On the efficient Explanation of Outlier Detection Ensembles through Shapley Values[⋆]

Simon Klüttermann[0000−0001−9698−4339], Chiara Balestra[0000−0002−3620−9012], and Emmanuel Müller[0000−0002−5409−6875]

TU Dortmund University, Dortmund, Germany
{simon.kluettermann,chiara.balestra,emmanuel.mueller}@cs.tu-dortmund.de

Abstract. Feature bagging models have revealed their practical usability in various contexts, among them in outlier detection, where they build ensembles to reliably assign outlier scores to data samples. However, the interpretability of so-obtained outlier detection methods is far from achieved. Among the standard black-box models interpretability approaches, we find Shapley values that clarify the roles of single inputs. However, Shapley values are characterized by high computational runtimes that make them useful in pretty low-dimensional applications. We propose *bagged Shapley values*, a method to achieve interpretability of feature bagging ensembles, especially for outlier detection. The method not only assigns local importance scores to each feature of the initial space, helping to increase the interpretability but also solves the computational issue; specifically, the *bagged Shapley values* can be exactly computed in polynomial time.

Keywords: Explainable Machine Learning · Polynomial Shapley values · Outlier Detection

1 Introduction

Detecting anomalous samples is a crucial task in various domains, ranging from fraud detection in financial systems [12] to identifying defective components in manufacturing processes [10]. Outlier detection can generally be categorized into two paradigms: *supervised* and *unsupervised*. Supervised methods rely on labeled data, explicitly defining and identifying anomalies during model training [11]. In contrast, unsupervised techniques operate without labeled anomalies, making them particularly valuable when labeled data is scarce or costly. Unsupervised outlier detection encompasses a myriad of approaches, each with unique strengths and limitations. One notable trend for unlabeled data involves ensemble methods; Ensemble techniques [31] leverage the diversity outputs of

[⋆] This work was supported by the Lamarr-Institute for ML and AI, the research training group *Dataninja*, the Research Center Trustworthy Data Science and Security, the Federal Ministry of Education and Research of Germany and the German federal state of NRW. The Linux HPC cluster at TU Dortmund University, a project of the German Research Foundation, provided the computing power.

multiple potentially different base models to produce more robust and accurate predictions, thus directly enhancing the performance and reliability of outlier detection algorithms [1]. The nature of the base models divides the ensembles among heterogeneous or homogeneous [14]; Some algorithms, e.g., DEAN [5] and IForest [16], use homogeneity to profit from a higher number of submodels.

The importance of outlier detection underscores the need for accurate and interpretable methods. Shapley values [25] have emerged as a promising technique for interpreting the contributions of individual features in black-box models. They offer mathematical guarantees of fairness that make them an attractive choice for outlier detection as well. However, their practical application poses a significant challenge due to the requirement of training an exponentially large number of models. While significant progress has been made in anomaly detection interpretability [15], challenges persist. The trade-off between interpretability and model complexity transferred to the computational complexity of the feature importance scores, thus remains an interesting topic of investigation.

In response to this challenge, we propose an innovative approach that leverages modern ensemble methods to approximate Shapley values efficiently and makes outlier detection methods based on feature bagging interpretable. First, we delve into the details, defining the *bagged Shapley values* and presenting a theoretical proof of our approach. The experimental results demonstrate our method's effectiveness in achieving efficient interpretability in outlier detection tasks with complex, high-dimensional data. The code is available on Github[1].

2 Related Work

Ensemble methods for outlier detection Ensemble methods emerged as a powerful paradigm for improving outlier detection algorithms w.r.t. reliability and performance [14]. Ensemble methods comprehend bagging [3], boosting [24], and stacking [23]. Bagging involves training multiple base models (e.g., k-nearest neighbors, Support Vector Machines, neural networks, among others) on possibly bootstrapped data samples and aggregating their predictions. Adapted to outlier detection, the ensemble's collective decision provides more robust results [1]. Homogeneity among the base models' types characterizes *homogeneous* outlier ensembles: Submodels usually differ only by a different initialization. DEAN [5] and Isolation Forest (IForest) [16] are prime examples of outlier detection methods employing such homogeneous ensembles; DEAN is based on multiple neural networks, while IForest relies on a collection of isolation trees.

