Safety and Robustness Verification of Autoencoder based regression models using NNV tool

Nithin Kumar Guruswamy and Neelanjana Pal
Vanderbilt University, Nashville TN USA
Nithin.kumar.guruswamy@vanderbilt.edu
Neelanjana.pal@vanderbilt.edu

INRODUCTION

A. Autoencoder:Autoencoders are a subset of Artificial Neural Networks(ANN) which try to reconstruct the input at the output.In general, an autoencoder model is composed of two main components: encoder and decoder. The encoder part compresses the input into a latent space representation and the decoder tries to recreate the input from the output provided by the encoder. In other terms the encoder reduces the feature dimensions of the input (Similar as PCA) and can thus be used for the data preparation steps for other machine learning models. Typical applications of autoencoders lies in data-denoising [1]–[3], high quality non-linear feature detection[4], anomaly detection [5], [6], fault classification [7], [8],imbalanced data classification [9] etc. Based on the learning objective all these applications can be broadly classified in twomain types:

1) Regression Task: Here the autoencoder is basically used to recreate the input at the output.

2) Classification Task: In this applications the complete autoencoder model is first trained as a regression model.Then the decoder part is removed from the model and only the encoder part is used for Classification purposes. A softmax+classification layer is added after the bottleneck and the input-labels are one hot coded and passed along with the input. These slightly modified model is then trained again to generate a autoencoder based classification model.

B. Neural Network Verification Tool [10]:Neural Network Verification Tool or NNV is a set based framework for neural network verification, developed by theVeriVital lab of Vanderbilt University. It supports several reachabilty algorithms for safety verification and robustness analysis of several types of deep neural networks(NN). In general for reachability analysis of a NN, all reachable sets are calculated from an input and updated after each layers of the network and after the output layer the reachable state-space becomes an 'n-dimensional' one, 'n' being the number of different classes."

The reachable set obtained from NNV tool contains all possible states of a DNN (deep neural network)from bounded input sets" and a DNN is declared as Safe if, and only if, their reachable sets do not violate safetyproperties (i.e., have a non-

empty intersection with any state satisfying the negation of the safety property)".

C. Verification of Autoencoder models:Although verification of Neural networks (e.g. feed-forward[11]–[14], Convolutional [15], [16], etc.) became quite popularwith time and a major work is going on in several labs worldwide. But as per the authors knowledge there is no prior verification work in the domain of NN based autoencoders.

1 Method-1(Done by Nithin) : Safety and robustness verification of Autoencoder for regression using sigma method:

Autoencoder Architecture:

Autoencoder is a self supervised Neural network which is used in many applications like classification , regression (timeseries forecasting) and anomaly detection methods.In this section, we are verifying autoencoder for regression tasks.

Like already mentioned in the previous section, **verification of the autoencoder for regression task is a novel problem and has not been addressed before in the prior literature**.

Here we are trying to novel approach to verify it using combination of NNV tool and 2 sigma method.

The autoencoder architecture is shown below. It is trained using keras APIs and has 8 layers

```
29
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 29)                870
_________________________________________________________________
dense_1 (Dense)              (None, 16)                480
_________________________________________________________________
dense_2 (Dense)              (None, 8)                 136
_________________________________________________________________
dense_3 (Dense)              (None, 4)                 36
_________________________________________________________________
dense_4 (Dense)              (None, 2)                 10
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 12
_________________________________________________________________
dense_6 (Dense)              (None, 8)                 40
_________________________________________________________________
dense_7 (Dense)              (None, 16)                144
_________________________________________________________________
dense_8 (Dense)              (None, 29)                493
=================================================================
Total params: 2,221
Trainable params: 2,221
Non-trainable params: 0
```

Fig 1 shows an autoencoder architecture used for verification

Autoencoder Verification Problem formulation:

We are trying to formally verify if a given autoencoder (Feedforward Neural network) meets some notion of specification defined over inputs and outputs.

