Meta-ensemble methods for outlier detection in real estate*

Viktor Khomyshyn[1,*,†], Oleh Pastukh[1,†], Vasyl Yatsyshyn[1,†], Ihor Baran[1,†] and Yuriy Ushenko[2,†]

[1] *Ternopil Ivan Puluj National Technical University, Ruska str., 56, 46001, Ternopil, Ukraine*
[2] *Yuriy Fedkovych Chernivtsi National University, Kotsiubynskoho str., 2, 58012, Chernivtsi, Ukraine*

Abstract

Ensemble technology is widely used for many data mining tasks, such as classification, clustering, and regression. Many ensemble-based algorithms have been proposed in the scientific literature for these tasks. Compared to clustering, classification, and regression problems, ensemble analysis is underrepresented in anomaly detection studies. In some cases, ensemble techniques have been implicitly used by many outlier analysis algorithms, but this approach is often hidden deep in the algorithm. And this is despite the fact that this issue is quite important in the context of outlier analysis. In this paper, a method for constructing an ensemble of several ensemble models (meta-ensemble) and a multi-level hierarchical ensemble with bypass connections is proposed. The proposed approach demonstrated higher values of AUC-ROC and P@n metrics on a real real estate dataset compared to classical ensembles. This indicates its ability to effectively generalize information and reduce the impact of individual errors of the underlying algorithms. Despite the somewhat longer computation time, the increase in detection accuracy makes this approach appropriate for tasks where the quality of anomaly detection is crucial. It has been found that the accuracy of outlier detection by meta-ensemble, measured through the specified metrics, significantly depends on the input data preparation techniques, in particular the method of encoding categorical features and scaling, as well as on the optimization of hyperparameters of basic or ensemble algorithms at the stage preceding the construction of the meta-ensemble.

Keywords

machine learning, meta ensemble learning, outlier detection, anomaly detection, real estate, CEUR-WS [1]

1. Introduction

Ensemble learning is one of the most powerful machine learning techniques and is used to improve the accuracy and reliability of models. This technology is based on the fact that instead of relying on a single model, ensemble learning combines (averages) the predictions of several models and forms a more accurate final result. The idea of ensemble learning is that by combining the strengths of several models (weak learners), the ensemble can potentially produce better results than any individual model. In this case, the weaker learners can correct each other's mistakes, which in turn leads to the creation of a stronger learner. Ensemble methods are useful when a single model may not cope with the complexity of the data set or when different models have complementary strengths. Ensembles can use several different algorithms, or variations of the same algorithm with different hyperparameters, or a combination of these options. At the same time, ensemble methods help reduce overfitting by smoothing out noisy predictions.

In any machine learning model, the total error is defined as the sum of three components [1]:

* *ITTAP'2025: 5th International Workshop on Information Technologies: Theoretical and Applied Problems, October 22-24, 2025, Ternopil, Ukraine, Opole, Poland*
[1,*] Corresponding author.
[†] These authors contributed equally.
homyshyn@gmail.com (V. Khomyshyn); oleg.pastuh@gmail.com (O. Pastukh); vyatcyshyn@gmail.com (V. Yatsyshyn); ihor.remm@gmail.com (I. Baran); y.ushenko@chnu.edu.ua (Y. Ushenko)
0000-0003-4369-501X (V. Khomyshyn); 0000-0002-0080-7053 (O. Pastukh); 0000-0002-5517-6359 (V. Yatsyshyn); 0000-0002-8153-2476 (I. Baran); 0000-0003-1767-1882 (Y. Ushenko)

$$Error = Bias^2 + Variance + Irreducible\ error\ . \quad (1)$$

Bias arises from incorrect assumptions made during the training phase of a model and is defined as the average difference between the predicted values of the model and the actual values of the target variable. The larger the bias, the less accurately the model predicts on the training data set. A high bias value can cause the training algorithm to miss important information and correlations between independent variables and class labels, causing the model to underfit.

Variance determines how sensitive the model is to small changes in the training data, that is, how much the model's predictions change when using different training samples. High variance makes the model sensitive to noise in the data set, the model generalizes poorly to new, unknown data, and is prone to overfitting.

Irreducible error is the part of the total error that cannot be reduced because it is caused by randomness in the data itself and arises from noise or unknown factors that affect the data.

One of the key aspects of ensemble learning is the trade-off between bias and variance. This is a well-known problem in machine learning and a fundamental principle of many regularization methods. The optimal balance between bias and variance is a key factor in building effective machine learning models and an important criterion that distinguishes a robust model from a weak one. In machine learning, models with high bias tend to have low variance and vice versa [2].

The optimal balance between bias and dispersion can be achieved by:

- increasing the amount of data, which allows reducing the impact of random noise in the training sample;
- using regularization, which limits the complexity of the model and reduces the risk of overfitting;
- hyperparameter optimization, which allows you to find the best ratio of model complexity and its stability;
- using ensemble and meta-ensemble technologies by combining multiple models or ensembles to improve generalization.

2. Related works

All ensembles consist of separate machine learning algorithms, which are called base algorithms, base learners, and less often base estimators, detectors. All these terms are usually used interchangeably in the literature. Base algorithms are divided into weak learners and strong learners. Weak learners are those that give a result slightly better than random guessing (for example, a shallow decision tree). In binary classification problems, weak classifiers are formally considered to be those that achieve approximately 50% accuracy. Strong learners are those that demonstrate high prediction accuracy (for example, a deep decision tree). In binary classification, they are formally considered to have an accuracy of 80% and higher. Ensemble methods, depending on the ensemble technology, can use both weak (mainly) and strong learners as base ones [3].

There are several popular ensemble techniques. Let's consider the features of each of them:

1. Bagging (Bootstrap Aggregating) – this method involves training multiple instances of the same base model using different subsets of the training data [4]. These subsets are created using bootstrap – a random sample of the training data with replacement. Each subset may contain repeated instances, and some instances may be omitted. The predictions of individual models are combined, often by majority voting for classification tasks or averaging for regression tasks. The main advantage of bagging is that it reduces the variance of the predictions, thereby helping to solve the problem of overfitting [5].
2. Voting is a simple ensemble technique in which several models are trained independently using the same training data set. Typically, the base models are different algorithms, or

more rarely, variations of the same algorithm with different settings. For voting to be effective, the base models must be diverse, that is, have different strengths and weaknesses. In binary classification problems, it is recommended to use an odd number of models to avoid coincidence in determining the majority class. The final result of the ensemble can be determined by voting or weighted voting (in classification problems) or by averaging or weighted averaging (in regression problems) [6].

