Work In Progress: Safety and Robustness Verification of Autoencoder-Based Regression Models using the NNV Tool

Neelanjana Pal
Department of Electrical and Computer Engineering
Vanderbilt University, USA
neelanjana.pal@vanderbilt.edu

Taylor T Johnson
Department of Electrical and Computer Engineering
Vanderbilt University, USA
taylor.johnson@vanderbilt.edu

This work in progress paper introduces robustness verification for autoencoder-based regression neural network (NN) models, following state-of-the-art approaches for robustness verification of image classification NNs. Despite the ongoing progress in developing verification methods for safety and robustness in various deep neural networks (DNNs), robustness checking of autoencoder models has not yet been considered. We explore this open space of research and check ways to bridge the gap between existing DNN verification methods by extending existing robustness analysis methods for such autoencoder networks. While classification models using autoencoders work more or less similar to image classification NNs, the functionality of regression models is distinctly different. We introduce two definitions of robustness evaluation metrics for autoencoder-based regression models, specifically the percentage robustness and un-robustness grade. We also modified the existing Imagestar approach, adjusting the variables to take care of the specific input types for regression networks. The approach is implemented as an extension of NNV, then applied and evaluated on a dataset, with a case study experiment shown using the same dataset. As per the authors' understanding, this work in progress paper is the first to show possible reachability analysis of autoencoder-based NNs.

1 Introduction

State-of-the-art and well-trained neural networks (NN) can easily be attacked by small perturbations in inputs, leading to significant aberrations in their outputs [14, 23, 33]. These input perturbations are not only limited to image-based networks but also apply to other input types as well, e.g., time-series data or input signals. Such lack of robustness poses serious risks to information integrity, privacy and security, and can be catastrophic in safety-critical applications [11, 29]. While verification of NNs with image inputs is a vastly growing research area; specifically, with recent ongoing works on safety and robustness checking of feedforward (FFNN), convolutional (CNN), and semantic segmentation networks (SSN); less has been done in the domain of autoencoder verification. Classification models using autoencoders work almost similar to usual classifiers, but there is a need for new research to develop verification techniques for regression models. The regression-based autoencoders regenerate the input in its output and thus can be checked using verification techniques whether the recreated output comes within a certain accepted range of the unperturbed input, in case there is a certain fault/attack on its input side.

In a prior work, the authors of [36] introduced a novel framework for NN verification named **Neural Network Verification** (NNV) [38] tool, capable of evaluating the robustness of several DNN architectures, e.g., FFNN, CNN, SSN, etc. Later, a new set-based approach, **Imagestar** [34, 36] is also incorporated into this tool. In this work in progress work, we explore similar methods in the context of autoencoder verification via experimenting on a sampled dataset and checking if the output lies within a pre-determined safe threshold around the corresponding uninterrupted input values, given a specific type of fault in the input.

Remke & Tran (Eds.): Symbolic-Numeric Methods
for Reasoning about CPS and IoT (SNR'21)
EPTCS 361, 2022, pp. 79–88, doi:10.4204/EPTCS.361.8

2 Related Work

Safety verification and robustness checking has attracted enormous attention in many application areas such as machine learning [1, 2, 20, 21, 30, 42, 43, 48], formal methods [8, 15, 16, 26, 35, 44–46], and security [13, 41, 42] etc. Many researchers are focused on developing new formal method based tools to address this vastly evolving field, some examples being satisfiability modulo theories (SMT) [16], polyhedron [35, 44], mixed-integer linear program (MILP) [8], interval arithmetic [41, 42], zonotope [30], input partition [46], linearization [43], abstract-domain [31] and star set based [36] methods. So far, several safety verification methodologies have been proposed for different neural network (NN) architectures, the focus mostly being on Feed-Forward Networks [7, 10, 36, 46], Convolutional Neural networks [3, 9, 17, 20, 31, 34], Semantic segmentation networks [4, 12, 19, 22, 25, 37, 50] and some on Recurrent Neural Networks [2, 18].

Autoencoder NNs are present in the literature for quite some time and their typical application lies in data-denoising [39, 40], high quality non-linear feature detection [5], anomaly detection [6, 28, 49], fault classification [24, 32], imbalanced data classification [47], etc.; but as per the authors' knowledge there is no prior verification work in the domain of autoencoder-based NNs.

