Semantic Data Augmentation for Deep Learning Testing using Generative AI

Sondess Missaoui
Department of Computer Science
University of York, York, UK
sondess.missaoui@york.ac.uk

Simos Gerasimou
Department of Computer Science
University of York, York, UK
simos.gerasimou@york.ac.uk

Nicholas Matragkas
Université Paris-Saclay, CEA, List
Paris, France
nikolaos.matragkas@cea.fr

***Abstract*—The performance of state-of-the-art Deep Learning models heavily depends on the availability of well-curated training and testing datasets that sufficiently capture the operational domain. Data augmentation is an effective technique in alleviating data scarcity, reducing the time-consuming and expensive data collection and labelling processes. Despite their potential, existing data augmentation techniques primarily focus on simple geometric and colour space transformations, like noise, flipping and resizing, producing datasets with limited diversity. When the augmented dataset is used for testing the Deep Learning models, the derived results are typically uninformative about the robustness of the models. We address this gap by introducing GENFUZZER, a novel coverage-guided data augmentation fuzzing technique for Deep Learning models underpinned by generative AI. We demonstrate our approach using widely-adopted datasets and models employed for image classification, illustrating its effectiveness in generating informative datasets leading up to a 26% increase in widely-used coverage criteria.**

***Index Terms*—Generative AI, Deep Learning Testing, Coverage Guided Fuzzing, Data Augmentation, Safe AI**

I. INTRODUCTION

The tremendous progress achieved by Deep Learning (DL) in several real-world challenging tasks like image classification [1], object detection [2] and natural language processing [3] led to its exponential adoption in various application domains, including medical diagnostics [4], autonomous driving [5] and infrastructure inspection [6]. A key ingredient in achieving these impressive results is the availability of large volumes of high-quality annotated datasets that adequately encode the characteristics of the target domain [7]. Within a typical DL pipeline, this data enables the construction of high-dimensional representations during training, and instruments the robustness and generalisability assessment during testing [8].

Data scarcity poses a major challenge in devising robust and competitive DL models [9]. This is particularly important in domains such as healthcare or security, where relevant data is not typically available because of privacy considerations or simply because such data does not exist. Data augmentation aims at alleviating this issue by intelligently synthesising new data informed by the available dataset. Compared to the manual creation of labelled datasets involving human experts, data augmentation is time-efficient and cost-effective [10].

Driven by traditional software engineering principles, data augmentation in DL testing is increasingly adopted to improve the diversity of the test set and achieve higher testing coverage [11]. Notwithstanding its merits, conventional augmentation techniques are limited to noise injection or to the application of content-preserving geometric and colour-space transformations, e.g., flipping, cropping, rotation, translation, [8], [12], [13]. These techniques cannot perform semantic transformations, like altering the content of an input [14], thus producing testing suites that, albeit yield higher coverage numerical results, are of low quality and lack semantic diversity [9].

Inspired by the emergence of generative AI models for input synthesis using latent representations [15], we posit that leveraging these models can inform the generation of diverse and realistic synthetic inputs that capture the underlying variability of the data distribution [16]. More specifically, generative AI models like stable diffusion can automatically learn the natural features and latent representations, and generate realistic images from textual prompts to create rich and diverse visual content with unprecedented precision [15].

We introduce GENFUZZER, a novel coverage-guided fuzzing approach for producing semantically-diverse test inputs for DL testing. Our approach uses inputs (seeds) from a dataset, including the ground truth and contextual information, and performs semantic data augmentation by extracting the mask of the seed and producing a textual prompt. These are both used by generative AI models (stable diffusion [15] in its current version) to conditionally fill the mask with a synthesised image conforming to the textual prompt. Then, by executing fidelity analysis and coverage tracing, GENFUZZER keeps only synthesised images whose fidelity scores exceed a given threshold (i.e., they are photorealistic) and increase the selected coverage criterion (i.e., they contribute to higher coverage), respectively. GENFUZZER contributes to addressing two main problems in DL testing: (i) automated test case generation through the semantic data augmentation method; and (ii) coverage-guided fuzzing by integrating a new key component for fidelity estimation of candidate test cases.