3. Boosting is an iterative method that is based on training weak learners sequentially and giving more weight to misclassified instances in each iteration. Each new model seeks to compensate for the shortcomings of all previous algorithms by effectively learning from their mistakes. Unlike bagging, which focuses on reducing variance, boosting focuses on reducing bias while creating a robust overall model. The advantage of boosting is that it iteratively focuses on cases that were difficult for previous models, which leads to better generalization to unknown data. Boosting can also capture complex relationships in data by combining the strengths of multiple models. More thorough experiments have shown that boosting sometimes overtrains [7].
4. Stacking is another way of combining different models, which introduces the concept of a meta-learning algorithm. The meta-model learns to combine the predictions of the base models and produces a final prediction. This usually leads to better performance than combining the predictions of individual models by voting or averaging [8]. Stacking allows you to experiment with combinations of different base models and meta-models, creating a hierarchical learning structure, which makes it a versatile ensemble technique.
5. Blending is an ensemble method that is a simplified version of stacking. The training data is divided into two parts: training and validation (hold-out cross-validation). Each base model is trained on the training data and makes a prediction on the validation data, the results of which are used as a new feature for the meta-model. The target variable is added to the predictions on the validation set and the resulting dataset is used by the meta-model for the final prediction. The advantage of blending over stacking is that using a separate “holdout” set to generate predictions for the base models helps to avoid overfitting, which can occur with stacking, when the same data is used for training and prediction.

Model combination is a key element of ensemble learning. To facilitate this process, an easy-to-use Python toolkit, Combo, has been proposed, which is used to aggregate models and estimates for various scenarios, including anomaly detection, clustering, classification, and initial estimation [9]. Combo provides a unified and consistent way to combine both raw and pre-trained models from popular machine learning libraries, including Scikit-learn, XGBoost, and LightGBM. Combo's performance is optimized using JIT and parallelization where possible, using numba and joblib libraries.

In [10], a comparison of three popular ensembles is presented, namely: Bagging, Boosting, and Stacking.

In [11], an approach to optimizing outlier detection ensembles for datasets with a small proportion of outliers (1-10%). Optimized outlier detection ensembles consist of outlier detection algorithms that estimate outliers and use tunable parameters. Automatic optimization determines parameter values that improve the discrimination between inliners and outliers. This increases the efficiency of outlier detection. Outliers are by definition rare, making optimization using a small number of samples useful.

In machine learning, “meta-model” and “meta-ensemble” are related but not identical concepts. A meta-model is a higher-level model that aggregates the output predictions of base models to give a final prediction. Meta-ensemble is an approach to building multi-level ensembles, i.e. an even more complex hierarchy of models (e.g. Stacking over Bagging and Boosting). Each sub-ensemble can use its own aggregation mechanism, such as voting, averaging, etc. The final meta-model combines the outputs of these ensembles, providing additional adaptability to the context of the problem. The main differences between these concepts are summarized and presented in Table 1.

Table 1
Key differences between meta-model and meta-ansemble

Parameter	Meta-model	Meta-ansemble
Definition	A specific model as a component of an ensemble system	Architecture, strategy
Appointment	Generalization optimization, correction of weak predictions	Improving accuracy, stability, generalization
Architecture	One level: model over predictions of other models	Multi level
Algorithm type	Usually regression, logistic regression, XGBoost, NN	Any ensemble strategy with model levels
Flexibility	Limited	Higher due to hierarchy
Control over learning	Learned separately from the base models	Coordinated as part of ensemble training
Computational complexity	Low	High
Using	Stacking, Blending	Complex ensemble systems

The number of scientific publications devoted to meta-ensemble is somewhat smaller, but several works are worth considering.

In [12], a new meta-ensemble learning approach is proposed, which is built on a multi-input multi-output (MIMO) configuration. This approach is easily applicable to existing meta-learning algorithms. Several subnetworks in a single model are trained simultaneously on multiple episodes and form an ensemble of predictions, using the capabilities of the model. Meta-ensemble learning is shown to achieve significant generalization improvements and increase the performance of meta-learning algorithms on classification tests.

In [13], a meta-optimization method is proposed, which is based on constructing a situational meta-problem for the interaction of several initial models during meta-training. Subsequently, during the meta-testing period, a meta-optimization method is implemented to minimize the loss of inter-model interaction during prediction, so that the interaction of several models can provide training on new data sets.

In [14], a meta-ensemble method called improved spatial forest is considered, which adds generated and more accurate features to the original features. New features are obtained from randomly selected original features. If the new features are more noticeable than the original ones, they are chosen by the learners. Thus, the ensemble can have more accurate base learners. However, a separate improved space is generated for each learner to create diversity. The author compared the original and improved spatial versions of the bagging, random forest, and rotational forest algorithms. The improved spatial versions generally have better or comparable results than the original ones. A theoretical framework for analyzing individual accuracy and differences of the base learning algorithms is also presented. The proposed method can be used with different ensemble methods.

In [15], two new approaches to automatically construct the structure of an ensemble of classifiers are proposed, which use meta-learning to determine three important parameters: the

type of classifier, the number of base classifiers, and the aggregation method. The main goal is to provide a reliable and stable structure in a simple and fast way. Empirical analysis has shown that the proposed approach allows creating reliable ensembles of classifiers for most of the analyzed cases.

In [16], the input data is considered at several levels of the hierarchy to determine the presence of outliers at these levels. This approach creates a tree-like structure of the outliers in a hierarchical order. Experiments with a real dataset have provided important information on the characteristics of the outliers, and numerical results have confirmed the effectiveness of this approach.

3. Research methodology

3.1. Previous research in this area

This work is the next sequential stage in the construction of effective anomaly detection techniques. Our previous research in this area [17] included a comprehensive evaluation of 23 outlier detection algorithms on a real dataset. The analysis used data from the real estate market of Ternopil, in particular the segment of apartment and room sales. The work also described the developed software for data collection and the dataset, which contains 760 real estate objects with 12 features. In this study, it is similar.

The results obtained allow us to single out from the total set the algorithms that provided the highest AUC-ROC and Precision @ Rank n metrics, optimal execution time and stability of the obtained results. In this study, we will consider a dataset with a proportion of anomalous objects of 10%, since, as our previous study showed, the efficiency of the algorithms on datasets with a higher proportion of anomalous objects does not change significantly.

