# Designing Out-of-distribution Data Detection using Anomaly Detectors: Single Model vs. Ensemble

Dejana Ugrenovic[1], Jens Vankeirsbilck[1], Davy Pissoort[2], Tom Holvoet[3] and Jeroen Boydens[1]
[1]Department of Computer Science, M-Group, KU Leuven Bruges Campus
[2]Department of Electrical Engineering, M-Group, KU Leuven Bruges Campus
[3]Department of Computer Science, KU Leuven
{Dejana.Ugrenovic | Jens.Vankeirsbilck | Davy.Pissoort | Tom.Holvoet | Jeroen.Boydens}@kuleuven.be

*Abstract* – **Image classification neural networks tend to give high probabilities to images they in fact do not recognize. This paper compares three approaches to detect such out-of-distribution data: One-Class Support Vector Machine, Isolation Forest and Local Outlier Factor. The experiments show that Isolation Forest outperforms the other two approaches. However, when combining the three algorithms using a majority voter, the results show that this ensemble is better at detecting out-of-distribution data than using the Isolation Forest algorithm solely.**

*Keywords* – **ensemble learning; neural networks; majority voting; out-of-distribution detection; design of detectors.**

## I. INTRODUCTION

Systems, such as biometric identification and medical systems as well as autonomous vehicles, rely on image recognition and classification for their correct and/or safe operation. This task is often performed by neural networks (NNs), which are trained using images of the different classes they are supposed to recognize. NNs are machine learning (ML) algorithms modeled loosely on the human brain, where a computer learns to perform certain tasks by using labeled training samples.

While these NNs perform excellent on images that belong to the classes used in the training process, in this paper referred to as *in-distribution data* (ID), they tend to get confused by images not belonging to the trained classes, in this paper called *out-of-distribution data* (OOD). Unfortunately, in real-case scenarios, the level of control over the input data is low. Therefore, it is very important for classifiers to be aware of new kinds of inputs [1,2]. OOD detectors are algorithms used for monitoring the inputs and outputs of NNs with the task to decide whether a given input is from the ID or from the OOD dataset. The OOD detectors can also be interpreted as an additional binary classification module that aims to predict true if the given data is from the OOD and false if it is not. Mind that, it is crucial to perform such a task without affecting the performance of the NN.

A naive solution to solving the OOD problem is increasing the size of the training data and explicitly adding OOD samples to it. This way the NN can learn to classify whether a test sample is from the ID or the OOD. However, to have a total OOD dataset is impossible, in real-world there is an infinite number of such samples. Furthermore, with the addition of such data, more complex NN architectures may have to be employed to correctly classify the training samples. This could decrease the performance of the NN and makes such an approach computationally expensive.

Hendrycks and Gimpel propose in [3] a baseline method to detect OOD samples by using a simple threshold-based detector mechanism, which requires no additional re-training of a network. Their method is based on the empirical observation that well-trained NNs tend to assign higher softmax probabilities to the ID samples compared to the OOD samples. However, according to the authors, while such an approach is a useful baseline, at times it is less effective, which leaves space for performance improvement. Softmax is a function that converts raw outputs of the last layer of an NN into probabilities by taking the exponents of each output and then normalizing each number by the sum of those exponents. After applying the softmax function, the sum of all the elements of the output vector is one.

Liang et al. further improved Hendrycks and Gimpel's work by applying temperature scaling to the output data and by adding small controlled perturbations to the input data [1]. Temperature scaling is calibrating the softmax outputs by scaling the last layer outputs that feed into softmax by a large constant. It is shown that by applying these methods, the distribution gap of the NN softmax output between the ID and OOD data is further enlarged. While effective, this approach requires access to every OOD dataset to determine the correct amount of perturbation and temperature scaling. In practice, it is possible to perform that on some OOD datasets, but it is impossible to create a complete OOD dataset.

Designing an OOD detector based on the softmax output of a trained NN is indeed a valid approach. However, in contrast to previous work, we propose to design them using anomaly detectors (ADs). One advantage of this approach is that these algorithms, only need one OOD dataset, used in the hyperparameter optimization process.

**The aims of this paper are**: 1) Demonstrate that OOD detectors can indeed be designed using AD algorithms, 2) Analyze which AD algorithm performs best and 3) Propose the creation of an ensemble of AD modules which should be able to outperform the single module configuration.

The remainder of this paper is structured as follows. Section 2 contains a description of the system architecture in which the OOD detector works in parallel with the NN and provides an output whether the input is OOD or not. It also describes briefly the mathematical methods used in the implementation of the AD algorithms. Section 3 provides technical details about the experimental optimization and

978-1-7281-7426-6/20/$31.00 ©2020 IEEE

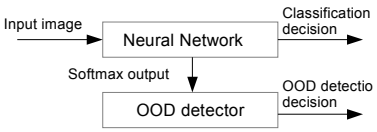


Fig.1. Detection of the OOD data

datasets used for training, validation and testing the AD algorithms. Section 4 presents the experiment results, while conclusions are drawn and future work is described in Section 5 and Section 6, respectively.