3 Background

3.1 Autoencoder

Autoencoders are a subset of Artificial Neural Networks (ANN) that try to reconstruct an input at the output. In general, an autoencoder model is composed of two main components: an encoder and a decoder. The encoder part compresses the input into a latent space representation, and the decoder tries to recreate the input from the encoder output. In other terms, the encoder reduces the feature dimensions of the input (similar to Principal Component Analysis [27]) and can thus be used for the data preparation steps for other machine learning models. Based on the learning objective, these applications can be broadly classified into two main categories:

1. Regression Task: Here, the autoencoder is used to recreate the input at the output.
2. Classification Task: In this application, a complete autoencoder model is first trained as a regression model. Then the decoder part is removed from the model; a softmax and a classification layer are added after the bottleneck, and the input labels are 'one hot-coded' and passed along with the input. This slightly modified model is then trained again to generate an autoencoder-based classification model.

As mentioned in Sec.1, classification models using autoencoders works almost similar to a regular classifier, hence the classification task is not included in the scope of this paper.

3.2 Neural Network Verification Tool [38]

The Neural Network Verification (NNV) tool is a set-based framework for NN verification [38]. It supports several reachability algorithms for safety verification and robustness analysis of different types of DNNs. In general, for reachability analysis of a NN, the output reachable sets are obtained in a layer-by-layer manner, from a given input, specified by the upper and lower bounds of perturbation around the actual one. The reachable sets at the final layer are the collection of all possible states of the DNN.

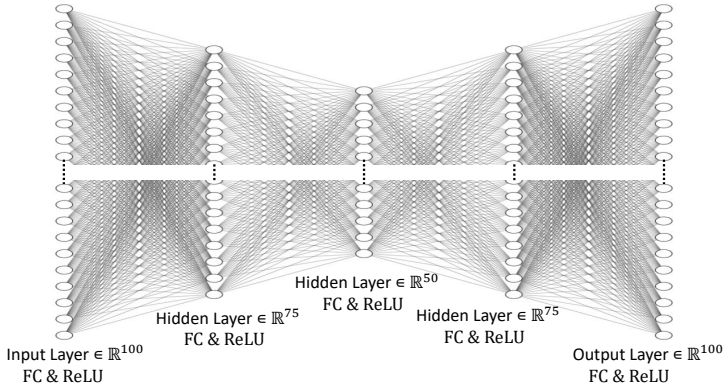


Figure 1: Autoencoder Model used in the paper [*FC: Fully connected]

The DNN is considered 'safe' iff there is no intersection between the output sets and the unsafe region, defined by the safety properties [38].

Reachability using NNV: For NN verification with image inputs, a new approach [34] was introduced as a part of NNV, called Imagestar. Imagestar is the first set-based approach to efficiently deal with deep CNNs, e.g., VGG16 and VGG19, which combines the operations on images with linear program solving.

The input for the regression model considered in this paper is a set of time-instanced signals. Imagestars [34] are not directly applicable here and require extension. For reachability analysis using NNV, an approach with a similar underlying concept of Imagestar is used, but the dimension of input will change to a 1D equivalent vector, and as the specific input used here will be a signal, we name this approach 'Signalstar'.

Similar to Imagestar, Signalstar is also a tuple of three variables $\langle c, V, P \rangle$, and the set of signals represented by the Signalstar is given as:

$$[\![\Theta]\!] = \{x \mid x = c + \sum_{i=1}^{m} (\alpha_i v_i) \mid P(\alpha_1, \alpha_2, ..., \alpha_m) = \top\} \tag{1}$$

where $c \in \mathbb{R}^n$ is the anchor signal or central signal with n time instances. Similar to Imagestar $V = \{v_1, v_2, ...v_m\}$ is generator signal instances, which is a set of m signals in $\mathbb{R}^n$. The generator signals are arranged to form the basis array of Signalstar $(n \times m)$. $P : \mathbb{R}^m \rightarrow \{\top, \bot\}$ is a predicate.

$$\Theta = \mathbf{c} + \alpha \mathrm{v} = \begin{bmatrix} 0 \\ 2 \\ 1 \\ 2 \end{bmatrix} + \alpha \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \mathrm{P} \equiv \begin{pmatrix} 1 \\ -1 \end{pmatrix} \alpha \leq \begin{pmatrix} 2 \\ 2 \end{pmatrix}$$
$$\boldsymbol{c} \in \mathbb{R}^{4\,X\,1\,X\,1} \quad \boldsymbol{v} \in \mathbb{R}^{4\,X\,1\,X\,1}$$

Figure 2: Signalstar

Another way of defining the Imagestar or Signalstar set is to use the upper and lower bounds of the attack centering the actual input. These bounds on each input parameter along with the predicates create the complete set of constraints the optimizer will solve to generate the initial set of states. An example of Signalstar is shown in Fig. 2.

