Testing Vision Neural Networks for Smart Cities

Luciano Baresi, Davide Xian Yi Hu, Giovanni Quattrocchi
Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria
{name.surname}@polimi.it

***Abstract*—Testing vision neural networks is crucial for ensuring the reliability and safety of AI-driven applications in smart cities. We propose *TestDiffusion*, a novel approach combining zero-shot learning and domain adversarial testing to efficiently generate diverse, high-quality test cases from existing ones. By leveraging a pre-trained diffusion model and reinforcement learning, *TestDiffusion* enhances the robustness of vision NNs, enabling better generalization to unseen urban conditions. An initial evaluation shows that *TestDiffusion* significantly outperforms state-of-the-art methods, improving the reliability of AI-based systems in the complex environments of smart cities.**

I. Introduction

The rapid advancement of information technology has paved the way for the development of smart cities, where the integration of digital infrastructure enhances the efficiency and quality of urban services. Central to this transformation is the use of Artificial Intelligence (AI) and Neural Networks (NNs) in various applications such as smart energy management, air quality control, and public safety [1].

Vision NNs are particularly important in smart cities due to their ability to process and interpret vast amounts of visual data from urban environments. In the context of smart mobility, for instance, vision NNs are crucial for tasks such as traffic management, smart lighting, and parking space detection. The reliability and safety of these models are key, as any failure can lead to significant consequences, including accidents and disruptions in urban mobility.

One of the major challenges in deploying these NNs in real-world scenarios is their ability to generalize from the training data to new, unseen inputs [2]. This generalization is essential for NNs to function correctly in the diverse and dynamic environments found in smart cities. Traditional software testing methods fall short when it comes to evaluating the performance of NNs due to the complexity and opacity of the models. NNs learn from data and approximate complex functions that are difficult to interpret, making it challenging to determine the correct output for a given input [3]. Moreover, obtaining test cases from different and novel domains is usually complex and expensive, as it often requires extensive manual data collection and labeling.

To address these challenges, this paper introduces *TestDiffusion*, a novel approach that combines zero-shot learning and domain adversarial testing for the metamorphic testing of vision NNs. *TestDiffusion* utilizes a pre-trained general-purpose diffusion model to generate test cases using textual prompts. Our approach is designed to efficiently transform available test cases into novel and diverse ones, leveraging reinforcement learning to optimize the search for the most suitable transformations. By providing an effective and cost-efficient solution for generating test cases, *TestDiffusion* aims to help the development of safer and more dependable AI systems in urban environments.

II. Solution Overview

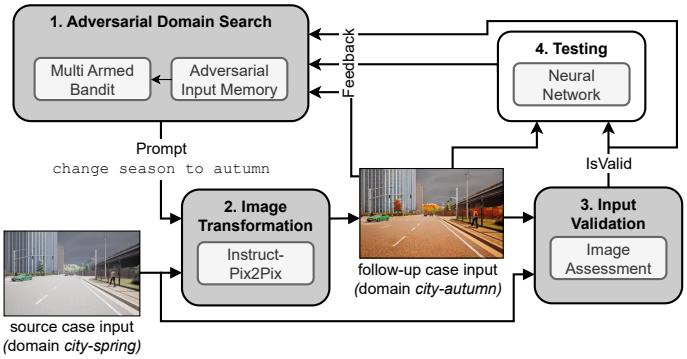

Fig. 1. *TestDiffusion*

TestDiffusion expects two user inputs: an existing set of test cases (i.e., input images and the expected outputs), referred to as source cases, and a list of textual prompts that describe the desired domain transformations. Such textual prompts should include natural language commands (e.g., `change season to autumn`, `make it rainy`, `make it darker`) to modify the input images. We assume these prompts to be defined by domain experts who can encode significant scenarios not represented in the available test cases, tailored to the specific task at hand.

Figure 1 shows an high-level overview of *TestDiffusion*. The first step of the process is the adversarial domain search based on reinforcement learning. At each iteration (i.e., for each test case generation), a Multi-Armed Bandit formulation is employed to identify the transformation (i.e., the prompt) that, when applied to a randomly selected existing test case, is most likely to cause the NN to mispredict. The algorithm is designed to balance the exploration of new prompts with the exploitation of known high-reward prompts. The reward function is calculated using three metrics: i) the *effectiveness* of the generated test case (or *follow-up case*) in producing erroneous outputs, ii) the *diversity* of the generated test case compared to the ones generated in previous iterations, and iii) the *quality* of the generated test case computed using Image Quality Assessment [4]. To calculate the diversity among generated test cases, our approach maintains a memory of

adversarial inputs, which aids in refining the selection process based on feedback from previous iterations.

Once a prompt is selected, the process moves to the image transformation stage, where *InstructPix2Pix* [5], a pre-trained diffusion model, takes a randomly selected source case input (e.g., an image in a city during spring) and the selected prompt (e.g., `change season to autumn`) to generate a follow-up case input. This transformation is designed to create realistic images that reflect the specified changes while maintaining the essential features of the original image (i.e., without changing the expected output).