## II. Proposed approach

This section presents the proposed OOD approach. First, the architecture of our solution is discussed, to be followed with an explanation of the different selected AD algorithms and techniques.

### *A. Architecture*

Our approach proposes an AD, which is an ML algorithm, that uses the softmax outputs as input for the detection of OOD data. As aforementioned, using the softmax outputs prevents having to re-train the NN itself.

As shown in Fig. 1, this architecture enables the NN to make the classification decision while the AD algorithm determines whether or not the input image is OOD.

### *B. OOD detection techniques*

For the OOD detection, we selected AD algorithms that can be trained on one-class data only, in this case, ID data only. Since in real-world scenarios, it is impossible to predict the OOD data, such types of algorithms are suitable for our problem. We have evaluated the performance of the following algorithms:

- One-Class Support Vector Machine (OCSVM) [4],
- Isolation Forest (IF) [5],
- Local Outlier Factor (LOF) [6] and
- Ensemble of the OCSVM, IF and LOF modules.

The OCSVM algorithm was first proposed by Schölkopf et. al. and in recent years, it has been widely used in various fields. The OCSVM is based on the traditional Support Vector Machine (SVM) algorithm but aims to solve classification problems that use only one type of samples. The OCSVM learns to distinguish between two classes in a training dataset, by fitting a hyperplane that optimally divides both classes [4].

The IF algorithm differs from many other AD algorithms in a way that it tries to detect anomalies rather than profiling normal model behavior. It is based on the idea that anomaly points are much easier to isolate than the normal data points. This algorithm generates partitions of the dataset by randomly selecting a feature and then randomly selecting a split value for that feature. It presumes that anomalies will have less random partitions compared to normal points [5].

The LOF algorithm tries to find anomalous data points by measuring the local deviation of a given data point with respect to its neighbors. The algorithm is based on the idea that the density around an anomaly point is significantly different from the density around its neighbors. It uses the relative density of a sample against its neighbors to calculate the degree of the object being marked as an anomaly [6].

Ensemble techniques, in which the decisions of multiple modules are combined, are widely used in ML approaches to improve the overall performance. In this paper, the ensemble method was implemented by combining OCSVM, IF and LOF modules and using majority voting. Majority voting represents a sum of outputs of individual classifiers, where the output of each classifier has a value of ±1.

## III. Experimental setup

This section contains the information about the experimental setup such as the NN architecture specifications, metrics used for the performance evaluation, descriptions of the datasets and details regarding the training and hyperparameter optimization.

### *A. NN architecture*

Similar to Liang et al., we adopt DenseNet trained on the CIFAR10 dataset as state-of-the-art NN architecture [1,7,8]. The CIFAR10 dataset is organized in 10 classes and consists of 50.000 training images and 10.000 test images. The number of epochs used for the training of the NN was 300, while the batch size was 64 with the momentum of 0. The learning rate starts at 0.1, dropped by a factor of 10 at 50% and 75% of the training progress, respectively. Depth of the NN L=100, growth rate k=12 and dropout rate = 0.

### *B. Evaluation criteria*

The output of the OOD detector can be categorized into four categories, depending on the provided input data:

- True Positive (TP): The detector correctly classifies OOD input data as OOD.
- True Negative (TN): The ID input image is correctly classified as ID by the detector.
- False Negative (FN): Although provided with an OOD image, the detector classified it as ID.
- False Positive (FP): The detector was given an ID image, but it incorrectly classified it as OOD.

A confusion matrix has been used to evaluate the performance of the model with the following attributes: Accuracy (1), Precision (2), Recall (3) and F1Score (4). The Accuracy measures the ratio of correct outputs over the total number of inputs. Precision shows the amount of correctly predicted OOD input data (TP) of all the OOD predictions (TP and FP). Recall presents the ratio of correctly identified OOD input images (TP) from all provided OOD images (TP and FN). F1Score presents the harmonic mean between recall and precision values [9].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precission = \frac{TP}{TP+FP} \quad (2)$$

$$Recall = \frac{TP}{TP+FN} \quad (3)$$

$$F1Score = \frac{2*Precision*Recall}{Precision+Recall} \quad (4)$$

### C. Datasets

Datasets used in the experiments consist of the softmax outputs, generated by applying trained NN to the images from the ID and OOD datasets.