3.3 Adversarial Attack on Signal or Signal Noise

An adversarial attack can be defined as some unwanted modification in the input, which causes the deviation in output w.r.t the actual result. Here, some signal noises added by the sensors can be thought of as equivalent to the adversarial attack on the input. For this paper, we will consider a simpler noise, called impulse noise, a random occurrence of spikes of very short duration and relatively high amplitude. These noises are alternatively termed as 'Spike faults'. While coming from a sensor, a common source for this noise is voltage spikes in the device. Because of its sporadic nature, it's really difficult to detect and analyze.

A sample fault signal (in red) overlapping the actual signal (in blue) is shown below in Fig. 3

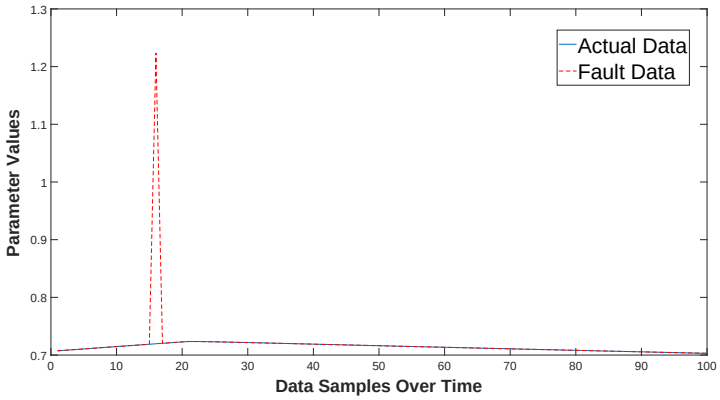


Figure 3: Sample Fault Signal and the actual signal

4 Problem Formulation: Verification of Autoencoder Models

The problem statement is devised with the help of and collaboration with TU Munich[1], where, autoencoder verification is an important part of a closed-loop system. It can be described as: first, a sensor collecting a signal with a fixed time range; then, passing the sampled sensor data to the autoencoder to get the best possible version of the time signal. Once the signal is reconstructed, it is sent to a controller for further applications.

Here, the task that we are concerned with, is to analyze the reconstructed output of the autoencoder using reachability methods and by defining some measures to protect the controller from faulty inputs. In other words, the evaluation process includes checking how big a fault can be tolerated by the controller, given an uninterrupted signal, the fault location, and an acceptable pair of upper-lower limits. In real life, this range depends on the devices the output signal will be passed to, and their allowable capacity of accepting min and max current/voltage values. The necessary components of the complete loop are as shown in Fig. 4.

While the data is passed from the sensor to the autoencoder, sensor noises may get added to the original data. As the controller will be designed to work with a specific range of input values, external errors can cause the controller to malfunction and thus can cause the entire closed-loop system to collapse. So

[1]Hongpeng Cao, Mirco Theile, Bingzhuo Zhong, Dr.Marco Caccamo of **Chair of Cyber-Physical Systems in Production Engineering, Technical University of Munich (TUM)**

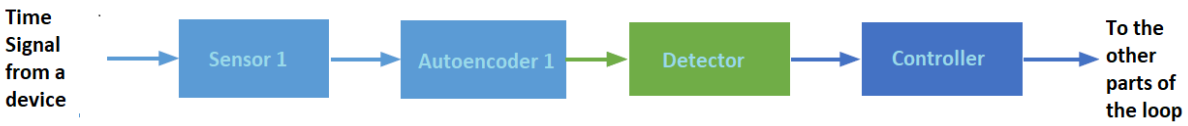


Figure 4: System Model

it is necessary to detect (in the Detector part of Fig. 4) if the reformed signal is within the acceptable range of the controller or not.