Then, the newly generated follow-up case then undergoes input validation. This stage ensures that the transformed image meets predefined quality criteria discarding unrealistic or low-quality transformations. After validation, the remaining follow-up cases are used to test the NN. This testing phase evaluates the NN's performance on the new inputs, identifying any potential faults or weaknesses in its ability to generalize across different scenarios. The results of this testing provide feedback that is used to refine the adversarial domain search, improving the selection of prompts in the next iterations.

This process is repeated multiple times, until the desired amount of test cases is generated.

III. Preliminary results and Conclusions

We conducted an initial evaluation to demonstrate the effectiveness of *TestDiffusion* compared against two state-of-the-art approaches: *FGSM (Fast Gradient Sign Method)* [6] and CycleGAN [7]. The former is a gradient-based adversarial attack approach that perturbs input images by adding small, intentional noise to maximize the NN's prediction error. The latter generates new test cases using Generative Adversarial Networks (GANs) that are tailored to a single specific transformation (i.e., adding rain to an image).

From a qualitative perspective, gradient-based approaches like FGSM allow testing the robustness of NNs by introducing small or even imperceptible perturbations that maximize prediction errors. However, they do not test NNs in different, varied real-world scenarios as *TestDiffusion* does. GAN-based approaches, such as CycleGAN, require training separate models for each transformation, making it cumbersome to support multiple transformations. In contrast, *TestDiffusion* exploits a pre-trained model that allows for any transformation to be applied via text prompts, providing a more versatile and efficient method for testing NNs across diverse scenarios.

Experiments setup. To compared the solutions also from a quantitative point of view, we run a set of experiments on a machine with a 32-core *AMD R9 5950X* processor, 64 GB RAM, and an *Nvidia 4090* GPU. The evaluation used a dataset on autonomous driving, being one of the most prominent use cases of vision NNs for smart cities, extracted from an extended Udacity simulator [8], containing $281,930$ images with steering angles. We tested the approaches on different NN architectures, namely, *DAVE-2* [9] and three *Resnet* variants: *Resnet-18*, *Resnet-34*, and *Resnet-50*. We used 178 prompts for *TestDiffusion* to modify the source images, altering characteristics like time of day, weather, road conditions, and background.

TABLE I
Comparison.

Approach	Detected Faults (%)	S-Faults (%)	L-Faults (%)
TestDiffusion	79.87%	24.01%	53.88%
FGSM	66.20%	41.34%	24.86%
CycleGAN	68.95%	33.21%	35.74%

Results. Table I presents the performance comparison of *TestDiffusion* against FGSM and CycleGAN, averaged across the different NN models. We employed three main metrics: i) *Detected Faults*, the % of generated test cases where the NN produced incorrect outputs, ii) *S-Faults*, the % of test cases with minor prediction errors, and iii) *L-Faults*, the % of test cases with substantial prediction errors that could cause significant real-world failures.

Results show that *TestDiffusion* outperforms both FGSM and CycleGAN. On average, *TestDiffusion* detected faults in 79.87% of the generated test cases, while FGSM and CycleGAN detected faults in 66.20% and 68.95% of the cases, respectively. This indicates that *TestDiffusion* is more effective at uncovering potential issues in NNs. *TestDiffusion* generates test cases that lead to higher L-Faults (53.88%) and lower S-Faults (24.01%). FGSM, probably due to the small applied perturbations, detected more small faults (41.34%) than large ones (24.86%). CycleGAN produced more balanced results, with 33.21% S-Faults and 35.74% L-Faults.

Conclusions. Testing NNs is crucial in smart cities to ensure the reliability and safety of AI-driven applications. Overall, these initial results demonstrate that *TestDiffusion* provides a comprehensive and effective testing approach, enhancing the robustness and reliability of vision NNs across diverse and challenging scenarios.

References

[1] Z. Allam and Z. A. Dhunny, "On Big Data, Artificial Intelligence and Smart Cities," *Cities*, vol. 89, pp. 80–91, 2019.

[2] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, " Exploring Generalization in Deep Learning," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 5947–5956.

[3] D.-X. Zhou, "Universality of Deep Convolutional NNs," *Applied and computational harmonic analysis*, vol. 48, pp. 787–794, 2020.

[4] S. Kastryulin, J. Zakirov, D. Prokopenko, and D. V. Dylov, "PyTorch Image Quality: Metrics for Image Quality Assessment," *CoRR*, vol. abs/2208.14818, 2022.

[5] T. Brooks, A. Holynski, and A. A. Efros, "InstructPix2Pix: Learning to Follow Image Editing Instructions," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18 392–18 402.

[6] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," in *Proc. of the Int. Conference on Learning Representations*, 2015.

[7] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," in *Proc. of the Int. Conference on Computer Vision*, 2017, pp. 2242–2251.

[8] A. Stocco, M. Weiss, M. Calzana, and P. Tonella, "Misbehaviour prediction for autonomous driving systems," in *Proc. of the Int. Conference on Software Engineering*, 2020, pp. 359–371.

[9] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to End Learning for Self-Driving Cars," *CoRR*, vol. abs/1604.07316, 2016.