In addition, it was found that each algorithm provides the best result when working in conjunction with a specific categorical feature encoder – Label Encoder or One-Hot Encoder. Of the 23 algorithms studied by us in [17], six were selected: CBLOF, FastABOD, KDE, kNN, OCSVM and QMCD. We will use these algorithms as the basis for building ensembles in the future.

The data set was standardized using the RobustScaler scaler, which is robust to outliers. However, research [18] indicates that the choice of scaling method affects the performance of machine learning algorithms, and the difference in performance between the best and worst scaling methods is relevant and statistically significant in most cases. The authors also show how the change in the performance of an ensemble model, considering different scaling methods, is determined by the performance of its underlying models.

3.2. Advanced analysis of basic algorithms

Given the data provided, at the first stage we investigated the selected basic algorithms in more detail, conducting testing using two methods of encoding categorical features - label encoding and unitary encoding and alternately using data set standardization (RobustScaler scaler) and unscaled data (in total - 4 options for representing the same data set). Testing for each data option was repeated 100 times, the data were averaged and considered the final result of the experiment. For all the algorithms studied, their standard hyperparameter settings were used for the purpose of fair comparison (Table 2).

The experiments were conducted on a PC with the Windows 10 Pro x64 operating system, an Intel(R) Core(TM) i3-10105F 3.70 GHz processor, and 32 GB of RAM.

Table 2
Results of basic outlier detection algorithms

Basic algorithm	Data scaling	Categorical feature encoder					
		Label Encoder			One-Hot Encoder		
		AUC-ROC	P@n	Time, ms	AUC-ROC	P@n	Time, ms
Standard hyperparameter settings							
CBLOF	yes	0.825	0.470	4.9	0.798	0.449	7.5
	no	0.768	0.536	4.6	0.766	0.532	6.3
FastABOD	yes	0.851	0.461	96	0.817	0.329	117
	no	–	–	–	0.726	0.408	126
KDE	yes	0.815	0.447	45	0.849	0.474	102
	no	0.565	0.000	11	0.623	0.000	25
kNN	yes	0.833	0.447	8.6	0.841	0.487	8.0
	no	0.789	0.526	7.0	0.788	0.526	6.8
OCSVM	yes	0.802	0.434	42	0.828	0.421	47
	no	0.437	0.066	105	0.553	0.132	113
QMCD	yes	0.755	0.303	3.9	0.882	0.500	12.4
	no	0.755	0.303	4.0	0.882	0.500	12.6
Algorithms with optimized hyperparameters							
CBLOF⁺*	no	0.889	0.658	7.2	0.889	0.658	8.9
FastABOD⁺	yes	–	–	–	0.841	0.434	720
kNN⁺	no	0.890	0.632	9.3	0.890	0.632	8.9
OCSVM⁺	yes	–	–	–	0.841	0.447	76

* The ⁺ symbol in the algorithm designation indicates that hyperparameter optimization has been performed.

Given the results obtained, it makes no sense to conduct research on unscaled data, since almost all algorithms showed a higher AUC-ROC for standardized data. The exception is the QMCD algorithm, the performance of which does not depend on scaling. However, we obtain slightly different results when hyperparameter optimization is performed on the algorithms under study. In our study, the Optuna framework was used for this task [19]. The goal of optimization is to obtain the maximum value of the AUC-ROC metric. If the increase in AUC-ROC by the algorithm as a

result of hyperparameter optimization was less than 0.01, it was considered that the increase in accuracy was insignificant and such a result was not included in Table 2.

The hyperparameter search space was set individually for each algorithm. The following optimal hyperparameter values were obtained:

- algorithm CBLOF: n_clusters=25, alpha=0.65;
- algorithm FastABOD: n_neighbors=15;
- algorithm KDE – optimization did not increase accuracy (less than 0.01);
- algorithm KNN: n_neighbors=31, method='median';
- algorithm OCSVM: kernel='rbf', nu=0.99, degree=2;
- algorithm QMCD – has no parameters for optimization.

In general, four of the six selected algorithms after hyperparameter optimization showed higher values of the AUC-ROC metric, and the maximum values of this value for the CBLOF$^+$ (AUC-ROC = 0.889) and kNN$^+$ (AUC-ROC = 0.890) algorithms were obtained precisely on unscaled data, which indicates the importance of including unscaled data in the analysis of the performance of anomaly detection algorithms.

3.3. Grouping of basic algorithms

According to the results presented in Table 2, the efficiency of outlier detection algorithms (according to the AUC-ROC indicator) is largely determined by the applied data preprocessing methods, in particular categorical feature coding and scaling. Therefore, in the future, depending on the set of data preprocessing methods (DPM), we will use four groups:

- Group 1 – Label Encoder, Robust Scaler;
- Group 2 – Label Encoder, without data scaling;
- Group 3 – One-Hot Encoder, Robust Scaler;
- Group 4 – One-Hot Encoder, without data scaling;

Each of the studied algorithms, which showed AUC-ROC > 0.80, will be assigned to the group in which it showed the highest detection accuracy. Please note that the same algorithm, if it provides a high AUC-ROC within a given set of data preprocessing methods, can be assigned to two groups simultaneously. If within a certain group the optimized version of the algorithm showed a higher AUC-ROC value, it replaces the basic one with default settings. To ensure the convenience of further analysis within each group, the basic algorithms are presented with the AUC-ROC metric indicated in brackets in descending order. If several algorithms have the same AUC-ROC value, we present them in descending order of the P@n metric. The results obtained are presented in Table 3.

Table 3
Ensemble groups of basic algorithms

Data scaling	Categorical feature encoder			
	Label Encoder		One-Hot Encoder	
yes	DPM group 1	FastABOD (0.851) kNN (0.833) CBLOF (0.825) KDE (0.815) OCSVM (0.802)	DPM group 3	QMCD (0.882) KDE (0.849) kNN (0.841) OCSVM$^+$ (0.841) FastABOD$^+$ (0.841)
no	DPM	KNN$^+$ (0.890)	DPM	KNN$^+$ (0.890)

			group 4	CBLOF⁺ (0.889)
	group 2	CBLOF⁺ (0.889)		QMCD (0.882)

3.4. Choosing ensemble algorithms

The next step is to test the effectiveness of ensemble methods for outlier detection. In our study, ensemble methods from the PyOD (Python Outlier Detection) library [20] are used. In general, all ensemble algorithms from this library can be divided into 3 categories according to the number of base learners (Figure 1):

- algorithms that do not require a base learner to work: INNE, LODA, IForest;
- algorithms that specify only one base detector: Feature Bagging (base_estimator parameter, by default – LOF detector);
- algorithms that allow you to specify a list of base detectors: LSCP (detector_list parameter), SUOD (base_estimators parameter).