5 Case Study

5.1 Experimental Steps

(1) **Dataset Selection and Autoencoder Model Training:** The dataset is provided by our collaborators in TU Munich [2]. Each signal segment consists of 100 time samples and is then normalized using Min-Max technique. The entire dataset is divided into, 2057 training samples and 363 test samples. For generating the fault signals, the same test samples are used with spike faults at random time instances. For the primary phase of work, only one fault per signal is considered as of now.

Autoencoder architecture used for this paper is a five-layer NN, shown in Fig. 1. The first two layers comprise the encoder part of the architecture and the last two, the decoder part. The third layer creates the latent space for the model. Here, the 'Adam' optimizer and 'Mean square error' loss function is used for training purposes.

(2) **Attacks/Perturbations on the Inputs for Verification:** Verification is always done on the inputs with certain noise or perturbations to check if the model can produce the same outputs correctly, i.e., regenerate the unperturbed inputs. As mentioned in Sec.3, spike faults are added for this regression task.

(3) **Reachability Analysis Using NNV:** Given a pair of upper-lower limits of the attack around the central signal, the input set created using Signalstar will propagate through the different layers and sometimes split into multiple sets (because of the presence of the non-linear activation functions/layers). Finally, at output layer produces the collection of output sets. The final output sets thus created are called the reachable sets of the model w.r.t the input attack. Once the output reachable sets are calculated using NNV, it is checked if they intersect with the unsafe regions (specified in terms of the faulty signal).

(4) **Evaluation Measures:** In the case of signal inputs, even after the presence of a fault, if the output set can recreate a similar input signal or if the bounds of the output signal are within a certain permissible limit of the input signal, the model is said to be robust. As autoencoder verification is a comparatively new direction of work, the measurement of success is relative. As analysis of safety and robustness verification has already been done on Classification based NNs, we can use a similar measure as well. Though for the regression model the evaluation measure is still an open question, we tried to devise two evaluation metrics for this paper:

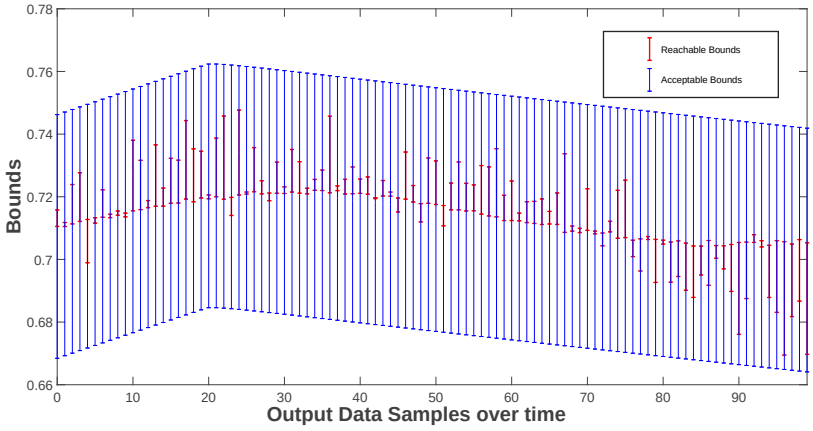
(a) **Percentage Robustness:** For calculating the robustness and safety measure for such application, percentage robustness measure can be used. It can be defined as the number of instances where the output bound is within the threshold values (i.e., the acceptable upper and lower bounds), divided by the total

[2]Hongpeng Cao, Mirco Theile, Bingzhuo Zhong, Dr. Marco Caccamo of **Chair of Cyber-Physical Systems in Production Engineering, Technical University of Munich (TUM)**

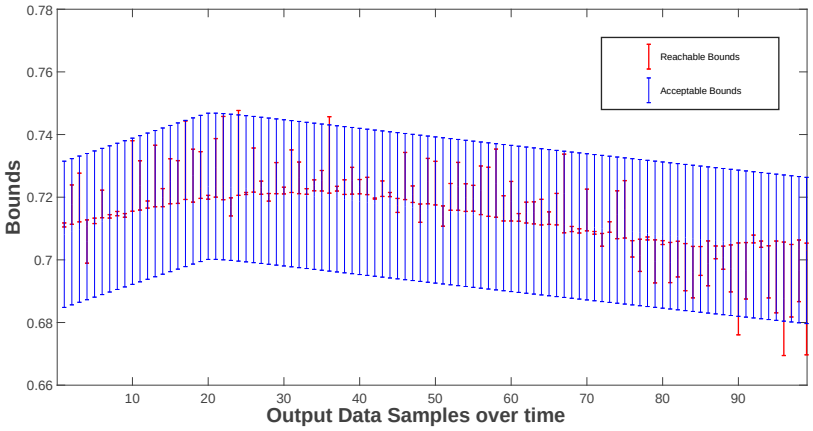
number of time instances. It gives a measure of the relative safety of the network w.r.t the input signal and for a given fault a value 100 being perfectly safe and 0 being completely unsafe.