II. BACKGROUND AND RELATED WORK

While software testing [17] follows a clear methodology involving validation and verification, Deep Learning (DL) testing is more complex due to the challenge of establishing precise system specifications against which the DNN model behaviour can be checked [11]. A growing body of DL testing

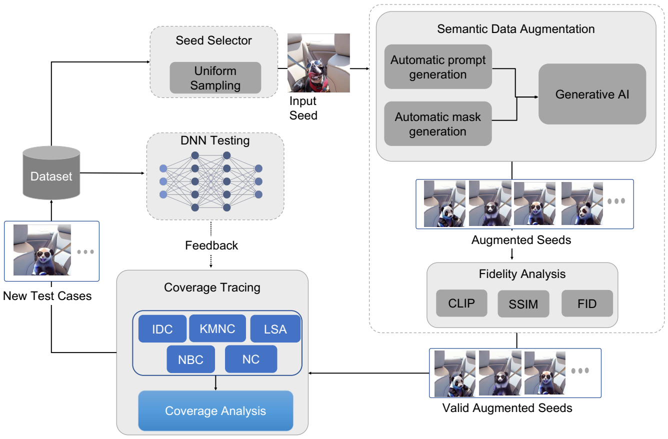

Figure 1: GenFuzzer workflow.

research has evolved that leverages and adapts techniques such as code coverage, test oracle, and coverage-guided fuzzing (CGF) from software testing. This line of research focuses on white box testing, proposing test adequacy criteria, e.g., neuron coverage (NC) [18], DeepGauge [19], DeepGini [20], DeepImportance [21], and analysis techniques [22] to improve the DNN model quality. Furthermore, CGF principles have been applied successfully to DL testing aiming at identifying bugs, i.e., erroneous behaviour, in real systems [23]. DeepHunter [24], DeepTest [13] and TensorFuzz [12] are fuzzing techniques that generate new tests cases through metamorphic transformations with the intention that the new test and its original (seed) share the same semantics from a human perspective. These approaches instrument mutations that encode possible real-life errors through simple geometric and colour space transformations, e.g., contrast, blurring, fog effect [25]. While these mutations are realistic, they are encoded as domain-specific metamorphic relations and lack sufficient semantic diversity to further extend the scope of testing. This gap is addressed by our new data augmentation GenFuzzer approach.

III. GenFuzzer

GenFuzzer (Fig. 1) is a coverage-guided fuzzing (CGF) approach that can enhance the size, quality, and semantic diversity of datasets such that extensible DNN testing can be performed. Using a dataset T, and a DNN model D trained on T, our approach produces synthetically augmented images using generative AI (e.g., inpainting diffusion models [26]). Then, it carries out a fidelity analysis step to select images that are to a sufficient level, photo-realistic, and effectively augment the dataset's feature space. Finally, GenFuzzer deploys extensible testing criteria as feedback to guide the selection of augmented images that enhance DNN testing. We run the trained DNN against the newly generated tests; test cases that increase coverage are kept. These test cases simulate real-world scenarios and are used to evaluate DNN robustness.

A. Problem Formulation

Semantic-based coverage-guided fuzzing for DNNs can be formulated as the problem of generating synthetic examples $x' = \mathcal{A}(x)$ that are semantically within the data domain $\mathcal{X}$, but with enough perturbation to enhance its feature space. This can be understood as adding ϵ perturbations to the original example x, and adding a large perturbation ε with new characteristics to the data domain $\mathcal{X}$. The objective is to minimize the overall perturbation ε and maximize the adequacy criterion λ, i.e., the coverage score $\mathcal{C}ov(x')$, so that the synthetic sample x' belongs to the in-domain distribution of $\mathcal{X}$. This problem can be encoded in the objective function:

$$E_{x\sim\mathcal{X}}\left[\min_{\varepsilon}\mathcal{A}(x+\epsilon), \max_{\lambda}\mathcal{C}ov(x')\right] \qquad (1)$$