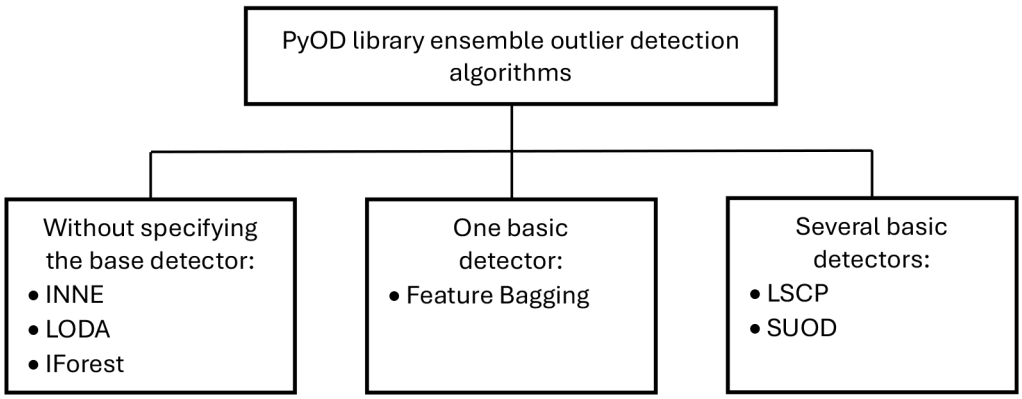

Figure 1: PyOD library ensemble outlier detection algorithms.

The analysis of the performance of ensemble algorithms of the first two categories has been considered in the publication [21]. In this paper, we will investigate the performance of algorithms of the third category. Their characteristics are given in Table 4.

Table 4
Research on ensemble algorithms for outlier detection

Ensemble algorithm	Decryption	Type	Year	Multicore	Source
LSCP	Locally Selective Combination of Parallel Outlier Ensembles	Unsupervised	2019	No	[22]
SUOD	Accelerating Large-scale Unsupervised Heterogeneous Outlier Detection	Unsupervised	2021	Yes	[23]

The LSCP algorithm uses a local approach to improve the accuracy and stability of the ensemble. The principle of its operation is to combine the results of several basic anomaly detection algorithms, taking into account the local context of each point, rather than the global

one. That is, instead of averaging or voting across all detectors (as in traditional ensembles), LSCP selects the most relevant (competent) models for each object, based on its local environment. This allows us to take into account the different effectiveness of models in different regions of the data space.

The SUOD algorithm works by efficiently detecting anomalies (outliers) in large datasets by running multiple base detectors in parallel and aggregating their results without significant time and memory consumption. SUOD first analyzes the input data (statistically or through pre-training) and selects the most relevant detectors from a collection of possible ones (e.g., accurate, fast, sensitive to different types of anomalies, etc.). The selected algorithms are run in parallel, usually using the Joblib library or the Multiprocessing module in Python, which significantly speeds up the process, especially on multi-core machines. After each base detector has provided its anomalous estimates, SUOD aggregates these values by averaging the results from all base detectors (parameter combination="average") or maximizing the values from all base detectors (combination="maximization"). This stage creates a single final anomaly estimate for each object.

3.5. Building ensembles based on LSCP and SUOD algorithms

For each of the four ensemble groups presented in Table 3, a set of basic algorithms was determined experimentally, using the LSCP and SUOD ensemble algorithms, which provided the highest AUC-ROC metric for the created ensemble. The improvement in the detection quality of the created ensemble was determined by the increase in the metrics ΔAUC-ROC and ΔP@n relative to the best basic algorithm of the group. To assess the impact of the complexity of the ensemble model on computational costs, the coefficient of increase in execution time k_t was proposed, which is defined as the ratio of the ensemble execution time T_a to the execution time of the best basic algorithm T_b, i.e. $k_t=T_a/T_b$. The value $k_t > 1$ indicates an increase in execution time, while $k_t < 1$ indicates its decrease. The results of the study are presented in Tables 5 and 6. If the metric increase in the ensemble results is negative, such a cell of the table is highlighted in gray. The best ensemble is marked in bold.

Table 5
Results of constructing ensembles based on the LSCP algorithm

Input data			Characteristics of the created ensemble			Comparison with the best baseline algorithm of the group		
DPM group	Basic group algorithms (AUC-ROC)	Selected basic algorithms	AUC-ROC	P@n	Time, ms	ΔAUC-ROC	ΔP@n	k_t
1	FastABOD (0,851) kNN (0,833) CBLOF (0,825) KDE (0,815) OCSVM (0,802)	kNN CBLOF	0.855	0.470	1 648	0.004	0.009	17
2	kNN⁺ (0,890) CBLOF⁺ (0,889)	kNN⁺ CBLOF⁺	0.886	0.650	1 469	- 0.004	0.018	158
3	QMCD (0,882) KDE (0,849)	QMCD kNN	0.907	0.566	2 336	0.025	0.066	188

	kNN (0,841) OCSVM$^+$ (0,841) FastABOD$^+$ (0,841)							
4	kNN$^+$ (0,890) CBLOF$^+$ (0,889) QMCD (0,882)	CBLOF$^+$ QMCD	0.912	0.620	1 616	0.022	- 0.012	182

Table 6
Results of ensemble construction based on the SUOD algorithm

Input data		Characteristics of the created ensemble				Comparison with the best baseline algorithm of the group		
DPM group	Basic group algorithms (AUC-ROC)	Selected basic algorithms	AUC-ROC	P@n	Time, ms	ΔAUC-ROC	ΔP@n	k_t
1	FastABOD (0,851) kNN (0,833) CBLOF (0,825) KDE (0,815) OCSVM (0,802)	FastABOD kNN CBLOF	0.840	0.458	2 516	- 0.011	- 0.003	26
2	kNN$^+$ (0,890) CBLOF$^+$ (0,889)	kNN$^+$ CBLOF$^+$	0.888	0.650	2 478	- 0.002	0.018	266
3	QMCD (0,882) KDE (0,849) kNN (0,841) OCSVM$^+$ (0,841) FastABOD$^+$ (0,841)	QMCD kNN	0.898	0.564	2 845	0.016	0.064	229
4	kNN$^+$ (0,890) CBLOF$^+$ (0,889) QMCD (0,882)	CBLOF$^+$ QMCD	0.912	0.621	2 237	0.022	-0.011	251