(b) **Un-robustness Grade:** This failure grade can be defined as the maximum difference between the actual signal and corresponding output bound (upper or lower) divided by the threshold value (acceptable deviation). The higher it is from value 1, the more it's deviated from the permissible range, i.e., the grade of Un-robustness is more.

5.2 Experiment Results



(a) Threshold = 0.0389



(b) Threshold = 0.0233

Figure 5: Output bounds (red) of the fault signal and bounds (blue) for the input signal with 2 different thresholds.

Figs. 5a and 5b depict the plots of output bounds corresponding to the sample signals shown in Fig. 3, both for the actual and its faulty counterpart; faulty signal bounds at the output being shown in red and the actual signal with permissible limits in blue for two separate acceptable ranges (for example +/-0.0389 and +/-0.0233). We can see from these two plots that whether the reconstructed output signal will be accepted for the controller or not, also depends on this range; while in Fig. 5a, the same output bounds come well within the range and hence considered robust; bounds for some time instances in Fig. 5b go

beyond the accepted range and hence is not safe to use for further process. For the example, Fig. 5b previously defined evaluation matrices can be calculated as:
Percentage Robustness: total time instance = 100; instances, where output bound is within the permissible limit = 95; therefore percentage Robustness = .95.
Un-robustness Grade: deviation is maximum at time instance 96; here lower bound is deviated maximum from the actual signal and the corresponding value is 0.6695; lower limit of permissible range at t = 96 is 0.7057;hence maximum deviation = 0.7057-0.6695 = 0.0363; finally the Grade of Un-robustness = 0.0363/0.0233 = 1.5536.

6 Conclusion

In this work in progress paper, we present the first formal verification approach for autoencoders, based on reachability using Imagestars. In our case study, the regression-based autoencoder model takes time-instanced signals from a device, which might have spike faults, and we analyzed the reconstructed output w.r.t an uninterrupted input. Given an accepted value of upper and lower limit (at output), we then checked if the reconstructed signal is within that range of the actual signal. We evaluated the results with two different threshold values.

Though this is the first stage of analysis of autoencoder based models, in future work, we will continue to develop better evaluation metrics that are more relevant to the use case where reachability-based analysis may be most effective. We can also modify other verification tools used for classification models and investigate comparative effectiveness with them. Incorporating other autoencoder applications as case studies is also planned for our future work. Another avenue can be to study, how the denoising property of an autoencoder help mitigate the effect of input faults in the output, and that in turn increases robustness. Robustness checking of variational autoencoder (VAE) models may also be an interesting direction to explore.

Acknowledgements

The material presented in this paper is based upon work supported by the National Science Foundation (NSF) through grant numbers 1910017, 1918450, and 2028001, the Defense Advanced Research Projects Agency (DARPA) under contract number FA8750-18-C-0089, and the Air Force Office of Scientific Research (AFOSR) under contract number FA9550-22-1-0019. Any opinions, findings, and conclusions or recommendations expressed in this work in progress publication are those of the authors and do not necessarily reflect the views of AFOSR, DARPA, or NSF.

References

[1] Michael Akintunde, Alessio Lomuscio, Lalit Maganti & Edoardo Pirovano (2018): *Reachability analysis for neural agent-environment systems*. In: *Sixteenth International Conference on Principles of Knowledge Representation and Reasoning*.

[2] Michael E Akintunde, Andreea Kevorchian, Alessio Lomuscio & Edoardo Pirovano (2019): *Verification of RNN-Based Neural Agent-Environment Systems*. In: *Proceedings of the 33th AAAI Conference on Artificial Intelligence (AAAI19). Honolulu, HI, USA. AAAI Press*, doi:10.1609/aaai.v33i01.33016006.