B. Seed Selection & Semantic Data Augmentation

In order to solve the optimization problem defined in (1) effectively and efficiently, we deploy text-conditional generative AI model [26]. Let D^j be our original trained DNN model on a training dataset T^r, and T^t be the testing set, both of which belong to the same data distribution domain. $SS(.)$ is the seed selection strategy that samples and selects input seeds based on a random uniform sampling strategy with sampling probability $P_i \in \mathcal{P}$, where $\mathcal{P}$ defines the sampling probability of all samples $x_i \in T^t$. Note that we can use T^r or T^t for seed sampling. Let $\mathcal{A}_\kappa$ be our augmentation technique for D^j with κ hyper-parameters. $\mathcal{A}_\kappa$ is a data-level mutation operator (as conventionally named in GCF) that targets the testing data T^t, by augmenting it with a set of new test cases $\mathcal{T}'$ to obtain the augmented testing set $T^t_\kappa = T^t \cup \mathcal{T}'$. Using the input image x_i in the test set as a reference (input seed), we represent the data augmentation technique as: $\mathcal{A}_\kappa(x_i) = Aug\left(x_i, \eta\right), x_i = SS\left(T^t, \mathcal{P}\right)$ where $Aug(.)$ is the generative AI augmentation technique and η are the specific parameters for the given technique. Our approach currently supports text-conditional generative AI [26] that enables editing specific parts of an image by providing a mask and a text prompt to generate augmented seeds automatically. In particular, diffusion model inpainting can be performed by sampling and replacing the known region (i.e., mask) of the image with a sample from data distribution $\mathcal{T}^t$. It is important to note that our approach is generic and can support different generative models, including Glide [27], eDiffi [28] and Imagic [29]. We also emphasize the importance of using a text-conditional generative model as it has the advantage of providing a control mechanism through textual prompts to adjust the mutations introduced to the original image. GenFuzzer automatically generates a mask and prompt for each input seed image x_i. To do so, we apply Mask R-CNN [30] for instance segmentation and use the image corresponding class c_i to select a mask m_i of the main object in x_i. For the text prompt t_i, we use natural language processing [31] to generate a prompt where the target object name is used to extract the named entity from within the input seed caption/description y_i and replace it with a newly selected named entity. The generative AI model consumes this information and replaces the masked region of the input seed image with the target object $c_i^{'} \in C \setminus \{c_i\}$, generating an augmented seed x_i^κ, where C represents the data classes of

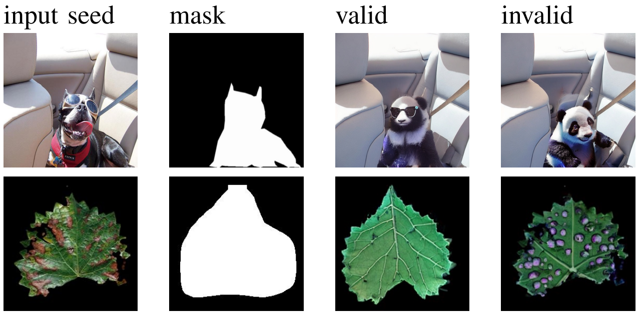

Figure 2: Inputs synthesised using GenFuzzer

T^t. This is repeated for all $C \setminus \{c_i\}$. Thus, for text-conditional generative AI $\eta = (m_i, y_i, t_i)$. Fig. 2 shows two examples of how the process is performed. The first example illustrates the input seed x_i (first row) with a caption y_i =*"a **dog** sitting in the back seat of a car with sunglasses on"*, the class $c_i = dog$ and the generated prompt t_i=*"a high-fidelity image of a **panda** sitting in the back seat of a car with sunglasses on*", where we replaced the main object dog with a panda.

C. Fidelity Estimation

As not all augmented seeds correspond to realistic images, our approach estimates the visual or textual fidelity of each augmented seed x_i^κ, retaining only those surpassing a predefined fidelity threshold given by:

$$\text{isValid}(x_i^\kappa, Q) = \begin{cases} True & \text{if } f(x_i^\kappa, Q) \geq \delta \\ False & otherwise \end{cases} \quad (2)$$

where $f(.)$ denotes the employed fidelity assessment technique. For visual fidelity, $Q = x_i$, i.e., the input seed, in which case we use FID [32] or SSIM [33]. For textual fidelity assessment, $Q = t_i$, i.e., the text prompt, and we use CLIP [34]. FID and SSIM are quantitative measures that identify the acceptable deviation between the original and mutated images. CLIP is widely used to guide image generation through textual input. δ is a domain-specific threshold experimentally defined for each assessment measure individually. This step involves the initial part of equation 1, aiming to minimize the overall perturbation by creating high-fidelity images with minimal alterations to the original input. It focuses solely on the object's mask c_i, leading to the creation of a set of valid augmented seeds.