Shapley Values, for intepretability and beyond Shapley values [25] originate from Cooperative Game Theory. Since their first applications, they gained prominence as a powerful tool for increasing the interpretability of machine learning black-box models [18,26,21]. Shapley values offer a theoretically sound framework for quantifying the impact of each feature or factor in a model's prediction; the scores, being the average marginal contribution across all possi-

[1] https://github.com/KDD-OpenSource/ensemble_shapley

ble feature combinations, are robust and interpretable. Attributing the contributions of the individual features revealed helpful for outlier detection [28,29], where Shapley values provide valuable insights into the importance of features in identifying anomalies. However, their practical use is contrasted by one significant challenge, namely their computational complexity. The exact computation of Shapley values requires evaluating a *value function* for every possible subset of players. Thus, the consequent exponential blow-up in computational cost soon renders their use for high-dimensional contexts infeasible. Approximation techniques have been implemented to make Shapley values more accessible; These include Monte Carlo sampling, stratification of players, and kernel approximations [6,7,18,4,27]. Each method addresses the efficient computation of Shapley values differently, with potential accuracy and computational cost trade-offs.

Interpretability for anomaly detection methods Interpreting anomaly detection methods is essential for understanding *why* single data points are considered anomalous, e.g., in safety-critical applications. Feature importance analysis plays an essential role [15]: Techniques such as feature attribution [9] are employed to highlight which features have the most significant impact on the detection. Additionally, we find rule-based models [19], decision trees [20], and model-agnostic techniques like LIME [21] and SHAP [18] to shed light on the decision-making process of anomaly detection models. Furthermore, visualizations are essential for enhancing trust in complex scenarios. Examples are heatmaps, scatter plots, and time-series representations [13,2].

3 Outlier detection ensembles

In our context, a set $X \subseteq \mathbb{R}^N$ of data points can be parted into two subsets: the set of *normal observation* indicated with X_{nor}, and the set of abnormal observations, indicated with X_{abn}. In unlabeled data, distinguishing normal from anomalous data is not always straightforward. We consider a model for outlier detection f, that aims at classifying each data point $x \in X$ as either *normal* or *anomalous*. Among the various anomaly detection methods, we focus on methods that provide to each data point a score measuring its *outlierness*.

Definition 1. *Given a set of data points X, we call* model *a function $a : X \mapsto \mathbb{R}$ where $a(x)$ represents the outlier score assigned by a to the sample x.*

The higher the value $a(x)$, the more likely x is considered to be an anomaly compared to the set X. On the same set X, various outlier detection models can be constructed. We indicate with $\mathcal{M}_X$ the set of models constructed on X.

Definition 2 (Ensemble). *Given a set of (sub)models $\mathcal{M}_X$, an ensemble is a function $A_{\mathcal{M}_X} : X \mapsto \mathbb{R}$ that assigns to each $x \in X$ its average outlier score, i.e.,*

$$A_{\mathcal{M}_X}(x) = \frac{1}{\|\mathcal{M}_X\|} \sum_{a \in \mathcal{M}_X} a(x). \tag{1}$$

The ensemble prediction is the average submodel prediction in the set $\mathcal{M}_X$.

Using the trick of projected data points in lower dimensional spaces, we reach the definition of bagging. We indicate with $\mathcal{N}$ the set of coordinates of X and with X_I the set of data points in X projected only on the $I \subseteq \mathcal{N}$ coordinates (or *features*), i.e., given $x \in X$ the corresponding point $x_I = (x_i)_{i \in I}$ and $I \subset \mathcal{N}$. Now we can define a subset $\mathcal{M}_{X_I} \subseteq \mathcal{M}_X$ as the set of submodels that belongs to $\mathcal{M}_X$ trained only on X_I.