[3] Greg Anderson, Shankara Pailoor, Isil Dillig & Swarat Chaudhuri (2019): *Optimization and abstraction: a synergistic approach for analyzing neural network robustness*. In: *Proceedings of the 40th*

ACM SIGPLAN Conference on Programming Language Design and Implementation, pp. 731–744, doi:10.1145/3314221.3314614.

[4] Anurag Arnab, Ondrej Miksik & Philip HS Torr (2018): *On the robustness of semantic segmentation models to adversarial attacks*. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 888–897, doi:10.1109/CVPR.2018.00099.

[5] M. Chen, X. Shi, Y. Zhang, D. Wu & M. Guizani (2017): *Deep Features Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network*. *IEEE Transactions on Big Data*, pp. 1–1, doi:10.1109/TBDATA.2017.2717439.

[6] Z. Chen, C. K. Yeo, B. S. Lee & C. T. Lau (2018): *Autoencoder-based network anomaly detection*. In: *2018 Wireless Telecommunications Symposium (WTS)*, pp. 1–5, doi:10.1109/WTS.2018.8363930.

[7] Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan & Ashish Tiwari (2018): *Learning and verification of feedback control systems using feedforward neural networks*. *IFAC-PapersOnLine* 51(16), pp. 151–156, doi:10.1016/j.ifacol.2018.08.026.

[8] Souradeep Dutta, Susmit Jha, Sriram Sankaranarayanan & Ashish Tiwari (2018): *Output range analysis for deep feedforward neural networks*. In: *NASA Formal Methods Symposium*, Springer, pp. 121–138, doi:10.1007/978-3-319-77935-5_9.

[9] Krishnamurthy Dj Dvijotham, Robert Stanforth, Sven Gowal, Chongli Qin, Soham De & Pushmeet Kohli (2020): *Efficient neural network verification with exactness characterization*. In: *Uncertainty in Artificial Intelligence*, PMLR, pp. 497–507.

[10] Ruediger Ehlers (2017): *Formal verification of piece-wise linear feed-forward neural networks*. In: *International Symposium on Automated Technology for Verification and Analysis*, Springer, pp. 269–286, doi:10.1007/978-3-319-68167-2_19.

[11] Hatem M Elattar, Hamdy K Elminir & Alaa Mohamed Riad (2019): *Conception and implementation of a data-driven prognostics algorithm for safety–critical systems*. *Soft Computing* 23(10), pp. 3365–3382, doi:10.1007/s00500-017-2995-7.

[12] Peter M. Full, Fabian Isensee, Paul F. Jäger & Klaus Maier-Hein (2020): *Studying Robustness of Semantic Segmentation under Domain Shift in cardiac MRI*, doi:10.1007/978-3-030-68107-4_24. arXiv:2011.07592.

[13] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri & Martin Vechev (2018): *Ai 2: Safety and robustness certification of neural networks with abstract interpretation*. In: *Security and Privacy (SP), 2018 IEEE Symposium on*, doi:10.1109/SP.2018.00058.

[14] Ian J Goodfellow, Jonathon Shlens & Christian Szegedy (2014): *Explaining and harnessing adversarial examples*. *arXiv preprint arXiv:1412.6572*, doi:10.48550/arXiv.1412.6572.

[15] Xiaowei Huang, Marta Kwiatkowska, Sen Wang & Min Wu (2017): *Safety verification of deep neural networks*. In: *International Conference on Computer Aided Verification*, Springer, pp. 3–29, doi:10.1007/978-3-319-63387-9_1.

[16] Guy Katz, Clark Barrett, David L Dill, Kyle Julian & Mykel J Kochenderfer (2017): *Reluplex: An efficient SMT solver for verifying deep neural networks*. In: *International Conference on Computer Aided Verification*, Springer, pp. 97–117, doi:10.1007/978-3-319-63387-9_5.

[17] Guy Katz, Derek A Huang, Duligur Ibeling, Kyle Julian, Christopher Lazarus, Rachel Lim, Parth Shah, Shantanu Thakoor, Haoze Wu & Aleksandar Zeljić (2019): *The marabou framework for verification and analysis of deep neural networks*. In: *International Conference on Computer Aided Verification*, Springer, pp. 443–452, doi:10.1007/978-3-030-25540-4_26.

