UDK 004.415.53

Metamorphic testing for generative artificial intelligence systems

Iakusheva S.F. (Moscow Institute of Physics and Technology)

Khritankov A.S. (Moscow Institute of Physics and Technology, Higher School of Economics University)

Harbachonak D.I. (Higher School of Economics University)

Generative artificial intelligence systems should be carefully tested because they can generate low-quality content, but the testing challenges the test oracle problem. Metamorphic testing helps to test programs without test oracles. In this study, we consider an AI system that generates personalized stickers. Personalized sticker is a thematic picture with a person's face, some decorative elements and phrases. This system is stochastic, complex and consists of many parts. We formulate requirements for the whole system and check them with metamorphic relations. The requirements focus on dependency on input images quality and resulting quality of generated stickers. Our testing methodology helps us to identify major issues in the system under test.

Keywords: metamorphic testing, generative artificial intelligence systems, quality control

1. Introduction

Nowadays artificial intelligence (AI) systems are commonly used for content generation, for example, Midjourney project and other diffusion neural networks [12]. Generative AI systems affect the users, and such content may have inappropriate quality or even be harmful. So, verification of the requirements for these systems should be precise.

One of the challenges is the test oracle problem [2]: difficulty or impossibility to compute if the program's output is correct. Generative AI can produce a lot of different items for one generation request, and every item may be called correct, so we only can check some metrics or use human supervision. Thus, verification of the quality requirements is complicated.

There are some methods to verify programs without test oracles, one of them is metamorphic testing [3]. In this study, we apply metamorphic testing to a generative AI system for picture generation. We consider an image-generation system [7], formulate two requirements, propose two metamorphic relations, test the system and find major issues.

The article has the following structure. Section 2 provides the background on the problems of testing AI systems. Section 3 describes the system under test. Section 4 gives metamorphic relations for the system. Then Section 5 describes the experiment. Section 6 concludes our work.

2. Related work

Many methods for image generation exist [6], for example: stylization, style transfer and animation, image composition, generating images from text (Adversarial Networks (GANs), Transformers and Diffusion Models). All these methods have their specific advantages and limitations.

There are many validation approaches for AI [13]: classification-based [8], model-based, learning-based [9], rule-based [4] and non-oracle (metamorphic [15]). Metamorphic testing [3] is one of the testing methods for programs with the test oracle problem. It allows us to easily generate test inputs and automate the test process. This method uses so-called metamorphic relations instead of verifying every test output or calculating metrics.

The metamorphic relation for a program can be represented as a function

$$R(x_1, x_2, \ldots, x_n, f(x_1), f(x_2), \ldots, f(x_n)) \longrightarrow \{0, 1\}, \tag{1}$$

where $n \geq 2$ is the total number of test inputs, x_i as a i-th program input and $f(x_i)$ is a i-th output.

Algorithm of metamorphic testing is quite simple. The system is run on several test inputs, for which some relation holds. Then, we check the presence of the corresponding relation between the test outputs. Thus, we consider the evolution in the system response when we change the input data. There is no need for checking every single test output for correctness.

Metamorphic testing is used for testing different stochastic AI systems like machine translators [10], web systems [1], image classifiers [5], image recognition systems [16], etc. Metamorphic testing is also used for testing stochastic systems. In this case, statistical methods are used for verifying the relations (for example, criteria [1], correlation [17], ANOVA [10]).

3. System under test

We consider a sticker-generating system (SGS) [7] as a system under test (SUT). We call a "sticker" a thematic picture with a person's face, some decorative elements and maybe praising phrases on it (Fig. 1). For example, a sticker can be presented to a user after finishing an online course. However, some input images cannot be transformed into stickers because they lack some parts of the face, have poor quality, etc. In this situation SGS does not generate anything and raises exceptions.

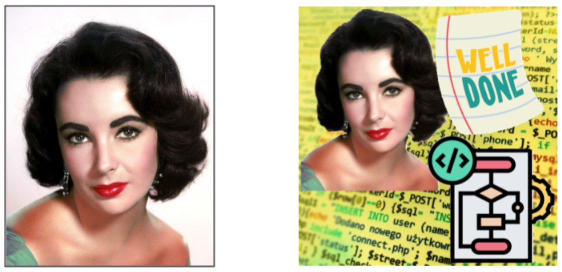

Fig. 1. Original image and a “programming” thematic sticker.

