IMPLEMENTING DEEP ISOLATION FORESTS ON TABULAR DATA

Akella Kalyan Lakshmi Sriniwasa **T.N.Sreevarshan**

ABSTRACT

This is one of the 2 sub reports for the group project which involves the testing of models. We test the Deep Isolation Forests mode in this sub report. The topic of our project is to select 2 papers based on Trajectory Anomaly Detection and replicate the results. The results are benchmarked against the Isolation Forests model and against the Extended Isolation Forests model. This is a significant difference from the original paper, as we do not compare this model against other deep learning models, and we use fewer models and fewer datasets compared to the original paper. Specifically, 3 datasets have been used, which will be detailed in later sections. Synthetic data has also been generated and used for experiments. However, on the datasets used and the tests that have been done, we have been able to replicate the results and show how DIF is a better model compared to the other mentioned model. For the model implementations, we used scikit learn for Isolation Forests, h2o for Extended Isolation Forests and DeepOD for deep isolation forests.We were also able to prove that Deep Isolation Forests is a model with a time complexity that is nearly linear with respect to both number of dimensions and size of the datasets. However, our results have not been extended to graph and time series data and are only valid for tabular datasets.

1 Introduction

Anomaly detection is a crucial machine learning method with diverse applications in fraud detection, medicine, and security. While Isolation Forests[1] has gained popularity for its linear time complexity and lack of assumptions about data characteristics, it struggles with hard anomalies and may misclassify artifacts introduced by the model itself, known as ghost regions. To address these limitations, advanced methods like Extended Isolation Forests[2] have been introduced, although they suffer from low memory bounds compared to Isolation Forests.

Deep Isolation Forests (DIF) represent a promising advancement in anomaly detection, leveraging neural networks to achieve non-linear isolation. Initially introduced in [3], the DIF model offers strong representation power and has demonstrated promising results in various applications. This report aims to replicate and evaluate the performance of the DIF model, benchmarking it against Isolation Forests and Extended Isolation Forests using tabular datasets.

In this technical report, we first provide an overview of the working of both the IF and DIF models, followed by a description of the packages used for implementation and the experimental setup. We then present the results of side-by-side comparisons and discuss the implications and limitations of the models.

Through this exploration, we aim to shed light on the effectiveness of the DIF model and its potential to overcome the shortcomings of traditional anomaly detection methods. Please refer to the original paper [3] and accompanying GitHub repository [4] for further insights.

2 Working of Isolation Forests

As a preliminary step, let's delve into the functioning of the Isolation Forest (IF) model. The IF model consists of isolation trees (iTree), which are binary trees constructed from a subsample of the dataset. During the construction of each tree, a branching criterion η is used to compare the jth dimension of an object o(j). This process continues recursively until all samples are fitted into the tree. Subsequently, the anomaly score for each data point is computed using the formula

$$\mathcal{F}_{iforest}(o|T) = 2^{E_{ti\in T}|p(o|T_i|/c(T)}$$

Here, $E_{ti\in T}|p(o|T_i)$ represents the average path length to the data point across all iTrees, and c(T) is a normalizing constant.

Isolation Forests employ linear partitioning using only one dimension, which can be limiting. In contrast, Extended Isolation Forests (EIF) also utilize linear partitioning but employ hyperplane-based isolation, allowing for consideration of multiple dimensions. This capability enables EIF to handle hard anomalies, which are anomalies that cannot be effectively separated by linear partitioning. However, it's important to note that the algorithmic bias introduced in the isolation strategies may impact the model's efficiency.

3 Working of Deep Isolation Forests

3.1 Theory behind the model

DIF first produces a random representation ensemble with an ensemble size of r, such that each representation contains t iTrees and a forest T containing t*r iTrees.iTree τ_i of X is initialized by a root node and some projected data P1. The k-th node with data pool of Pk is branched into two leaf nodes with disjoint subsets, i.e., P2k = {x|x(jk≤ηk, x ∈ Pk} and P2k+1 = {x|x(jk) > ηk, x ∈ Pk}, where jk is selected uniformly at random among all the dimensions of the newly created data space {1, · · · , d}, x(jk) is the jkth dimension of the projected data object, and ηk is a split value within the range of {x(jk)|x ∈ Pk}. After constructing T , the abnormality of a data object o is evaluated by the isolation difficulty in each iTree of the forest T using the scoring function $\mathcal{F}(o|T) = \Omega_{\tau_i \sim T} I(o|\tau_i)$, where I(o|τi) denotes a function to measure the isolation difficulty in iTree τi, and Ω denotes an integration function. DIF has 2 components, a random representation ensemble forest and an isolation based anomaly scoring function. The time efficiency of the former is improved by the Computation Efficiency Representation Ensemble method(CERE) while the accuracy of the latter is improved by the Deviation Enhanced Anomaly Scoring Function(DEAS). Please refer to the original paper[3] for further reading of the implementation and proof that the time complexity of the DIF model is linear with both data size and number of dimensions.