[18] Igor Khmelnitsky, Daniel Neider, Rajarshi Roy, Benoît Barbot, Benedikt Bollig, Alain Finkel, Serge Haddad, Martin Leucker & Lina Ye (2020): *Property-Directed Verification of Recurrent Neural Networks*. *arXiv preprint arXiv:2009.10610*, doi:10.48550/arXiv.2009.10610.

[19] Marvin Klingner, Andreas Bar & Tim Fingscheidt (2020): *Improved Noise and Attack Robustness for Semantic Segmentation by Using Multi-Task Training With Self-Supervised Depth Estimation*. In: *Pro-*

ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, doi:10.1109/CVPRW50498.2020.00168.

[20] Panagiotis Kouvaros & Alessio Lomuscio (2018): *Formal verification of cnn-based perception systems*. arXiv preprint arXiv:1811.11373, doi:10.48550/arXiv.1811.11373.

[21] Alessio Lomuscio & Lalit Maganti (2017): *An approach to reachability analysis for feed-forward relu neural networks*. arXiv preprint arXiv:1706.07351, doi:10.48550/arXiv.1706.07351.

[22] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz & Demetri Terzopoulos (2020): *Image Segmentation Using Deep Learning: A Survey*, doi:10.48550/arXiv.2001.05566. arXiv:2001.05566.

[23] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi & Pascal Frossard (2016): *Deepfool: a simple and accurate method to fool deep neural networks*. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2574–2582, doi:10.1109/CVPR.2016.282.

[24] Nauman Munir, Jinhyun Park, Hak-Joon Kim, Sung-Jin Song & Sung-Sik Kang (2020): *Performance enhancement of convolutional neural network for ultrasonic flaw classification by adopting autoencoder*. NDT & E International 111, p. 102218, doi:10.1016/j.ndteint.2020.102218.

[25] Gabriel Oliveira, Claas Bollen, Wolfram Burgard & Thomas Brox (2017): *Efficient and robust deep networks for semantic segmentation*. The International Journal of Robotics Research 37, p. 027836491771054, doi:10.1177/0278364917710542.

[26] Luca Pulina & Armando Tacchella (2010): *An abstraction-refinement approach to verification of artificial neural networks*. In: International Conference on Computer Aided Verification, Springer, pp. 243–257, doi:10.1007/978-3-642-14295-6_24.

[27] Markus Ringnér (2008): *What is principal component analysis?* Nature biotechnology 26(3), pp. 303–304, doi:10.1038/nbt0308-303.

[28] Mayu Sakurada & Takehisa Yairi (2014): *Anomaly detection using autoencoders with nonlinear dimensionality reduction*. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, pp. 4–11, doi:10.1145/2689746.2689747.

[29] Divya Saxena & Vaskar Raychoudhury (2017): *Design and verification of an NDN-based safety-critical application: A case study with smart healthcare*. ieee transactions on systems, man, and cybernetics: systems 49(5), pp. 991–1005, doi:10.1109/TSMC.2017.2723843.

[30] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel & Martin Vechev (2018): *Fast and effective robustness certification*. In: Advances in Neural Information Processing Systems, pp. 10825–10836.

[31] Gagandeep Singh, Timon Gehr, Markus Püschel & Martin Vechev (2019): *An abstract domain for certifying neural networks*. Proceedings of the ACM on Programming Languages 3(POPL), p. 41, doi:10.1145/3290354.

[32] Wenjun Sun, Siyu Shao, Rui Zhao, Ruqiang Yan, Xingwu Zhang & Xuefeng Chen (2016): *A sparse auto-encoder-based deep neural network approach for induction motor faults classification*. Measurement 89, pp. 171–178, doi:10.1016/j.measurement.2016.04.007.

[33] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow & Rob Fergus (2013): *Intriguing properties of neural networks*. arXiv preprint arXiv:1312.6199, doi:10.48550/arXiv.1312.6199.

[34] Hoang-Dung Tran, Stanley Bak, Weiming Xiang & Taylor T Johnson (2020): *Verification of deep convolutional neural networks using imagestars*. In: International Conference on Computer Aided Verification, Springer, pp. 18–42, doi:10.1007/978-3-030-53288-8_2.