D. Coverage Tracing

Our approach uses coverage guidance to filter valid augmented seeds and identify new test cases $\mathcal{T}'$. GenFuzzer keeps seeds that bring new semantic information to the testing set and adds them to the fuzzing queue. The semantic diversity of these seeds is quantified based on their ability to maximize selected coverage criteria when added to the test set T^t (latter part in (1)). $\mathcal{T}'$ largely increases the fuzzing effectiveness and enables adding the informative inputs into the test set. Several test adequacy criteria exist for analysing the inner behaviours of DNN models and providing feedback metrics for our fuzzer to determine $\mathcal{T}'$. GenFuzzer currently supports neuron coverage (NC) [18], DeepImportance's IDC [21], likelihood-based surprise adequacy (LSA) [35], and the neuron-level criteria k-Multisection Neuron Coverage (KMNC) and Neuron Boundary Coverage (NBC) from DeepGauge [19]. Each metric uses a specific test adequacy criterion that identifies the parts of DL logic exercised by a given test set.

Table I: Datasets and DNN models used in our experiments

Dataset	DNN model	#Layers	Accuracy
COCO [36]	Vgg19 [37]	19	82.04%
CIFAR-10 [38]	LeNet5 [2]	5	77.20%
Leaves[1]	All-ConvNet [39]	18	97.73%

IV. Evaluation

A. Experimental Setup

We have implemented GenFuzzer as a self-contained fuzz testing framework in Python based on the open-source machine learning framework Keras with Tensorflow (v2.6) backend. We extensively evaluate GenFuzzer on 3 different DNNs trained on the datasets described in Table I. The COCO dataset was filtered, and only images belonging to the superclass 'animal' were selected, resulting in 10 classes with 250 samples each.

Hyper-parameter analysis was carried out to select the optimal threshold δ for each fidelity assessment metric, aiming to identify photorealistic seeds. δ was set to $0.8, 0.6$ for CLIP and SSIM (higher is better), respectively. After normalization between $[0, 1]$, the FID threshold was set to 0.2 (lower is better). Concerning the coverage criteria, we used for each approach the hyper-parameters recommended in its original research paper. We set the threshold for NC to 0.7. For KMNC, the k value, i.e., the number of multisections is set to 10. For LSA, we manually choose the layer in each of the DNN models. We deployed Gaussian noise to create a new fuzzer "*Random-Noise*" for the comparative study GenFuzzer denoted 'Random-Inpainting'. Gaussian noise is a metamorphic technique that adds white noise to x_i by adding a random number to every colour channel of each pixel. It has the mean μ and standard deviation σ of the random noise as η parameters.

B. Results and Discussion

We have performed a set of experiments to demonstrate the usability of semantic data augmentation, i.e., using text-conditional generative AI for fuzz testing. We instantiated our ideas and answered two main questions.

RQ1 (**Photo-realism**): *How effective is* GenFuzzer *in generating synthetic images that are semantically meaningful?* Assessing the fidelity of the synthesised inputs (step III-C) is key in establishing the effectiveness of GenFuzzer. To answer **RQ1**, we perform a large-scale controlled study using three image datasets (Table I). This experiment is designed to quantitatively assess the quality of the augmented images. For each selected input seed, the fuzzer mutates $|C| - 1$ times (with C representing the data classes), and we deduct 1 to count for the input seed class. Each augmented seed is validated against the input seed and text prompt. Only those maximising the fidelity scores are kept. Table II shows the ratio of valid synthetic images generated by different strategies identified through the Fidelity Estimation step. Overall, according to SSIM, the

Table II: (%) Valid images generated by Stable Diffusion, according to fidelity scores FID and SSIM

	SSIM	FID	# augmented seeds
COCO	49.44%	92.0%	2750
Leaves	58.54%	89.04 %	1260
CIFAR 10	28.88%	8.11%	1800

Table III: The ratio of valid images according to CLIP.