Algorithm 1 *Construction of Deep Isolation Trees*

Input: $\mathcal{D}$ - input dataset
Output: $\mathcal{T}$ - forest of deep isolation trees

Initialise $\mathcal{T} \leftarrow \varnothing$
Generate representations $\{\mathcal{X}_u\}_{u=1}^{r}$ via $\mathscr{G}_{\text{CERE}}$
for $u = 1$ to r **do**
  for $i = 1$ to t **do**
    Initialise an isolation tree τ_i by setting the root node using $\mathcal{P}_1 \subseteq \mathcal{X}_u$, $|\mathcal{P}_1| = n$
    while $\mathcal{P}_k$ is a leaf node of tree τ_i **do**
      if $|\mathcal{P}_k| > 1$ and the depth is smaller than J **then**
        Randomly select a dimension $j_k \in \{1, \cdots, d\}$
        Randomly select a split point η_k between the *max* and *min* values of dimension j_k in $\mathcal{P}_k$
        $\mathcal{P}_{2k} \leftarrow \{\boldsymbol{x} | \boldsymbol{x}^{(j_k)} \leq \eta_k, \boldsymbol{x} \in \mathcal{P}_k\}$
        $\mathcal{P}_{2k+1} \leftarrow \{\boldsymbol{x} | \boldsymbol{x}^{(j_k)} > \eta_k, \boldsymbol{x} \in \mathcal{P}_k\}$
      end if
    end while
    $\mathcal{T} \leftarrow \mathcal{T} \cup \tau_i$
  end for
end for
return $\mathcal{T}$

Figure 1[7]: Algorithm for construction of deep isolation trees.

Algorithm 2 *Deviation-enhanced Anomaly Scoring*

Input: $\boldsymbol{o}$ - data object, $\mathcal{T}$ - set of deep isolation trees
Output: anomaly score $\mathscr{F}_{\text{DEAS}}(\boldsymbol{o}|\mathcal{T})$

Generate representations $\{\boldsymbol{x}_u\}_{u=i}^{r}$ via $\mathscr{G}_{\text{CERE}}$
for $u = 1$ to r **do**
  for $i = 1$ to t **do**
    Initialise $k \leftarrow 1$, $\beta \leftarrow 0$, $p(\boldsymbol{x}_u|\tau_i) \leftarrow \varnothing$
    while $|\mathcal{P}_k| > 1$ and not reaching J **do**
      if $\boldsymbol{x}_u^{(j_k)} \leq \eta_k$ **then**
        $k \leftarrow 2k$
      else
        $k \leftarrow 2k + 1$
      end if
      $p(\boldsymbol{x}_u|\tau_i) \leftarrow p(\boldsymbol{x}_u|\tau_i) \cup k$, $\beta \leftarrow \beta + |\boldsymbol{x}_u^{(j_k)} - \eta_k|$
    end while
    $g(\boldsymbol{x}_u|\tau_i) \leftarrow \beta / |p(\boldsymbol{x}_u|\tau_i)|$
  end for
end for
return $\mathscr{F}_{\text{DEAS}}(\boldsymbol{o}|\mathcal{T}) \leftarrow 2^{-\mathbb{E}_{\tau_i \in \mathcal{T}} \frac{|p(\boldsymbol{x}_u|\tau_i)|}{C(T)}} \times \mathbb{E}_{\tau \in \mathcal{T}}\big(g(\boldsymbol{x}_u|\tau_i)\big)$