[35] Hoang-Dung Tran, Patrick Musau, Diego Manzanas Lopez, Xiaodong Yang, Luan Viet Nguyen, Weiming Xiang & Taylor T. Johnson (2019): *Parallelizable Reachability Analysis Algorithms for Feed-Forward Neural Networks*. In: 7th International Conference on Formal Methods in Software Engineering (FormaliSE2019), Montreal, Canada, doi:10.1109/FormaliSE.2019.00012.

[36] Hoang-Dung Tran, Patrick Musau, Diego Manzanas Lopez, Xiaodong Yang, Luan Viet Nguyen, Weiming Xiang & Taylor T. Johnson (2019): *Star-Based Reachability Analysis for Deep Neural Networks*. In: *23rd International Symposisum on Formal Methods (FM'19)*, Springer International Publishing, doi:10.1007/978-3-030-30942-8˙39.

[37] Hoang-Dung Tran, Neelanjana Pal, Patrick Musau, Xiaodong Yang, Nathaniel P Hamilton, Diego Manzanas Lopez, Stanley Bak & Taylor T Johnson (2021): *Robustness Verification of Semantic Segmentation Neural Networks using Relaxed Reachability*. In: *Proceedings of the 33rd International Conference on Computer-Aided Verification*, Springer, doi:10.1007/978-3-030-81685-8_12.

[38] Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak & Taylor T Johnson (2020): *NNV: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems*. In: *International Conference on Computer Aided Verification*, Springer, pp. 3–17, doi:10.1007/978-3-030-53288-8_1.

[39] Pascal Vincent, Hugo Larochelle, Yoshua Bengio & Pierre-Antoine Manzagol (2008): *Extracting and composing robust features with denoising autoencoders*. In: *Proceedings of the 25th international conference on Machine learning*, pp. 1096–1103, doi:10.1145/1390156.1390294.

[40] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol & Léon Bottou (2010): *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.* *Journal of machine learning research* 11(12).

[41] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang & Suman Jana (2018): *Efficient formal safety analysis of neural networks*. In: *Advances in Neural Information Processing Systems*, pp. 6369–6379, doi:10.48550/arXiv.1809.08098.

[42] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang & Suman Jana (2018): *Formal Security Analysis of Neural Networks using Symbolic Intervals*. *arXiv preprint arXiv:1804.10829*, doi:10.48550/arXiv.1804.10829.

[43] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Duane Boning, Inderjit S Dhillon & Luca Daniel (2018): *Towards Fast Computation of Certified Robustness for ReLU Networks*. *arXiv preprint arXiv:1804.09699*, doi:10.48550/arXiv.1804.09699.

[44] Weiming Xiang, Hoang-Dung Tran & Taylor T Johnson (2017): *Reachable set computation and safety verification for neural networks with ReLU activations*. *arXiv preprint arXiv:1712.08163*, doi:10.48550/arXiv.1712.08163.

[45] Weiming Xiang, Hoang-Dung Tran & Taylor T Johnson (2018): *Output reachable set estimation and verification for multilayer neural networks*. *IEEE transactions on neural networks and learning systems* 29(11), pp. 5777–5783, doi:10.1109/TNNLS.2018.2808470.

[46] Weiming Xiang, Hoang-Dung Tran & Taylor T Johnson (2019): *Specification-Guided Safety Verification for Feedforward Neural Networks*. *AAAI Spring Symposium on Verification of Neural Networks*, doi:10.48550/arXiv.1812.06161.

[47] C. Zhang, W. Gao, J. Song & J. Jiang (2016): *An imbalanced data classification algorithm of improved autoencoder neural network*. In: *2016 Eighth International Conference on Advanced Computational Intelligence (ICACI)*, pp. 95–99, doi:10.1109/ICACI.2016.7449810.

[48] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh & Luca Daniel (2018): *Efficient neural network robustness certification with general activation functions*. In: *Advances in Neural Information Processing Systems*, pp. 4944–4953, doi:10.48550/arXiv.1811.00866.

[49] Chong Zhou & Randy C Paffenroth (2017): *Anomaly detection with robust deep autoencoders*. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 665–674, doi:10.1145/3097983.3098052.

[50] W. Zhou, J. Berrio, S. Worrall & Eduardo M. Nebot (2020): *Automated Evaluation of Semantic Segmentation Robustness for Autonomous Driving*. *IEEE Transactions on Intelligent Transportation Systems* 21, pp. 1951–1963, doi:10.1109/TITS.2019.2909066.