Here in this project, we consider input mean +/- 2 Sigma threshold as the input specification and output specification A +/- 1 as normalized values.

Given a pretrained feedforward/convolutional neural network *f: X -> Y*, where typically assume X = R^n , Y = R^m.

A set of inputs , Xo subset of Y

Compute the set of outputs, *f(Xo)*

Finally check if ~P intersection f(Xo) == NULL ?

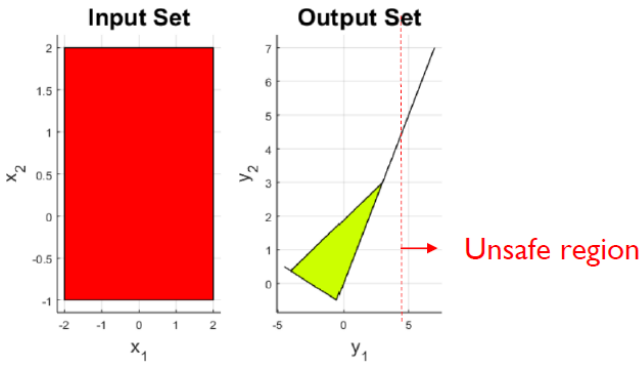

Fig 2 shows an reachable set and unsafe regions

Dataset and preprocessing

The dataset used to verify is credit card usage dataset which has 29 dimensions like time of transaction , amount of transaction etc.

The Autoencoder model is trained using credit card usage dataset which has 160000 records.epoch of 100 , batch size of 256.The trained model is saved in h5 format.The test data consist of 40000 samples.

NNV tool:

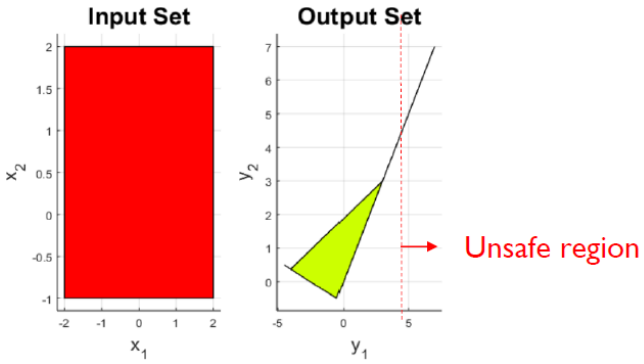

Fig 3 shows an unsafe region

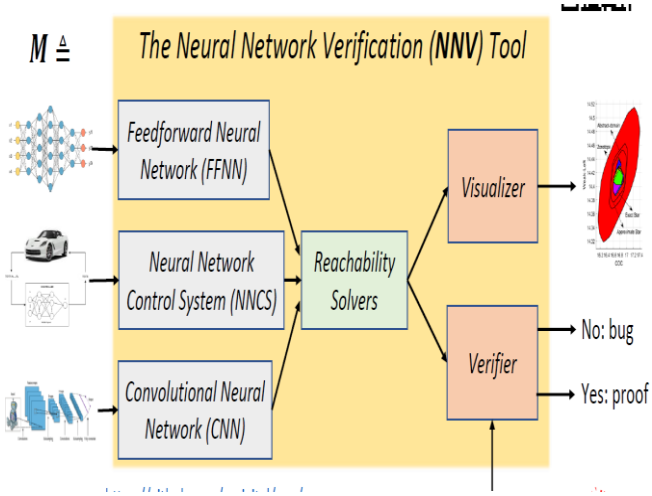

Fig 4 shows an NNV tool block diagram used to verify autoencoder

The keras h5 autoencoder model needs to be imported into the NNV matlab tool using importKerasNetworks API and needs to be converted into .mat file. Then it has to be converted into CNN based autoencoder model.

Input test data:

Then we need to define an normalized input set of upper and lower bounds. The credit card usage dataset has 84807 records as test set.The upper and lower bounds are selected as mean+/- 2 sigma as shown in figure 5. This input set is converted into starset.