Figure 2: Algorithm behind DEAS method

4 Data and Training Strategy

Data. We first produced 3 synthetic datasets, one dataset is a single blob centered around the origin, the other is a pair of blobs, one centered at (0,10) and the other centered along (10,0). The third is a sinusoidal dataset. These datasets are fed into the three models, namely Isolation forests, Extended Isolation Forests and Deep Isolation Forests. For the next step, we tested the models on 3 datasets actually present in the original paper, named Fraud[5], Shuttle and Pageblocks. Fraud is for fraudulent credit card transaction detection. Pageblocks and Shuttle are provided by an anomaly benchmark study [6]. After this, we generated synthetic data using code from the official GitHub repository from the create_scal_data.py[7] file. 18 datasets were generated.The first 9 datasets had 5000 data points(with 500 anomalies) with dimensions varying from 16 to 4096(in multiples of 2), while the other 9 datasets had 32 features with sizes varying from 1000 to 256,000 and having 5% anomalies. The runtimes of all three models were computed and plotted.

Training. The packages used for the models are: scikit learn[8](for IF model), h2o[9] for EIF and DeepOD[10] for DIF models. The parameters for training all synthetic datasets were set to their normal values. For the 3 actual datasets, the parameters set were as follows:DIF uses 50 representations (r=50) and 6 isolation trees per representation (t=6), with 256 as subsampling size (n=256) for each iTree. IF and EIF methods use 300 trees (the ensemble size is the same as DIF). The subsampling size is set as 256. We use the maximum extension level of EIF, i.e., the extension level is adaptively set as the dimensionality minus 1. AUC ROC and AUC PR are the two metrics used for evaluating the datasets.

5 Results

5.1 Blob Datasets

In testing with blob datasets, we discover the ghost regions in isolation forests and we also observe how the EIF and models deal with these issues for all three datasets.

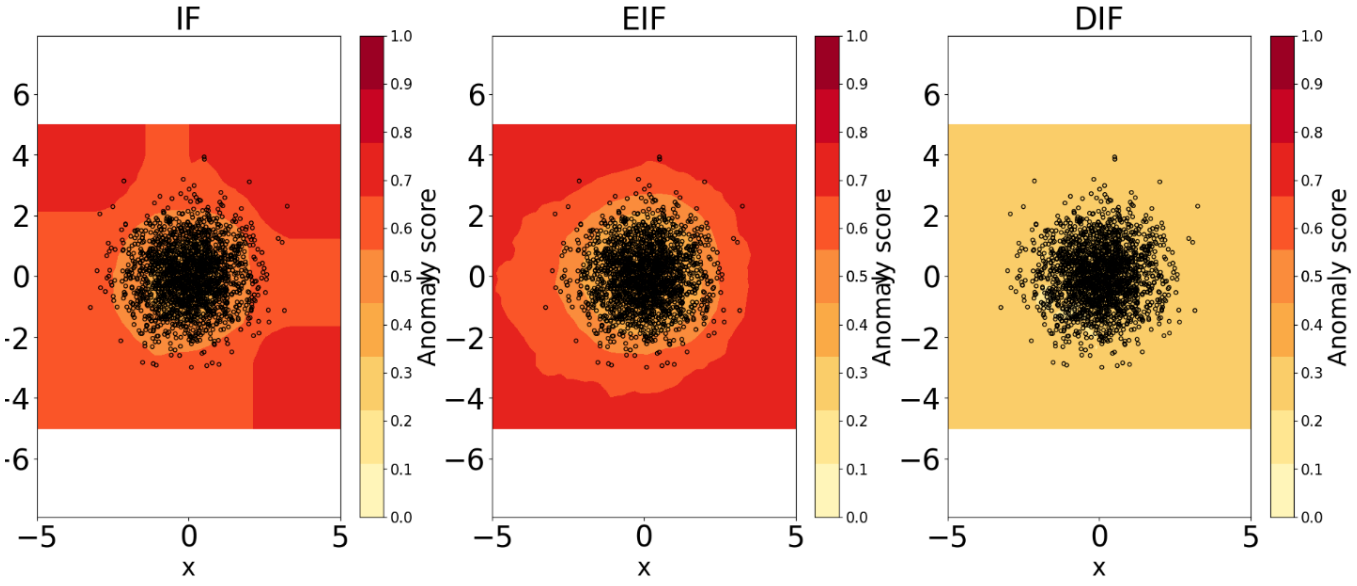

fig 3: Anomaly score heatmap for single blob dataset.