Previously [7], this system was compared with the simple basic solution. The users decided that generated stickers were more interesting and aesthetically pleasing. Unfortunately, this method is not automatic.

SGS contains many components: a neural network (NN) for face segmentation (MediaPipe), a NN for determining the topic of the sticker description (SpaCy), and auxiliary tools for blending, adding glasses, multification (AnimeGAN), style transfer and quality control.

Of course, NN components of SGS can be tested with metamorphic testing individually. But the whole system is much more complex, and good quality of the components does not guarantee the good quality of the system. Our purpose is to verify requirements for the whole system. So, we consider it as a “black box” and do not use its internal structure.

4. Proposed method

4.1. Requirements and Metamorphic relations

Let us consider SUT as a stochastic function $F(x, t)$ where x denotes an input image and t denotes a text description.

We choose two requirements from different areas to test the SUT.

The first requirement relates to the user’s needs. Any user should recognize his or her face on the sticker. Also, this sticker can be used as a profile photo. So, SUT should preserve the face in such a way that the face on the sticker should look almost the same as the face on the original image. So, if we use the sticker as an input for generating a new sticker, this new sticker should be generated. We interpret this as a

Requirement 1: *every sticker is an ideal input and new stickers can be produced from every sticker.*

So, we formulate metamorphic relation:

MR1: $count(F(x_i, t)) = count(F(F(x_i, t), t))$, $1 \leq i \leq n$

which means that the number of successfully generated stickers is not reduced after repeated generation.

The second requirement origins in mathematical models of SUT components. Segmentation NN should work worse on images with the poor quality (e.x. with noise and low contrast). We can decrease the quality of the image until SUT rejects it. And if we decrease the quality even more, this image also will be rejected by SUT. So, image quality should affect the number of generated stickers.

Requirement 2: *the worse the image quality, the less the number of successfully generated stickers.*

If we sequentially degrade the input images' quality, the number of successfully generated stickers should decrease monotonously.

MR2: $count(F(x_{i,0}, t)) \geq count(F(x_{i,1}, t) \geq \cdots \geq count(F(x_{i,m}, t))$, $1 \leq i \leq n$.

Due to the stochasticity, the monotonicity may be violated. So, we use a statistical criterion to verify it. We check MR2 using B-spline interpolation method [14]. We slightly modify it (https://gitlab.com/mlrep/mldev-metamorphic).

4.2. Experiment

We conduct an experiment in order to check MR1 and MR2.

For the MR1 we use 36 open-source portrait images of different ages and races. We try to create 100 stickers from every image and then try to create a sticker from every generated sticker. The number of stickers is determined automatically as the number of generated files. After that, we compare the number of generated stickers and number of stickers generated from stickers.

For the MR2 we use the same 36 images. We repeatedly apply gaussian noise to get a sequence of 30 gradually blurred images. Then, we try to generate 30 stickers from each image. We apply a criterion of monotonicity for each sequence to check if it decrease monotonously.

4.3. Results

Table 1 represents the result. MR1 fails completely and MR2 fails in a small percent of cases. This means that the system does not preserve faces well, but almost corresponds it's mathematical model.

Table 1

Results for SUT.

MR	Proportion of fulfilled MRs	Proportion of failed stickers
MR1	0.07	0.16
MR2	0.97	-

The experiment shows that SUT has major problems with face preserving and some minor issues with dependency on image quality. We can assume that NN components of the system work well, but the system in general has quality problems. So, proposed metamorphic relations are useful for testing and detecting problems.

4.4. Other observations

Our tests show that SUT changes input images a lot. So, if we use the output sticker as input, the second output will look less like the original image. If we repeat this many times, we will get an image with an unrecognizable face on it. Example of this transformation is provided on Figure 2.

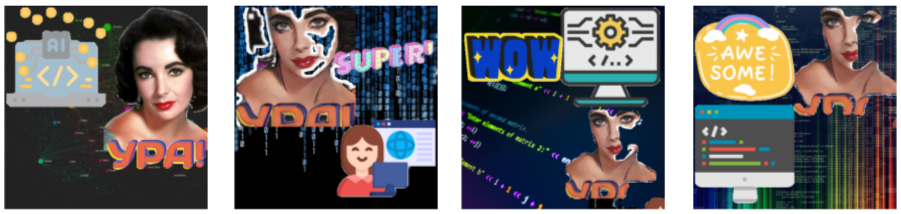

Fig. 2. 4 steps to get an image with unrecognizable face

Table 2 provides information about a small experiment with an image on Figure 1. We take the original image and lower its quality with three different methods. Then, we repeat sticker generation until possible.