The results presented in Tables 5 and 6 allow us to state the following:

1. Combining the base algorithms that provide the best individual anomaly detection accuracy does not always allow creating an ensemble that outperforms the best base algorithm of the group. In our study, three out of eight algorithm groups had lower ensemble accuracy.
2. Higher accuracy of outlier detection is achieved by ensembles with stronger baseline or baseline optimized algorithms, with accuracy increasing when moving from Label Encoder to One-Hot Encoder and from scaled data to unscaled.
3. Including algorithms with low AUC-ROC values in the ensemble, in our case OCSVM (AUC-ROC=0.802), leads to a deterioration in the final ensemble forecast, no matter what combination of other strong learners it is combined with.

4. An important problem with ensembles is the increase in computational complexity and resource intensity, which leads to a significant increase in forecasting time (in some cases, hundreds of times), especially when working with large data sets. This can slow down ensemble-based systems operating in real time.
5. Experiments on optimizing hyperparameters of the ensemble algorithms LSCP and SUOD either did not yield any increase in the AUC-ROC metric at all, or the increase was less than 0.01, and therefore we do not present these data.

3.6. Meta-ensemble

Meta-ensemble is an approach in machine learning in which predictions from several different ensemble models are combined into a new ensemble. In scientific publications, other names for this term are also found: ensemble of ensembles, hierarchical ensemble. In essence, these are the second and higher levels of ensemble. This technique is aimed at increasing the reliability and stability of models, reducing the variability of results, but significantly complicates the system. Each ensemble model has its own strengths and weaknesses. Combining their predictions can help compensate for individual shortcomings and achieve better overall performance, especially on complex tasks. This reduces the risk that a single ensemble model is overfitting or suboptimal, and improves the model's ability to make accurate predictions on new, unseen data. Meta-ensemble allows you to "squeeze" even more accuracy out of the data set and can be used for critical tasks where the error of the base ensembles is still unacceptable and you need to get high prediction reliability.

3.7. Meta-ensemble with bypass connections

In modern anomaly detection systems, the effectiveness of a model is often determined by its ability to combine different classification strategies and maintain stability on complex and heterogeneous data. We propose to implement a multi-level hierarchical ensemble structure with bandwidth connections. This will preserve the useful information signal from lower levels and increase the power of the meta-level.

The architecture of the proposed 3-level model is as follows:

Level 1 (basic detectors) are basic anomaly detection algorithms or their hyperparameterized versions. Basic detectors can include:

- individual algorithms – CBLOF, kNN, QMCD, etc.;
- ensemble algorithms, for which it is not necessary to specify a base learner – INNE, LODA, IForest;
- ensemble algorithms or techniques for which one base learner is specified, for example Feature Bagging over one of the individual algorithms.

Level 2 (aggregators / meta-algorithms / meta-models) are ensemble techniques, algorithms or models that combine predictions from several basic first-level detectors, for example: Voting, Stacking, Blending, Logistic Regression, XGBoost, etc.

Level 3 (meta-ensemble) – functionally performs the same tasks as level 2, but its input can receive both forecast data from level 2 and directly from level 1, avoiding the loss of critical information during aggregation (meta-ensemble with bypass connections).

In our study, the LSCP and SUOD ensemble algorithms of the PyOD library are used to aggregate results at levels 2 and 3.

3.8. Construction of meta-ensemble based on LSCP and SUOD algorithms

To construct meta-ensemble, we will supplement our study with the results of the analysis of the performance of the optimized ensemble algorithms INNE and LODA, which showed the best accuracy in the second (Label Encoder, without data scaling) and fourth (One-Hot Encoder, without data scaling) groups of data preprocessing methods (Table 7). In the future, we will

conduct our study on such data. The results of constructing meta-ensemble based on the algorithms LSCP and SUOD, which showed the highest accuracy when combining the basic components, are presented in Tables 8 and 9.

Table 7
Results of optimized ensemble algorithms INNE and LODA

DPM group	Optimized ensemble algorithm	AUC-ROC	P@n	Time, ms
2	INNE$^+$	0.883	0.615	142
2	LODA$^+$	0.907	0.564	101
4	INNE$^+$	0.883	0.615	155
4	LODA$^+$	0.885	0.540	107

Table 8
Results of constructing meta-ensemble based on the LSCP algorithm

Input data		Characteristics of the created meta-ensemble				Comparison with the best baseline algorithm of the group		
DPM group	Basic ensembles or algorithms (AUC-ROC)	Selected basic algorithms	AUC-ROC	P@n	Time, ms	ΔAUC-ROC	ΔP@n	k_t
2	INNE$^+$ (0.883) LODA$^+$ (0.907) kNN$^+$ (0.890) CBLOF$^+$ (0.889)	LODA$^+$, CBLOF$^+$	0.926	0.671	1 507	0.019	0.107	15
4	INNE$^+$ (0.883) LODA$^+$ (0.885) LSCP [CBLOF$^+$, QMCD] (0.912)	LODA$^+$, LSCP [...]	0.929	0.648	4 768	0.017	0.028	2.9
	SUOD [CBLOF$^+$, QMCD] (0.912) kNN$^+$ (0.890) CBLOF$^+$ (0.889) QMCD (0.882)	LODA$^+$, SUOD [...]	0.928	0.649	5 889	0.015	0.030	3.6