Strategies	Coco	Leaves	CIFAR 10
Stable Diffusion	1596 (±58.0%)	1232 (97.78%)	226 (20.56%)
Gaussian Noise	450 (50%)	277 (22%)	7 (0.67%)

fidelity rates of COCO and Leaves datasets are generally higher than CIFAR-10 for both augmentation techniques, e.g., 49.44% of augmented seeds are evaluated as valid for COCO, while only 28.88% are valid augmented seeds for CIFAR-10. The FID metric provides different results, with 92.0% of augmented seeds by SD being valid for COCO and 8.11% for CIFAR-10. Those results support findings in other studies like [40]. In fact, FID and SSIM use different indicators to evaluate the quality of images, which leads to incompatibility in their assessment results. Another observation is that SSIM similarity scores are unsurprisingly low, e.g., only 28.88% of CIFAR augmented seeds are valid. SSIM quantifies the similarity between input and augmented seed based on three key features (luminance, contrast, and structure). With SD, the augmentation happens by introducing disturbing changes into the input image latent space, i.e., larger changes in these features, which consequently results in a high level of dissimilarity and low SSIM values.

Thus, a deeper analysis using CLIP score is performed and enables us to compare the augmented seed to the text prompt used to generate it. The results are reported in Table III and assess the augmented seed fidelity using the CLIP metric with threshold $\delta = 0.8$ after normalisation. The CLIP results are in line with FID and demonstrate the effectiveness of SD inpainting in generating photo-realistic images, especially for Leaves and COCO datasets with up to 97.78% of augmented seeds being evaluated as high fidelity for the Leaves dataset and 58% for COCO. The fidelity rate of CIFAR-10 is generally lower than other datasets with only 20.56%. Intuitively, the reason behind the low quality of CIFAR-10 augmented seeds is due to the resolution of CIFAR-10 which is relatively low. **Answer to RQ1: Results show that with proper constraint design and parameter tuning, GENFUZZER with Stable Diffusion is effective in generating high-fidelity synthetic test inputs**.

RQ2 (**Effectiveness**): *How effective is Stable Diffusion for Coverage Guided-Fuzzing compared to metamorphic mutation?* i.e., can the generated test cases improve a given set in terms of testing capability? To answer **RQ2**, the experiments are designed to evaluate the effect of semantic data augmentation output, i.e., valid augmented seeds (cf. Fig. 1), on improving coverage in DNN testing under different criteria. We intensively evaluated two fuzzing strategies: (1)“*Random-Inpainting*”: adopts the uniform sampling seed prioritization strategy with Stable Diffusion Inpainting for data augmentation. (2)“*Random-Noise*”: this is used as a baseline strategy. It adopts uniform sampling with Gaussian Noise as a data augmentation technique. To reduce the influence of randomness, each fuzzer execution has been repeated 10 times and the results have been averaged and illustrated in Table IV. The row Init. represents the coverage achieved by the initial test set T^t. Compared to the initial test set (row Init.), we notice that test cases generated by Random-Inpainting improve the coverage scores across all the criteria when added to the test set by up to 26%. For instance, there was no difficulty in enhancing the IDC, KMNC, and LSA criteria, when the test set was augmented with the new test cases, as they went respectively from 40.29%, 2.61%, 22.21% to 43.77% 16.71% 38.94% for the COCO dataset. On the other hand, with the Gaussian Noise, there was no significant increase, and in some cases, we noticed even a drop in the initial coverage scores. In most scenarios, Random-Inpainting outperforms Random-Noise in terms of coverage scores when comparing the results of CGF strategies. **Answer to RQ2: GENFUZZER with semantic augmentation is more effective to maximize coverage than random (*$Init.$* row in Table IV), and traditional CGF metamorphic technique (i.e., Gaussian Noise), especially for those criteria that are difficult to cover, i.e., IDC, LSA.**

Table IV: Average results in (%) of coverage criteria over 10 runs by fuzzer with different data augmentation strategies