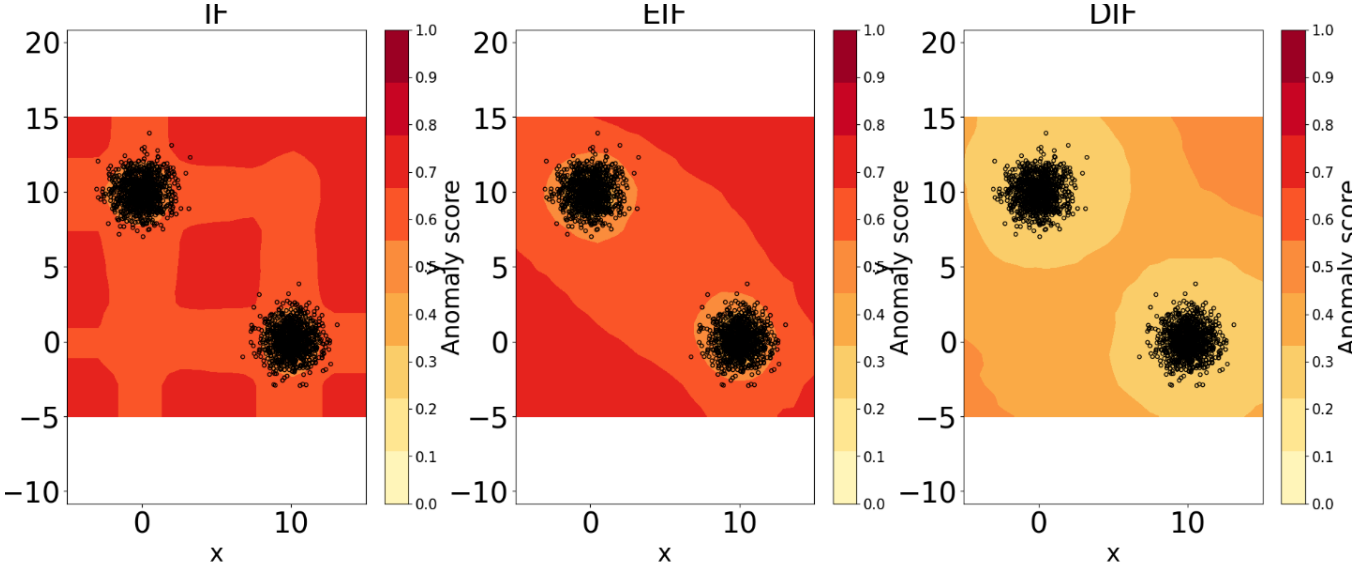

fig 4: Anomaly score heatmap for double blob dataset

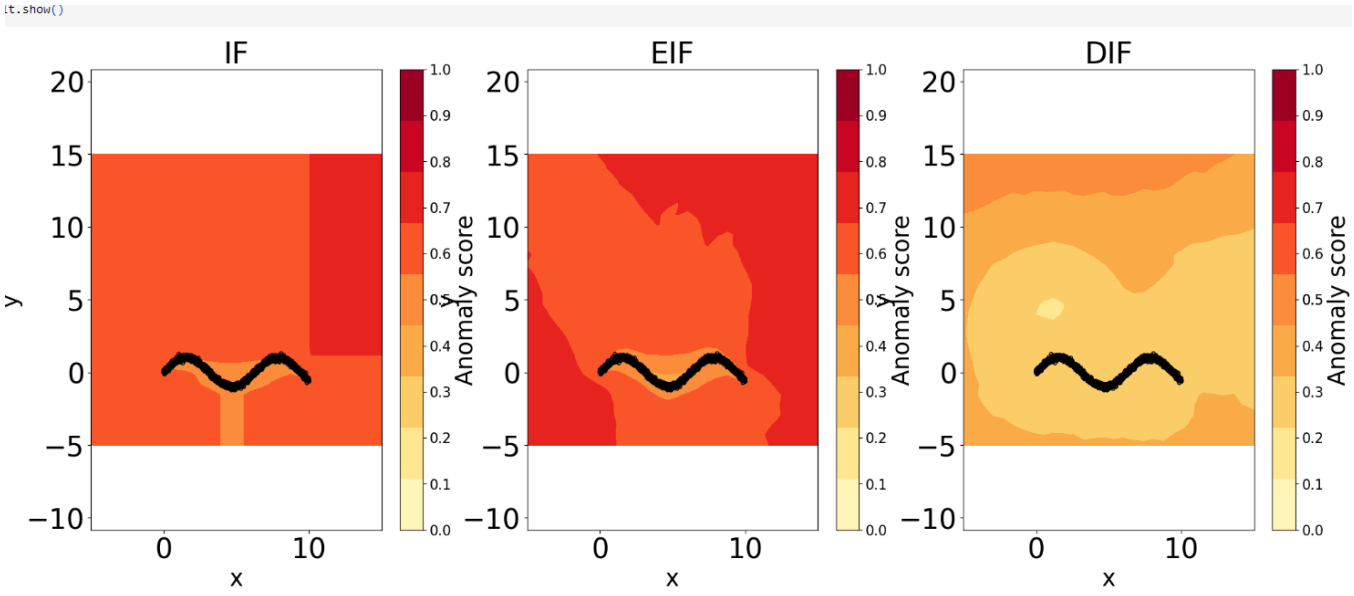

fig 5: Anomaly score heatmap for sinusoidal dataset

5.2 Time Complexity of the models

On the synthetic datasets, the models fit, and the time for this was