Table 9
Results of constructing meta-ensemble based on the SUOD algorithm

Input data			Characteristics of the created meta-ensemble			Comparison with the best baseline algorithm of the group		
DPM group	Basic ensembles or algorithms (AUC-ROC)	Selected basic algorithms	AUC-ROC	P@n	Time, ms	ΔAUC-ROC	ΔP@n	k_t
2	INNE$^+$ (0.883) LODA$^+$ (0.907) kNN$^+$ (0.890) CBLOF$^+$ (0.889)	LODA$^+$, CBLOF$^+$	0.927	0.671	2 323	0.020	0.107	23
4	INNE$^+$ (0.883) LODA$^+$ (0.885) LSCP [CBLOF$^+$, QMCD] (0.912) SUOD [CBLOF$^+$, QMCD] (0.912) kNN$^+$ (0.890) CBLOF$^+$ (0.889) QMCD (0.882)	LODA$^+$, LSCP [...], KNN$^+$, CBLOF$^+$	0.930	0.673	5 121	0.018	0.052	2.3
		LODA$^+$, SUOD [...], KNN$^+$, CBLOF$^+$	0.932	0.679	5 832	0.020	0.058	2.6
		LODA$^+$, LSCP [...], CBLOF$^+$, QMCD	0.930	0.675	3 992	0.018	0.054	1.8
		LODA$^+$, SUOD [...], CBLOF$^+$, QMCD	0.930	0.675	4 781	0.018	0.054	2.1
		LODA$^+$, LSCP [...], KNN$^+$, CBLOF$^+$, QMCD	0.929	0.678	5 037	0.017	0.057	2.3
		LODA$^+$, SUOD [...], KNN$^+$, CBLOF$^+$, QMCD	0.929	0.679	5 813	0.017	0.058	2.6

3.9. Description of the structure of meta-ensemble and their metrics

The full record of the meta-ensemble formulas presented in Tables 8 and 9 is as follows:

- meta-ensemble № 1 – LSCP [LODA$^+$, CBLOF$^+$]
- meta-ensemble № 2 – SUOD [LODA$^+$, CBLOF$^+$]
- meta-ensemble № 3 – LSCP [LODA$^+$, LSCP [CBLOF$^+$, QMCD]]
- meta-ensemble № 4 – LSCP [LODA$^+$, SUOD [CBLOF$^+$, QMCD]]
- meta-ensemble № 5 – SUOD [LODA$^+$, LSCP [CBLOF$^+$, QMCD], KNN$^+$, CBLOF$^+$]
- meta-ensemble № 6 – SUOD [LODA$^+$, SUOD [CBLOF$^+$, QMCD], KNN$^+$, CBLOF$^+$]
- meta-ensemble № 7 – SUOD [LODA$^+$, LSCP [CBLOF$^+$, QMCD], CBLOF$^+$, QMCD]
- meta-ensemble № 8 – SUOD [LODA$^+$, SUOD [CBLOF$^+$, QMCD], CBLOF$^+$, QMCD]
- meta-ensemble № 9 – SUOD [LODA$^+$, LSCP [CBLOF$^+$, QMCD], KNN$^+$, CBLOF$^+$, QMCD]
- meta-ensemble № 10 – SUOD [LODA$^+$, SUOD [CBLOF$^+$, QMCD], KNN$^+$, CBLOF$^+$, QMCD]

Meta-ensemble No. 1–4 are conventional, as they are built only on the basis of ensemble algorithms, while meta-ensemble No. 5–10 additionally have pass-through (bypass) connections between levels.

The recording formula will be explained using the example of meta-ensemble No. 6: the final forecast is carried out by the SUOD ensemble technique, the input of which is the results of the optimized ensemble algorithm LODA$^+$, the SUOD data of the ensemble of the algorithms CBLOF$^+$ and QMCD, as well as the results of the optimized algorithms KNN$^+$, CBLOF+ directly.

Figure 2 shows the structure of meta-ensemble No. 1, which is the fastest of the studied ones, and Figure 3 shows meta-ensemble No. 6, which is the most accurate. These figures show the indication of metrics at intermediate stages of the meta-ensemble operation. Figures 4 and 5 show the generalized AUC-ROC and P@n range diagrams of the studied meta-ensembles, respectively.

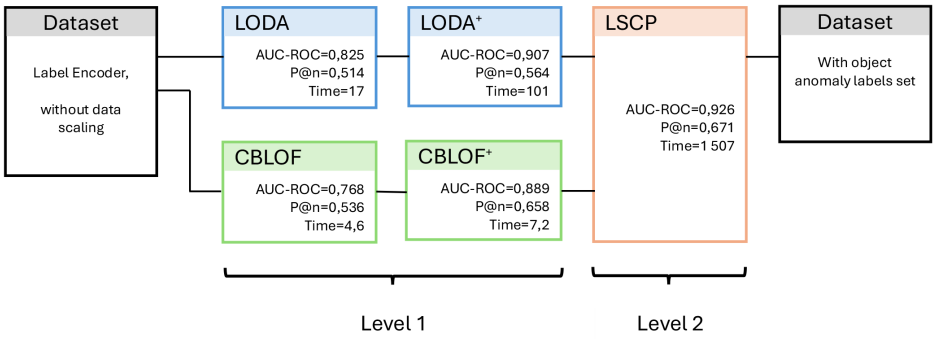

Figure 2: Meta-ensemble LSCP [LODA+, CBLOF+].