Model	Strategies	**IDC$_4$**	KMNC	NBC	LSA	NC
Vgg19 + COCO	Init	40.29%	2.61%	13.66%	22.21%	5.93%
	Random-Inpainting	**43.77**%	**16.71**%	**31.79**%	**38.94**%	**13.21**%
	Random-Noise	29.01%	14.60%	13.26%	18.12%	4.86%
LeNet5 + CIFAR-10	Init.	21.06%	13.78%	**1.61**%	1.31%	54.90%
	Random-Inpainting	**45.13**%	**21.24**%	0.73%	**1.46**%	**67.20** %
	Random-Noise	5.13%	14.27%	0.0 %	0.0 %	46.99 %
All-ConvNet + Leaves	Init.	42.86%	19.62%	11.67	49.59%	25.44%
	Random-Inpainting	**68.75**%	**35.23**%	**42.55**%	**69.17**%	**32.90**%
	Random-Noise	37.50%	19.89%	21.69%	35.75%	8.29%

V. CONCLUSIONS AND FUTURE WORK

We introduced GENFUZZER, a novel CGF method that uses semantic data augmentation to optimise the test case generation for DL testing. Our approach can significantly improve its coverage exploration ability and performs well in generating semantically-diverse test suites. Unlike existing work, GENFUZZER advances quality assurance for DL by leveraging generative AI models like Stable Diffusion. In the future, we plan to design more comprehensive semantic mutation techniques using different generative AI models and use them to guide the fuzzing, thereby improving the ability to detect failures and improve the overall DNN testing process. We also plan to conduct extensive experiments to evaluate the robustness of our approach. This will involve a broader range of data and a comparison with other advanced augmentation methods like AugMix [41] and mixup [42]. Furthermore, we will gather additional data on GENFUZZER’s efficacy in detecting defects introduced by DNNs during deployment.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[2] C.-W. Zhang, M.-Y. Yang, H.-J. Zeng, and J.-P. Wen, "Pedestrian detection based on improved lenet-5 convolutional neural network," *Journal of Algorithms & Computational Technology*, vol. 13, p. 1748302619873601, 2019.

[3] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *International Conference on Neural Information Processing Systems*, 2014, pp. 3104–3112.

[4] G. Litjens, T. Kooi, B. E. Bejnordi *et al.*, "A survey on deep learning in medical image analysis," *Medical Image Analysis*, vol. 42, pp. 60–88, 2017.

[5] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner *et al.*, "End to end learning for self-driving cars," 2016.

[6] K. D. Julian, J. Lopez, J. S. Brush, M. P. Owen, and M. J. Kochenderfer, "Policy compression for aircraft collision avoidance systems," in *IEEE Digital Avionics Systems Conference (DASC)*, 2016, pp. 1–10.

[7] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[8] X. Gao, R. K. Saha, M. R. Prasad, and A. Roychoudhury, "Fuzz testing based data augmentation to improve robustness of deep neural networks," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 2020, pp. 1147–1158.

[9] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[10] T. Tran, T. Pham, G. Carneiro, L. Palmer, and I. Reid, "A Bayesian data augmentation approach for learning deep models," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[11] J. M. Zhang, M. Harman, L. Ma, and Y. Liu, "Machine learning testing: Survey, landscapes and horizons," *IEEE Transactions on Software Engineering*, vol. 48, no. 1, pp. 1–36, 2020.

[12] A. Odena, C. Olsson, D. Andersen, and I. Goodfellow, "Tensorfuzz: Debugging neural networks with coverage-guided fuzzing," in *International Conference on Machine Learning*. PMLR, 2019, pp. 4901–4911.

[13] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 303–314.

[14] T. Y. Chen, F.-C. Kuo, H. Liu, P.-L. Poon, D. Towey, T. Tse, and Z. Q. Zhou, "Metamorphic testing: A review of challenges and opportunities," *ACM Computing Surveys (CSUR)*, vol. 51, no. 1, pp. 1–27, 2018.

[15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[16] G. Harshvardhan, M. K. Gourisaria, M. Pandey, and S. S. Rautaray, "A comprehensive survey and analysis of generative models in machine learning," *Computer Science Review*, vol. 38, p. 100285, 2020.