The ID dataset is represented with the CIFAR10 images dataset as the NN from our experiments is trained on that dataset. The CIFAR-10 dataset contains images from the following categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. OOD datasets are described in [1] and provided as a part of their code release:

- The TinyImageNet dataset is a subset of ImageNet images [10]. It contains 10.000 test images from 200 different classes. Some of the classes are comic book, potpie, poodle, lampshade, etc. We constructed two datasets, TINc and TINd, by randomly cropping the image patches to the size 32×32 or downsampling each image to the size 32×32, respectively [11].
- The Large-scale Scene UNderstanding dataset (LSUN) contains a testing set of 10.000 images with the following categories: bedroom, kitchen, living room, dining room, bridge, tower, restaurant, conference room, classroom and church outdoor. [12]. Two datasets are generated from the LSUN testing set. LSUNc dataset is generated by randomly cropping the images to the size of 32x32, while LSUNd dataset is generated by downsampling the images to the size of 32x32.
- The iSUN is a subset of SUN images [13]. The content of this dataset is focused on the scenes (indoor and outdoor images). There are 8925 images in iSUN dataset and each image is downsampled to size 32x32.
- The Gaussian is a synthetic Gaussian noise dataset containing 10.000 random 2D Gaussian noise images, where the RGB value of every pixel is sampled from an independent and identically distributed Gaussian with a mean 0.5 and unit variance. Each pixel of the image is clipped to the value between the range [0, 1].
- The Uniform is a synthetic uniform noise dataset containing 10.000 images, where the RGB value of every pixel is independently and identically sampled from the uniform distribution on [0, 1].

### D. Training and hyperparameter procedure

Most of the ML algorithms have number of parameters and hyperparameters that define the performance of the model. Model parameters are obtained automatically during the training process, whereas model hyperparameters are set manually before the training processes. The process of finding the best set of hyperparameters is called hyperparameter optimization. We used a grid-search algorithm to find the optimal hyperparameters that provide the highest accuracy of a model, for each of the algorithm (OCVM, IF, LOF). Grid-search means that the model is trained and a metric is calculated for every hyperparameter combination from a manually defined set [14].

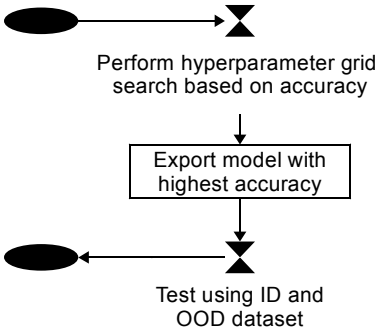


Fig. 2. Training with the hyperparameter optimization and testing procedure

Hyperparameters that were optimized with the corresponding optimal values are:

- OCSVM - nu (0.57), gamma (8.35) and kernel (rbf).
- IF - n_estimators (80), max_features (0.2) and contamination (0.1).
- LOF – contamination (0.1) and n_neighbors (0.2).

Each of the OOD algorithms (OCSVM, IF, LOF) was trained, optimized and tested independently using the process shown in Fig 2. OOD detection algorithms are trained on the ID dataset. The ID dataset is composed of softmax outputs generated from 30.000 CIFAR10 training image dataset. Besides the training dataset, the validation dataset is used in the process of hyperparameter optimization. The validation dataset is composed of one ID and one OOD dataset. It is based on the softmax outputs generated from the 10.000 CIFAR10 training images (ID) and on 10.000 TINc images (OOD). Testing procedure follows after founding the model with the highest accuracy.

## IV. Results

After the optimal hyperparameters were found, the tests are performed on six datasets composed of the ID and the OOD datasets. Each of the six datasets is made as a combination of the softmax outputs generated from the 10.000 CIFAR10 test images (ID dataset) and the different OOD datasets. The test datasets are named as the corresponding input OOD and ID datasets: Cifar10/TINd (CTINd), Cifar10/LSUNc (CLSUNc), Cifar10/LSUNd (CLSUNd), Cifar10/iSUN (CiSUN), Cifar10/Gaussian (CGaussian) and Cifar10/Uniform (CUniform). On each test dataset, we compared the performance between applied single modules and ensemble. The main results are summarized in Table 1. We performed tests on the six datasets mentioned in Section III-D, but Table 1 shows results only for the CTINd, CLSUNd and the CiSUN datasets. The conclusions drawn from the experiments on the other three test datasets are comparable.

Table 1 indicates that AD algorithms can indeed be used to design OOD detectors. The four selected metrics achieve high scores, indicating a good OOD detection performance for these algorithms. Although not the main focus of this paper, the table indicates that the IF algorithm outperforms the OCSVM and LOF algorithms for all four metrics. This shows that IF, or the math behind this approach, seems to be better suited to detect OOD data.