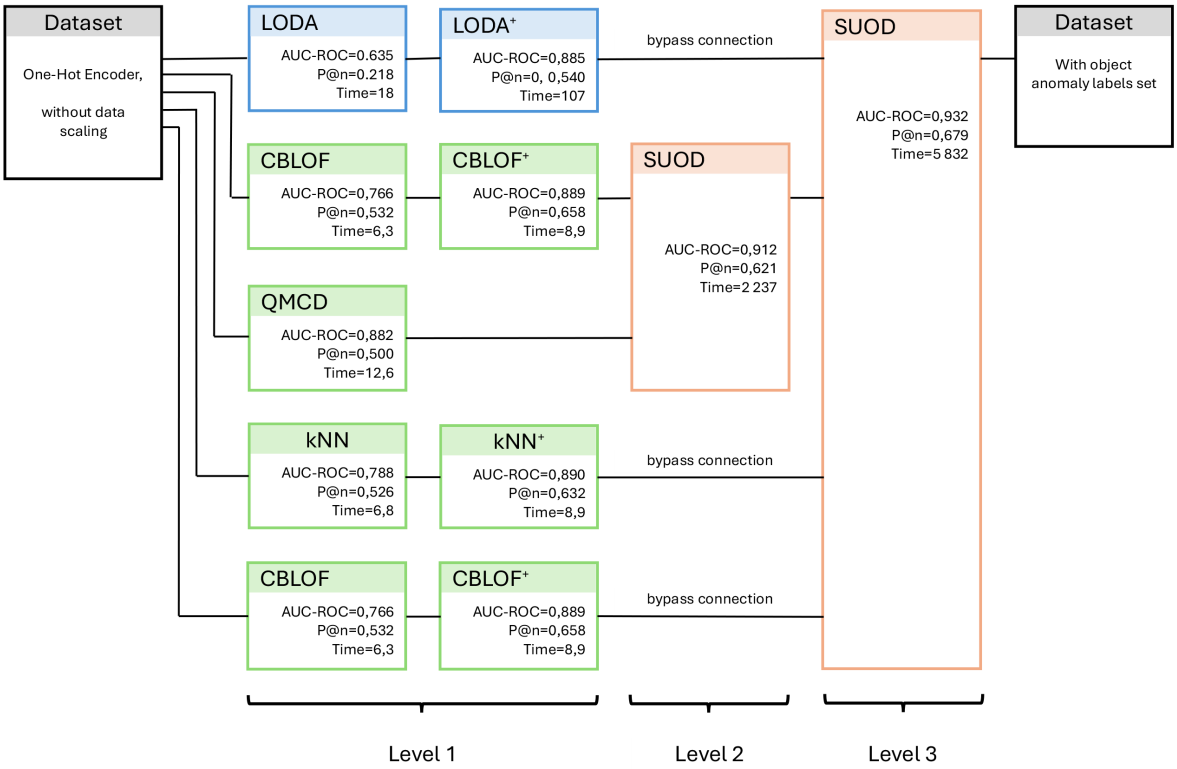

Figure 3: Meta-ensemble with bypass connections SUOD [LODA+, SUOD [CBLOF+, QMCD], KNN+, CBLOF+].

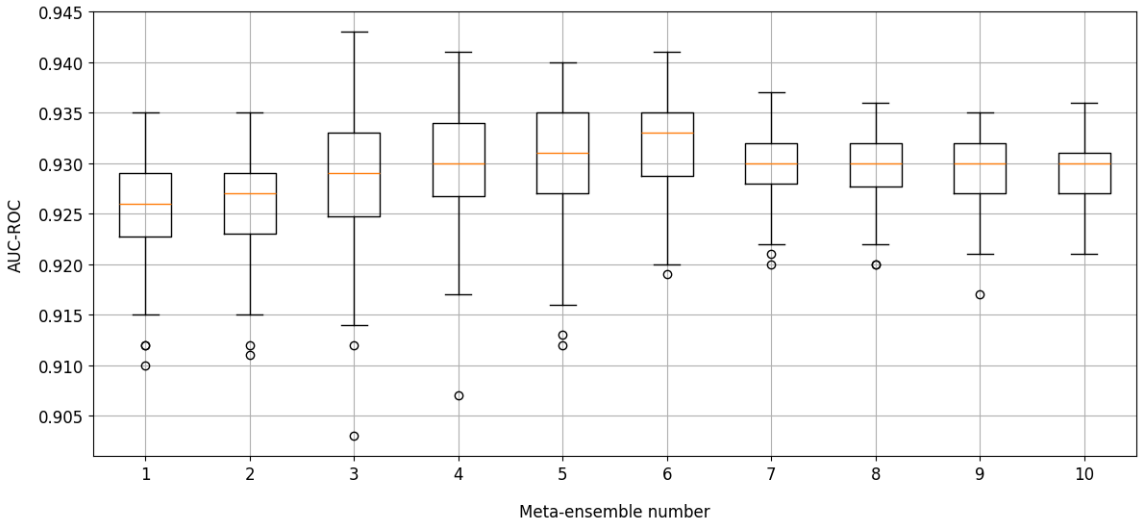

Figure 4: Boxplot of AUC-ROC for the studied meta-algorithms.

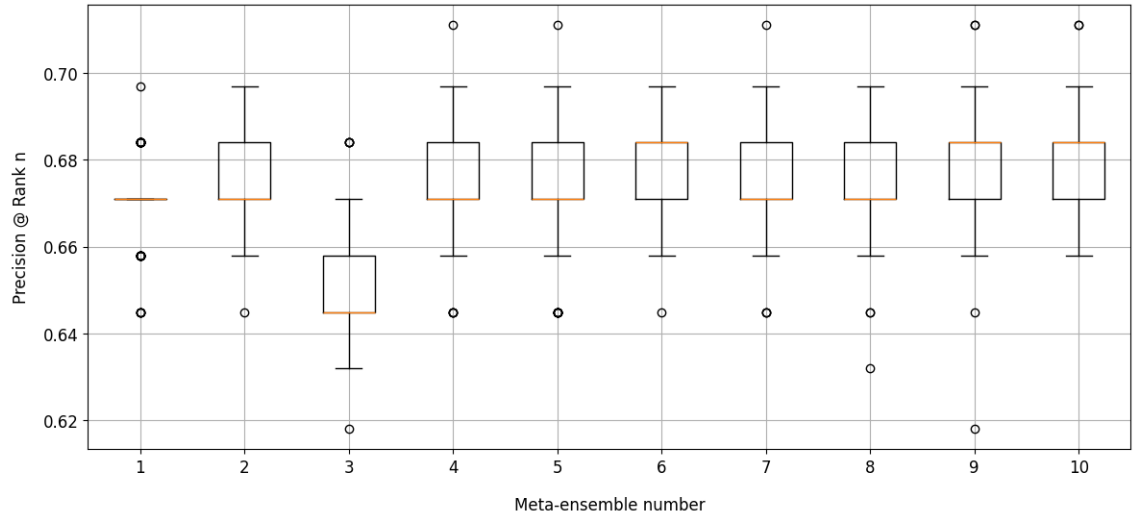

Figure 5: Boxplot of Precision @ Rank n for the studied meta-algorithms.

4. Conclusions

The paper describes and investigates an approach to anomaly detection based on the concept of ensemble ensembles. The proposed system is multi-level and combines individual algorithms (CBLOF, kNN, QMCD, etc.), ensemble algorithms that do not require a base learner (INNE, LODA), and ensemble techniques that combine predictions from several base detectors (LSCP, SUOD) from the PyOD library of the Python programming language. The results of experiments on a real dataset have shown that meta-ensemble provides higher quality anomaly detection by AUC-ROC and P@n metrics compared to individual models and classical ensembles, reduces the dependence of results on a specific algorithm, demonstrates noise resistance and high stability of results.