A starset can be a complex representation of a bounded convex polyhedron and is tuple of <c,V,P>. where where c belongs to R^n, V = {v1,v2,…,vm} is a set of basis vectors. P(Alpha)

= [alpha1,alpha2,…alpham]' is a predicate variable.

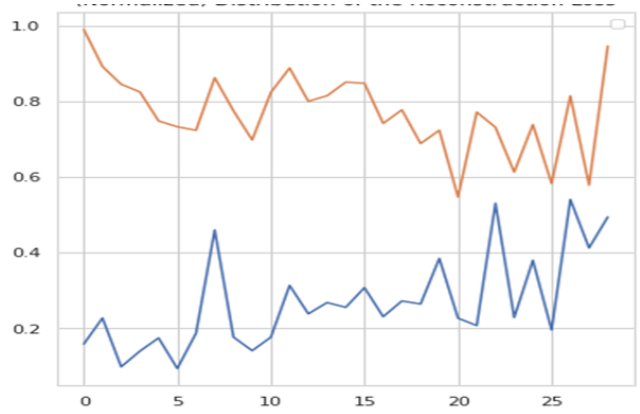

Fig 5 shows input test data upper bound(mean+2 Sigma) and (mean-2Sigma) lower bound.

The input starset is then converted into ImageStar set suitable for feeding into a CNN.

ImageStar sets are an extension of starsets used to represent infinite sets of multi-channel images, But here it is used to represent credit card usage time-series data.

Reachability calculation for input test data using NNV tool :

The imageStar output from nnv tool calculates the normalized output reachability set as shown in figure 6.

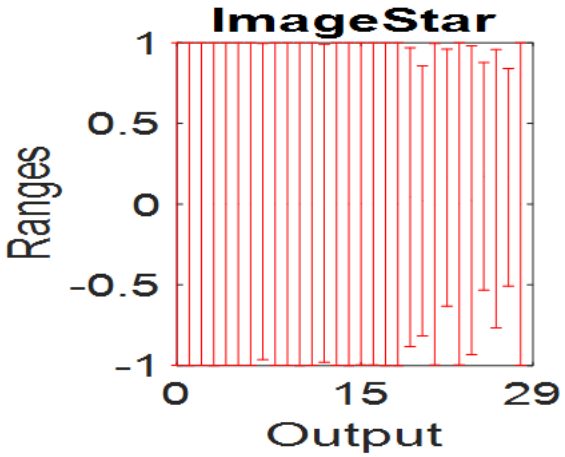

Fig 6 shows an normalized output reachability set for the normalized input test data (ImageStar) generated by NNV tool shown in fig 5.

Output bounds calculation from Keras Autoencoder model:

The output from the Keras autoencoder is analysed and lower and upper bound is as shown in the figure 7.

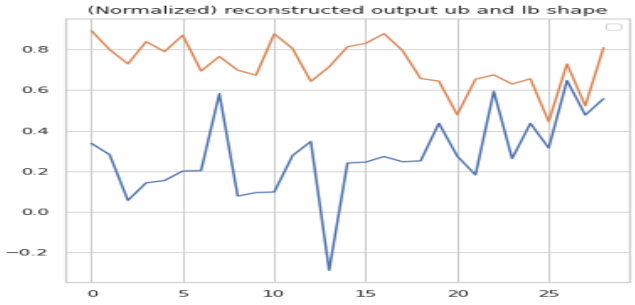

Fig 7 shows an normalized output set generated from the normalized input set by using keras autoencoder model for normalized input shown in fig 4

Method 2(Done by Neelanjana):

Please refer the submission done by Neelanjana in Brightspace to compare her methodology and experiments.

Conclusion:

The ranges of the keras produced output bounds shown in Fig 7 are within the ranges of NNV generated normalized output bounds as shown in Fig 6.

This verifies the input specification that any random test input within +/- 2 sigma bounds as

shown in fig 5 are within the bounds produced by the NNV tool output as show in Fig 6. So this verifies the Autoencoder for the regression task for credit card usage dataset.