noted. If the dataset is too large, the runtime was saved as 10^9 seconds. The graphs plotting the dimensionality and size against the time are log scale on the y axis and linear on x axis.

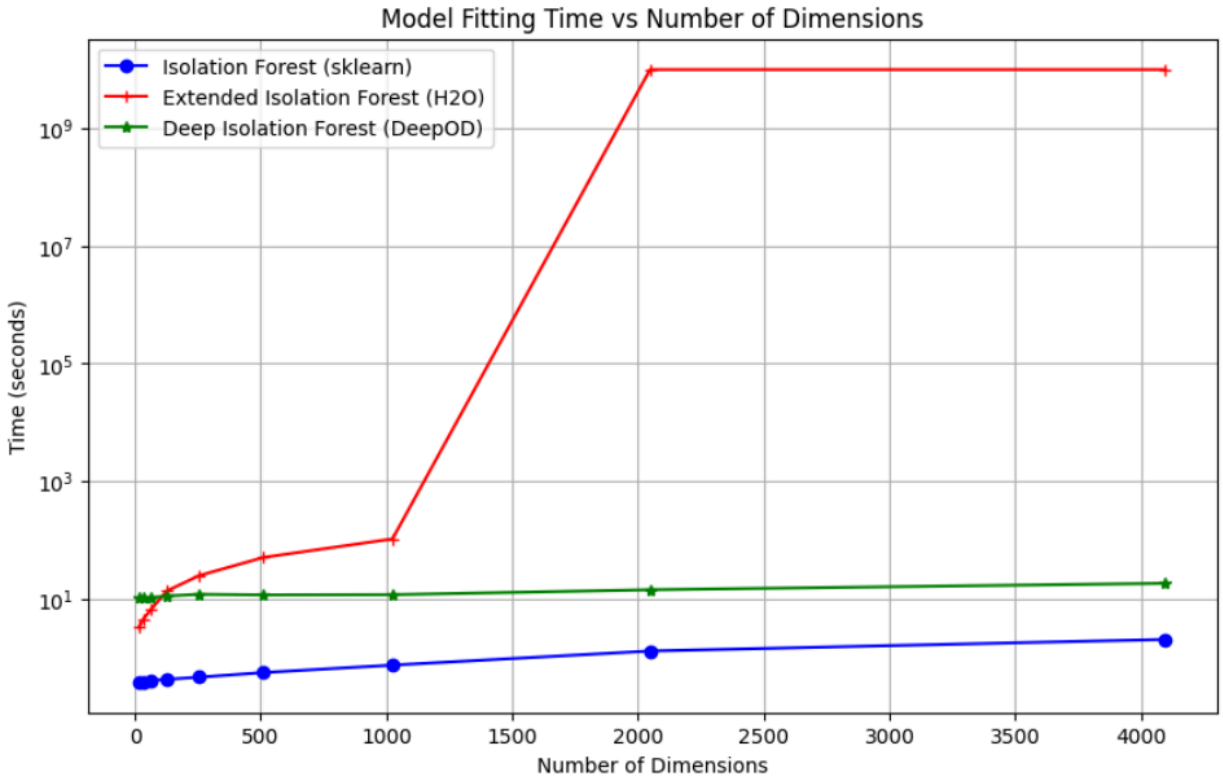

fig 6 : Plot of dimensionality vs training time for the models.