Furthermore, the results show that the ensemble configuration using majority voting outperforms the single module approach with regard to three metrics, i.e. Accuracy, Precision and F1 Score.

Overall, this shows that designing OOD detectors using one or an ensemble configuration of AD algorithms is feasible and even only need one OOD dataset to be optimized to work efficiently.

## V. Conclusion

Systems relying on NN to classify images, need a way to know whether or not the NN actually recognizes the input image. In other words, an approach to detect out-of-distribution data is necessary. In this paper, we proposed to design an OOD detector based on AD algorithms. To validate this approach, we used OCSVM, IF, LOF and an ensemble configuration using majority voting, as OOD detectors. The experiments show that AD algorithms can indeed be used to design OOD detectors. The advantage of these algorithms over previous work is that these only need one OOD dataset instead of multiple such datasets to be fine tuned. For the selected algorithms, IF outperforms OCSVM and LOF for all considered metrics. However, the ensemble configuration achieves higher scores for three of the four considered metrics. This indicates that an ensemble of techniques is better suited to detect OOD data than a single algorithm.

## VI. Future work

In this paper, the AD algorithms were optimized for the accuracy metric. This can be thought of as *being optimized for availability*, since it focuses on TP and TN. From a safety point of view, which is related to the OOD data problem, optimizing for recall and focusing on the low amount of FNs, could be better. Future experiments will indicate which impact this has on the algorithms and the other metrics.

## Acknowledgment

The authors thank Raul Sena Ferreira from LAAS-CNRS, as well as Wim Casteels and his colleagues from the IDLab research group at University of Antwerp, for their valuable feedback on the research presented in this paper.

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No 812.788 (MSCA-ETN SAS).

This publication reflects only the authors' view, exempting the European Union from any liability. Project website: http://etn-sas.eu/.

Table 1. Comparison of the performance of single algorithms and majority voting technique [in %]

| | | OCSVM | IF | LOF | MAJORITY VOTING |
|---|---|---|---|---|---|
| **CTIND** | **Accuracy** | 91 | **93** | 91 | **93** |
| | **Precision** | 89 | 91 | 90 | **95** |
| | **Recall** | 94 | **95** | 92 | 91 |
| | **F1Score** | 92 | **93** | 91 | **93** |
| **CLSUND** | **Accuracy** | 91 | **93** | 91 | **93** |
| | **Precision** | 89 | 91 | 91 | **95** |
| | **Recall** | 93 | **97** | 95 | 94 |
| | **F1Score** | 91 | 94 | 93 | **95** |
| **CISUN** | **Accuracy** | 91 | **93** | 91 | **93** |
| | **Precision** | 88 | 90 | 90 | **95** |
| | **Recall** | 94 | **97** | 95 | 94 |
| | **F1Score** | 91 | 93 | 92 | **94** |

## References

[1] S. Liang, Y. Li, and R. Srikant, "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks," arXiv:1706.02690v4 [cs], 2018

[2] A. Vyas, N. Jammalamadaka, X. Zhu, D. Das, B. Kaul, and Theodore L. Willke, "Out-of-distribution detection using an ensemble of self supervised leave-out classifiers," https://arxiv.org/abs/1809.03576, 2018

[3] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," arXiv:1610.02136v3 [cs], 2018.

[4] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," Neural Computation, vol. 13, no. 7, pp. 1443-1471, 2001.

[5] F. T. Liu, K. M. Ting and Z. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, 2008, pp. 413-422

[6] M.M. Breunig, H. Kriegel, R.T. Ng and J. Sander. "LOF: Identifying Density-Based Local Outliers." ACM SIGMOD, 2000.

[7] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261-2269

[8] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto. 2009.

[9] H. Alves, B. Fonseca and N. Antunes, "Experimenting Machine Learning Techniques to Predict Vulnerabilities," 2016 Seventh Latin-American Symposium on Dependable Computing (LADC), Cali, 2016, pp. 151-156

[10] J. Deng, W. Dong, R. Socher, L. Li, K. Li and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248-255

[11] F. Li, A. Karpathy, and J.Johnson. "Tiny imagenet visual recognition challenge"
Internet: https://tiny-imagenet.herokuapp.com, [June 24, 2020].

[12] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, J. XiaoLsun, "Construction of a large-scale image dataset using deep learning with humans in the loop," arXiv:1506.03365 [cs], 2015.

[13] P. Xu, K. A. Ehinger, Y. Zhang, A. Finkelstein, S. R. Kulkarni, and J. Xiao, "TurkerGaze: crowdsourcing saliency with webcam based eye tracking," arXiv:1504.06755 [cs], 2015.

[14] D. Chicco. "Ten quick tips for machine learning in computational biology." BioData Min. pp. 10-35, 2017.