[17] M. Pezze and M. Young, *Software testing and analysis: process, principles, and techniques*. John Wiley & Sons, 2008.

[18] K. Pei, Y. Cao, J. Yang, and S. Jana, "Deepxplore: Automated whitebox testing of deep learning systems," in *Proceedings of the 26th Symposium on Operating Systems Principles*, 2017, pp. 1–18.

[19] L. Ma, F. Juefei-Xu, F. Zhang, J. Sun, M. Xue, B. Li, C. Chen, T. Su, L. Li, Y. Liu *et al.*, "DeepGauge: Multi-granularity testing criteria for deep learning systems," in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018, pp. 120–131.

[20] Y. Feng, Q. Shi, X. Gao, J. Wan, C. Fang, and Z. Chen, "Deepgini: prioritizing massive tests to enhance the robustness of deep neural networks," in *Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2020, pp. 177–188.

[21] S. Gerasimou, H. F. Eniser, A. Sen, and A. Cakan, "Importance-driven deep learning system testing," in *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*. IEEE, 2020, pp. 702–713.

[22] H. F. Eniser, S. Gerasimou, and A. Sen, "DeepFault: Fault localization for deep neural networks," in *International Conference on Fundamental Approaches to Software Engineering*. Springer, 2019, pp. 171–191.

[23] O. Chang, J. Metzman, M. Moroz, M. Barbella, and A. Arya, "Oss-fuzz: Continuous fuzzing for open source software," *URL: https://github.com/google/ossfuzz*, 2016.

[24] X. Xie, L. Ma, F. Juefei-Xu, M. Xue, H. Chen, Y. Liu, J. Zhao, B. Li, J. Yin, and S. See, "DeepHunter: a coverage-guided fuzz testing framework for deep neural networks," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2019, pp. 146–157.

[25] Z. Q. Zhou and L. Sun, "Metamorphic testing of driverless cars," *Communications of the ACM*, vol. 62, no. 3, pp. 61–67, 2019.

[26] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 6840–6851, 2020.

[27] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, "Glide: Towards photorealistic image generation and editing with text-guided diffusion models," *arXiv preprint arXiv:2112.10741*, 2021.

[28] Y. Balaji, S. Nah, X. Huang, A. Vahdat, J. Song, K. Kreis, M. Aittala, T. Aila, S. Laine, B. Catanzaro *et al.*, "ediffi: Text-to-image diffusion models with an ensemble of expert denoisers," *arXiv preprint arXiv:2211.01324*, 2022.

[29] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, and M. Irani, "Imagic: Text-based real image editing with diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6007–6017.

[30] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2961–2969.

[31] B. Srinivasa-Desikan, *Natural Language Processing and Computational Linguistics: A practical guide to text analysis with Python, Gensim, spaCy, and Keras*. Packt Publishing Ltd, 2018.

[32] A. Obukhov and M. Krasnyanskiy, "Quality assessment method for GAN based on modified metrics inception score and Fréchet inception distance," in *Software Engineering Perspectives in Intelligent Systems: Proceedings of 4th Computational Methods in Systems and Software 2020, Vol. 1 4*. Springer, 2020, pp. 102–114.

[33] D. Brunet, E. R. Vrscay, and Z. Wang, "On the mathematical properties of the structural similarity index," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1488–1499, 2011.

[34] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8748–8763.

[35] J. Kim, R. Feldt, and S. Yoo, "Guiding deep learning system testing using surprise adequacy," in *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 2019, pp. 1039–1049.

[36] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *13th European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[37] V. Rajinikanth, A. N. Joseph Raj, K. P. Thanaraj, and G. R. Naik, "A customized VGG19 network with concatenation of deep and handcrafted features for brain tumor detection," *Applied Sciences*, vol. 10, no. 10, p. 3429, 2020.

[38] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on Cifar-10," *Unpublished manuscript*, vol. 40, no. 7, pp. 1–9, 2010.

[39] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[40] A. Borji, "Pros and cons of GAN evaluation measures: New developments," *Computer Vision and Image Understanding*, vol. 215, p. 103329, 2022.

[41] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "AugMix: A simple data processing method to improve robustness and uncertainty," Dec. 2019.

[42] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.