This experiment shows that after 7 or 8 iterations all the stickers contain unrecognizable faces. So, we can estimate the number of steps to get such images. If we decrease image

Table 2

Number of stickers which are successfully generated from stickers

Number of repetitions	1	2	3	4	5	6	7	8	9
Original	100	79	39	21	7	1	0		
Darkened	100	82	37	16	6	2	1	0	
Low-contrast	100	81	36	15	5	2	0		
White-black	100	81	37	16	7	2	1	1	0

quality (blur, change the contrast, add color filter, etc.), the number of steps will decrease. So, potentially such series can be used for metamorphic testing too.

5. Conclusion

In this paper, we apply metamorphic testing to a generative AI system. We study the problem of AI testing, choose a generative system, propose two metamorphic relations, test the system and detect some major issues. Our metamorphic relations are derived from different groups of requirements: user and technical. Thus, we show that a metamorphic testing method is useful for verifying the requirements and testing such systems.

In this paper, we consider a relatively simple sticker generation system. Application to more complex systems could be a direction of future work.

References

1. Ahlgren J. et al. Testing Web Enabled Simulation at Scale Using Metamorphic Testing. In 2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP) //IEEE, 140š149. – 2021.
2. Barr E. T. et al. The oracle problem in software testing: A survey //IEEE transactions on software engineering. 2014. Vol. 41, №. 5. P. 507–525.
3. Chen T. Y. et al. Metamorphic testing: A review of challenges and opportunities //ACM Computing Surveys (CSUR). 2018. Vol. 51, №. 1. P. 1–27.
4. Deason W. H. et al. A rule-based software test data generator //IEEE transactions on Knowledge and Data Engineering. 1991. Vol. 3, №. 1. P. 108–117.
5. Dwarakanath A. et al. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing //Proceedings of the 27th ACM SIGSOFT international symposium on software testing and analysis. 2018. P. 118–128.
6. Harbachonak, D.: Issledovaniye metodov mashinnogo obucheniya dlya sozdaniya personalizirovan-

nyh stikerov (Study of Machine Learning Methods for Personalized Stickers Creation). Proceedings of 65th MIPT Conference, Applied Mathematics and Informatics. 2023.

7. Harbachonak, D.: Issledovaniye metodov mashinnogo obucheniya dlya sozdaniya personalizirovannyh stikerov (Study of Machine Learning Methods for Personalized Stickers Creation). B.Sc. thesis, HSE University, Moscow (2023).
8. Last M., Friedman M., Kandel A. The data mining approach to automated software testing //Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. 2003. P. 388–396.
9. Meinke K., Niu F. A learning-based approach to unit testing of numerical software //IFIP International Conference on Testing Software and Systems. Berlin, Heidelberg : Springer Berlin Heidelberg. 2010. P. 221–235.
10. Pesu D. et al. A Monte Carlo method for metamorphic testing of machine translation services //Proceedings of the 3rd International Workshop on Metamorphic Testing. 2018. P. 38–45.
11. ur Rehman F., Izurieta C. Statistical Metamorphic Testing of Neural Network Based Intrusion Detection Systems //2021 IEEE International Conference on Cyber Security and Resilience (CSR). IEEE, 2021. P. 20–26.
12. Rombach R. et al. High-resolution image synthesis with latent diffusion models //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022. P. 10684–10695.
13. Tao C., Gao J., Wang T. Testing and quality validation for ai software–perspectives, issues, and practices //IEEE Access. 2019. Vol. 7. P. 120164–120175.
14. Wang J. C., Meyer M. C. Testing the monotonicity or convexity of a function using regression splines //Canadian Journal of Statistics. 2011. Vol. 39, №. 1. P. 89–107.
15. Xie X. et al. Testing and validating machine learning classifiers by metamorphic testing //Journal of Systems and Software. 2011. Vol. 84, №. 4. P. 544–558.
16. Zhang Z. et al. Deepbackground: Metamorphic testing for deep-learning-driven image recognition systems accompanied by background-relevance //Information and Software Technology. 2021. Vol. 140. P. 106701.
17. Zhou Z. Q., Tse T. H., Witheridge M. Metamorphic robustness testing: Exposing hidden defects in citation statistics and journal impact factors //IEEE Transactions on Software Engineering. 2019. Vol. 47, №. 6. P. 1164–1183.