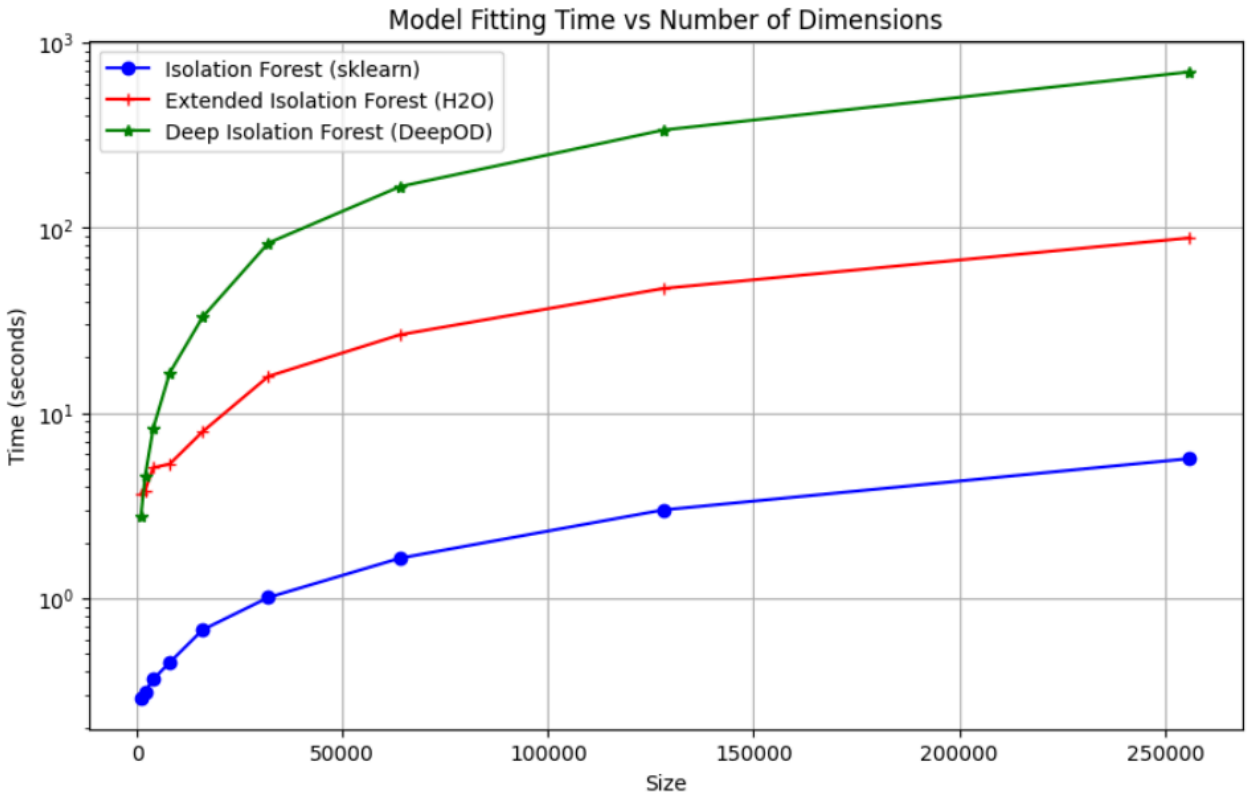

Figure 7: Plot of dataset size against training time

5.3 Comparison of DIF against IF and EIF on datasets

The datasets used are as previously mentioned. For these datasets, while the obtained AUC ROC and AUC PR scores are significantly worse when compared to the results obtained in the paper, the general trends that are expected mostly hold true. Two tables are shown below summarizing the results.

