trizio, “AgoneTest: Automated creation and assessment of Unit tests leveraging Large Language Models,” in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*, Association for Computing Machinery, Inc, Oct. 2024, pp. 2440–2441. doi: 10.1145/3691620.3695318.

[7] H. Koziolek, V. Ashiwal, S. Bandyopadhyay, and C. K. R, “Automated Control Logic Test Case Generation using Large Language Models,” May 2024, [Online]. Available: http://arxiv.org/abs/2405.01874

[8] A. Guo *et al.*, “SoVAR: Build Generalizable Scenarios from Accident Reports for Autonomous Driving Testing,” in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*, Association for Computing Machinery, Inc, Oct. 2024, pp. 268–280. doi: 10.1145/3691620.3695037.

[9] Z. Wang, K. Liu, G. Li, and Z. Jin, “HITS: High-coverage LLM-based Unit Test Generation via

Method Slicing," Aug. 2024, [Online]. Available: http://arxiv.org/abs/2408.11324

[10] T. Le, T. Tran, D. Cao, V. Le, T. Nguyen, and V. Nguyen, "KAT: Dependency-aware Automated API Testing with Large Language Models," Jul. 2024, doi: 10.1109/ICST60714.2024.00017.

[11] L. Yang *et al.*, "On the Evaluation of Large Language Models in Unit Test Generation," in *Proceedings - 2024 39th ACM/IEEE International Conference on Automated Software Engineering, ASE 2024*, Association for Computing Machinery, Inc, Oct. 2024, pp. 1607–1619. doi: 10.1145/3691620.3695529.

[12] T.-O. Li *et al.*, "Nuances are the Key: Unlocking ChatGPT to Find Failure-Inducing Tests with Differential Prompting," Sep. 2023, [Online]. Available: http://arxiv.org/abs/2304.11686

[13] S. Kang, J. Yoon, N. Askarbekkyzy, and S. Yoo, "Evaluating Diverse Large Language Models for Automatic and General Bug Reproduction," Nov. 2023, [Online]. Available: http://arxiv.org/abs/2311.04532

[14] S. Bhatia, T. Gandhi, D. Kumar, and P. Jalote, "Unit Test Generation using Generative AI : A Comparative Performance Analysis of Autogeneration Tools," Feb. 2024, [Online]. Available: http://arxiv.org/abs/2312.10622

[15] A. M. Dakhel, A. Nikanjam, V. Majdinasab, F. Khomh, and M. C. Desmarais, "Effective Test Generation Using Pre-trained Large Language Models and Mutation Testing," Aug. 2023, [Online]. Available: http://arxiv.org/abs/2308.16557

[16] C. Lemieux, J. Priya Inala, S. K. Lahiri, and S. Sen, "CODAMOSA: Escaping Coverage Plateaus in Test Generation with Pre-trained Large Language Models." [Online]. Available: https://github.com/microsoft/codamosa.

[17] K. El Haji, C. Brandt, and A. Zaidman, "Using GitHub Copilot for Test Generation in Python: An Empirical Study," in *Proceedings - 2024 IEEE/ACM International Conference on Automation of Software Test, AST 2024*, Association for Computing Machinery, Inc, Apr. 2024, pp. 45–55. doi: 10.1145/3644032.3644443.

[18] S. Kang, J. Yoon, and S. Yoo, "Large Language Models are Few-shot Testers: Exploring LLM-based General Bug Reproduction," Jul. 2023, [Online]. Available: http://arxiv.org/abs/2209.11515

[19] M. Nabeel, D. D. Nimara, and T. Zanouda, "Test Code Generation for Telecom Software Systems using Two-Stage Generative Model," Apr. 2024, [Online]. Available: http://arxiv.org/abs/2404.09249

[20] J. Hoffmann and D. Frister, "Generating Software Tests for Mobile Applications Using Fine-Tuned Large Language Models," in *Proceedings - 2024 IEEE/ACM International Conference on Automation of Software Test, AST 2024*, Association for Computing Machinery, Inc, Apr. 2024, pp. 76–77. doi: 10.1145/3644032.3644454.

[21] N. Alshahwan *et al.*, "Automated Unit Test Improvement using Large Language Models at Meta," in *FSE Companion - Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering*, Association for Computing Machinery, Inc, Jul. 2024, pp. 185–196. doi: 10.1145/3663529.3663839.

[22] A. Deljouyi, "Understandable Test Generation Through Capture/Replay and LLMs," in *Proceedings - International Conference on Software Engineering*, IEEE Computer Society, Apr. 2024, pp. 261–263. doi: 10.1145/3639478.3639789.

[23] L. Plein, W. C. Ouédraogo, J. Klein, and T. F. Bissyandé, "Automatic Generation of Test Cases based on Bug Reports: A Feasibility Study with Large Language Models," in *Proceedings - International Conference on Software Engineering*, IEEE Computer Society, Apr. 2024, pp. 360–361. doi: 10.1145/3639478.3643119.

Кратка биографија:



Давид Фејеш рођен је у Вршцу 1999. год. Основне академске студије на Факултету техничких наука из области Инжењерство информационих система завршио је 2023. год. Исте године уписује мастер студије на истом смеру.
Контакт: david.fejes@gmail.com