The maximum accuracy indicators of AUC-ROC = 0.932 and P@n = 0.679 were provided by the meta-ensemble with bandwidth connections based on the SUOD ensemble technique by hierarchically combining the predictions of the optimized ensemble algorithm LODA+, SUOD data of the ensemble of the CBLOF+ and QMCD algorithms, as well as by directly bypassing the results of the optimized algorithms KNN+, CBLOF+. Higher detection accuracy is achieved on unscaled data sets using the One-Hot Encoder categorical feature encoder, which indicates the importance of the correct choice of the data preprocessing method.

The study showed that the use of multi-level ensemble structures significantly increases computational costs, so their choice should be justified in the context of balancing the calculation time and the required accuracy. The correct choice of meta-ensemble allows, with a slight decrease in accuracy (less than 1%), to reduce the detection time by 2-3 times.

The developed approach can be effectively applied in information systems with high error costs. However, it should be taken into account that the need for additional settings, which includes the selection of basic algorithms, the selection of the ensemble type, the search for optimal hyperparameters and the aggregation method to obtain the final forecast, can complicate the process of developing such information systems.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Kyriakides, G. (2019). Hands-On Ensemble Learning with Python. Packt Publishing Ltd.

[2] James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). An Introduction to Statistical Learning with Applications in Python. Springer. https://doi.org/10.1007/978-3-031-38747-0

[3] Kunapuli, G. (2023). Ensemble Methods for Machine Learning. Manning.

[4] Breiman, L. (1996). Bagging predictors. Machine Learning, 24, 123–140. https://doi.org/10.1007/BF00058655.

[5] Зубик, Л., & Зубик, Я. (2023). Ефективність ансамблевих методів машинного навчання для обробки великих даних. Вісник НУВГП. Технічні науки, 2(102), 449–462. https://doi.org/10.31713/vt2202337.

[6] Džeroski, S., Panov, P., & Ženko, B. (2009). Ensemble Methods in Machine Learning. In Encyclopedia of Complexity and Systems Science (pp. 5317–5325). Springer. https://doi.org/10.1007/978-0-387-30440-3

[7] Izenman, A. (2013). Modern Multivariate Statistical Techniques. Regression, Classification, and Manifold Learning. Springer.

[8] Wolpert, D. (1992). Stacked Generalization. Neural Networks, 5(2), 241–259. https://doi.org/10.1016/S0893-6080(05)80023-1.

[9] Zhao, Y., Wang, X., Cheng, C., & Ding, X. (2020). Combining machine learning models using combo library. Proceedings of the AAAI Conference on Artificial Intelligence, 34(9), 13648–13649. https://doi.org/10.1609/aaai.v34i09.7111

[10] Shah, M., Gandhi, K., Patel, K., Kantawala, H., Patel, R., & Kothari, A. (2023). Theoretical Evaluation of Ensemble Machine Learning Techniques. In Proceedings of the 5th International Conference on Smart Systems and Inventive Technology (ICSSIT 2023). https://doi.org/10.1109/ICSSIT55814.2023.10061139.

[11] Reunanen, N., Räty, T., & Lintonen, T. (2020). Automatic optimization of outlier detection ensembles using a limited number of outlier examples. International Journal of Data Science and Analytics, 10, 377–394. https://doi.org/10.1007/s41060-020-00222-4.

[12] Ha, S., Yoon, Y., & Lee, J. (2023). Meta-ensemble learning with a multi-headed model for few-shot problems. ICT Express, 9(5), 909–914. https://doi.org/10.1016/j.icte.2022.09.001.

[13] Zhang, Z., Zhou, L., Wu, Y., & Wang, N. (2024). The meta-learning method for the ensemble model based on situational meta-task. Front. Neurorobot, 18. https://doi.org/10.3389/fnbot.2024.1391247

[14] Amasyali, M. (2019). Improved Space Forest: A Meta Ensemble Method. IEEE Transactions on Cybernetics, 49(3), 816–826. https://doi.org/10.1109/TCYB.2017.2787718.

[15] Silva, R., Canuto, A., Barreto, C., & Xavier-Junior, J. (2021). Automatic recommendation method for classifier ensemble structure using meta-learning. IEEE Access, 9, 106254–106268. https://doi.org/10.1109/ACCESS.2021.3099689.
[16] Duari, G., & Kumar, R. (2023). Hierarchical learning of outliers. In Advances in Cognitive Science and Communications. ICCCE 2023. Cognitive Science and Technology (pp. 869–875). Springer. https://doi.org/10.1007/978-981-19-8086-2_83.
[17] Pastukh, O., & Khomyshyn, V. (2025). Efficiency research of cluster analysis methods for detecting outliers in real estate market. Herald of Khmelnytskyi National University. Technical sciences, 3(1), 362–381. https://doi.org/10.31891/2307-5732-2025-351-45.
[18] Amorim, L., Cavalcanti, G., & Cruz, R. (2023). The choice of scaling technique matters for classification performance. Applied Soft Computing, 133, 109924. https://doi.org/10.1016/j.asoc.2022.109924.
[19] Optuna - A hyperparameter optimization framework. Optuna. https://optuna.org.
[20] PyOD 2.0.5 documentation. https://pyod.readthedocs.io/en/latest/.
[21] Khomyshyn, V., Pastukh, O., Yatsyshyn, V., Zagorodna, N., & Baran, I. (2025). Comparative analysis of ensemble methods for outlier detection in real estate. In Proceedings of the 3rd International Workshop on Computer Information Technologies in Industry 4.0 (CITI 2025), June 11–12, 2025, Ternopil, Ukraine, pp. 79-91. https://ceur-ws.org/Vol-4057/paper5.pdf.
[22] Zhao, Y., Nasrullah, Z., Hryniewicki, M., & Li, Z. (2019). LSCP: Locally Selective Combination in Parallel Outlier Ensembles. In Proceedings of the 2019 SIAM International Conference on Data Mining (pp. 585–593). https://doi.org/10.48550/arXiv.1812.01528.
[23] Zhao, Y., Hu, X., Cheng, C., Wang, C., Wan, C., Wang, W., Yang, J., Bai, H., Li, Z., Xiao, C., Wang, Y., Qiao, Z., Sun, J., & Akoglu, L. (2021). SUOD: Accelerating Large-scale Unsupervised Heterogeneous Outlier Detection. In Proceedings of the 4th Conference on Machine Learning and Systems (MLSys). https://doi.org/10.48550/arXiv.2003.05731.