Dataset	IF score	EIF score	DIF score
Fraud	0.3628	0.5	0.5634
Shuttle	0.5049	0.5	0.9430
Pageblocks	0.3429	0.5	0.6949

Table 1: AUC ROC scores for the three models for the different datasets used

Dataset	IF score	EIF score	DIF score
Fraud	0.0013	0.0017	0.0022
Shuttle	0.00497	0.004972	0.04167
Pageblocks	0.0789	0.09916	0.2597

Table 2: AUC PR scores of different models against for the datasets

6 Summary of Results

The results obtained using both the synthetic and actual datasets are summarized in the above graphs and tables. For the actual datasets, it is evident that DIF performs significantly better than either EIF or IF models. While the AUC ROC and AUC PR scores obtained in our experiments are significantly worse that those obtained in the paper, it is evident that the general trends that were obtained are the same as in the paper, which could simply be because of poor implementation of the code. For analyzing the time complexity, we assumed that if a model ran out of memory, the time taken was stored as 10^9 seconds. We discovered that the graphs are roughly logarithmic, i.e.

$$log(time) \propto log(dimensionality\ or\ size)$$

This implies that the time is linear with both dimensionality and size. The EIF model ran out of memory on the high dimensionality datasets, however, this is probably because of the h2o package not using sufficient storage space during rather than the EIF model having space constraints. We have also observed that the DIF model runs faster than the EIF model when a GPU is used as the accelerator.

7 Limitations

Some limitations observed in this study are:

1. Poor implementations of the models leading to much lower scores than the original paper.
2. Limited number of datasets being used compared to the original study.
3. The models implemented were only for tabular data, so it is unknown how effective the models are on graph or time series data.
4. Due to the original GitHub packages not installing on our devices, we were forced to use other packages. While this did not affect DIF much due the original authors suggesting DeepOD as an alternative, it caused many issues in the other models, like the EIF implementation from the h2o package running out of memory during the dimensionality time series analysis.

8 Conclusion

In conclusion, our study focused on implementing and evaluating the Deep Isolation Forests (DIF) model on tabular datasets, comparing its performance against the traditional Isolation Forests (tIF) and Extended Isolation Forests (EIF) models. Our findings underscore the superiority of the DIF model, which effectively addresses inherent flaws in the tIF model while demonstrating enhanced efficiency compared to both tIF and EIF models. Notably, DIF exhibits a linear time complexity with respect to dataset size and dimensionality, offering promising scalability for large-scale applications. However, our study also revealed limitations, including a restricted dataset selection compared to the original paper, potential implementation discrepancies leading to inferior performance relative to the paper's results, and the utilization of alternative open-source packages rather than the authors' implementations. Despite these constraints, our study successfully replicated the results reported in the original paper, validating the robustness of our findings. Moving forward, addressing these limitations through further experimentation and refinement of implementation methodologies could provide deeper insights into the efficacy of the DIF model across diverse datasets and scenarios.

References

[1] F. T. Liu, K. M. Ting and Z. -H. Zhou, "Isolation Forest," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17. keywords: {Application software;Credit cards;Detectors;Constraint optimization;Data mining;Information technology;Laboratories;Isolation technology;Performance evaluation;Astronomy;anomaly detection;outlier detection;novelty detection;isolation forest;binary trees;model based},

[2] S. Hariri, M. C. Kind and R. J. Brunner, "Extended Isolation Forest," in IEEE Transactions on Knowledge and Data Engineering, vol. 33, no. 4, pp. 1479-1489, 1 April 2021, doi: 10.1109/TKDE.2019.2947676. keywords: {Forestry;Vegetation;Distributed databases;Anomaly detection;Standards;Clustering algorithms;Heating systems;Anomaly detection;isolation forest},

[3] H. Xiang, H. Hu and X. Zhang, "DeepiForest: A Deep Anomaly Detection Framework with Hashing Based Isolation Forest," 2022 IEEE International Conference on Data Mining (ICDM), Orlando, FL, USA, 2022, pp. 1251-1256, doi: 10.1109/ICDM54844.2022.00163. keywords: {Deep learning;Training;Representation learning;Neural networks;Memory management;Forestry;Feature extraction;Deep neural networks;anomaly detection;DeepiForest;tree-embedding;cascaded architecture},

[4] https://github.com/xuhongzuo/deep-iforest/tree/main

[5] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

[6] G. O. Campos, A. Zimek, J. Sander, R. J. Campello, B. Micenkova, ´ E. Schubert, I. Assent, and M. E. Houle, “On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study,” Data Mining Knowl. Discovery, vol. 30, no. 4, pp. 891–927, 2016.

[7] https://github.com/xuhongzuo/deep-iforest/blob/main/create_scal_data.py

[8] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[9] https://github.com/h2oai/h2o-3/blob/master/h2o-py/demos/extended-isolation-forest-introduction.ipynb

[10] https://github.com/xuhongzuo/DeepOD/blob/main/deepod/models/tabular/dif.py