REFERENCES

[1] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extract-ing and composing robust features with denoising autoencoders," inProceedings of the 25th international conference on Machine learning,2008, pp. 1096–1103.

[2] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, andL. Bottou, "Stacked denoising autoencoders: Learning useful represen-tations in a deep network with a local denoising criterion."Journal ofmachine learning research, vol. 11, no. 12, 2010.

[3] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-basednetwork anomaly detection," in2018 Wireless Telecommunications Sym-posium (WTS), 2018, pp. 1–5.

[4] M. Chen, X. Shi, Y. Zhang, D. Wu, and M. Guizani, "Deep featureslearning for medical image analysis with convolutional autoencoderneural network,"IEEE Transactions on Big Data, pp. 1–1, 2017.

[5] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deepautoencoders," inProceedings of the 23rd ACM SIGKDD internationalconference on knowledge discovery and data mining, 2017, pp. 665–674.

[6] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders withnonlinear dimensionality reduction," inProceedings of the MLSDA 20142nd Workshop on Machine Learning for Sensory Data Analysis, 2014,pp. 4–11.

[7] W. Sun, S. Shao, R. Zhao, R. Yan, X. Zhang, and X. Chen, "A sparseauto-encoder-based deep neural network approach for induction motorfaults classification,"Measurement, vol. 89, pp. 171–178, 2016.

[8] N. Munir, J. Park, H.-J. Kim, S.-J. Song, and S.-S. Kang, "Performanceenhancement of convolutional neural network for ultrasonic flaw classi-fication by adopting autoencoder,"NDT & E International, vol. 111, p.102218, 2020.

[9] C. Zhang, W. Gao, J. Song, and J. Jiang, "An imbalanced dataclassification algorithm of improved autoencoder neural network," in2016 Eighth International Conference on Advanced Computational Intelligence (ICACI), 2016, pp. 95–99.

[10] H.-D. Tran, X. Yang, D. M. Lopez, P. Musau, L. V. Nguyen, W. Xiang,S. Bak, and T. T. Johnson, "Nnv: The neural network verification toolfor deep neural networks and learning-enabled cyber-physical systems,"inInternational Conference on Computer Aided Verification. Springer,2020, pp. 3–17.

[11] H.-D. Tran, D. M. Lopez, P. Musau, X. Yang, L. V. Nguyen, W. Xiang,and T. T. Johnson, "Star-based reachability analysis of deep neuralnetworks," inInternational Symposium on Formal Methods. Springer,2019, pp. 670–686.

[12] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari, "Learning andverification of feedback control systems using feedforward neural net-works,"IFAC-PapersOnLine, vol. 51, no. 16, pp. 151–156, 2018.

[13] R. Ehlers, "Formal verification of piece-wise linear feed-forward neuralnetworks," inInternational Symposium on Automated Technology forVerification and Analysis. Springer, 2017, pp. 269–286.

[14] W. Xiang, H.-D. Tran, and T. T. Johnson, "Specification-guidedsafety verification for feedforward neural networks,"arXiv preprintarXiv:1812.06161, 2018.

[15] K. D. Dvijotham, R. Stanforth, S. Gowal, C. Qin, S. De, and P. Kohli,"Efficient neural network verification with exactness characterization,"inUncertainty in Artificial Intelligence. PMLR, 2020, pp. 497–507.

[16] H.-D. Tran, S. Bak, W. Xiang, and T. T. Johnson, "Verification ofdeep convolutional neural networks using imagestars," inInternationalConference on Computer Aided Verification. Springer, 2020, pp. 18–42.

[17] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, andH. Chen, "Deep autoencoding gaussian mixture model for unsupervisedanomaly detection," inInternational Conference on Learning Represen-tations, 2018.

[18] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, andM. Vechev, “Ai2: Safety and robustness certification of neural networkswith abstract interpretation,” in2018 IEEE Symposium on Security andPrivacy (SP). IEEE, 2018, pp. 3–18