The bagging procedure is meant to randomly cover the information in X, considering only the projection of X in smaller-sized subsets. We refer to the size of the data points in the projection as *bag*. Having $X \subseteq \mathbb{R}^N$ fixed and $\text{bag} \leq N$, we can get $\binom{N}{\text{bag}}$ different subsets of size bag from the N features.

Definition 3 (Bagging). *After fixing the* bag*, the* bagging procedure *consists in defining the model $b_{S,a} \in \mathcal{M}_S$ such that $b_{S,a}(x)$ is the result of a model a when trained on the data set X_S and S is a subset of N whose size is $|S| = bag$.*

The bagging procedure does not fix either the model a from $\mathcal{M}_X$ or the set $S \subseteq \mathcal{N}$, thus potentially covering, using sufficiently many random seeds, all the information contained in X. We write $b_{S,a|\text{seed}}$ for the specific *bagging submodel* resulting after we fixed the seed for the random sampling of S and the model a. Finally, we can construct the so-called *feature bagging* ensemble based on the bagging technique.

Definition 4 (Feature Bagging). *Given a dataset X and a set of models $\mathcal{M}_X$, we define the function $f_{\mathcal{M}_X} : X \mapsto \mathbb{R}$ such that it assign to each $x \in X$ the score defined as*

$$f_{\mathcal{M}_X}(x) = \lim_{n \to \infty} \frac{1}{n} \sum_{j=0}^{n} b_{S,a|seed[j]}(x). \tag{2}$$

where seed is an eventually infinite vector of randomly drawn seeds.

A similar definition could also be made for non-outlier detection ensembles as long as the output is a linear combination of the submodel predictions. Still, feature bagging is most commonly used in outlier detection.

4 The *bagged Shapley values*

A cooperative game is a pair $(\mathcal{N}, v)$ where $\mathcal{N}$ represents the set of *players*, and v is a function over the subsets of $\mathcal{N}$. v assigns to each *coalition* of players a worth, i.e., a positive real number representing the score obtained by the players as a team. Usually, the monotonicity of the value function is assumed, i.e., $v(\mathcal{A}) \leq v(\mathcal{B})$ if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{N}$.

The Shapley values are a *fair* assignment of weights to the single players that consider the role of the single players in any single coalition. Given a game $(\mathcal{N}, v)$, $\phi_v(i)$ represents the Shapley value of player i:

Definition 5 (Shapley Value). *Given a game $(\mathcal{N}, v)$, the Shapley value $\phi_v(i)$ of player i is defined as*

$$\phi_v(i) = \sum_{\mathcal{S} \subseteq \mathcal{N}, i \notin \mathcal{S}} \frac{|\mathcal{S}|! \cdot (|\mathcal{N}| - |\mathcal{S}| - 1)!)}{|\mathcal{N}|!} \left[v(\mathcal{S} \cup \{i\}) - v(\mathcal{S}) \right] \tag{3}$$

We refer to $v(\mathcal{S} \cup \{i\}) - v(\mathcal{S})$ as the *marginal contribution of i to $\mathcal{S}$*. Shapley values have a flexible and straightforward definition, depending only on v and the number of players; this made them the object of study in various circumstances and applications. However, their computation results in an NP-hard problem that approximation approaches can only partly solve. We show that the exact computation of Shapley value-similar scores for feature bagging ensembles can be easily reduced to a polynomial time.

We introduce the *bagged Shapley values*; their definition perfectly aligns with the impossibility of training an ensemble method with less than *bag* features. We rewrite the definition of Shapley values from Equation (3) $\phi_{f_{\mathcal{M}_X}(x)}(i)$ for feature bagging ensembles, where $x \in X \subseteq \mathbb{R}^N$ is a data point, $f_{\mathcal{M}_X}$ is the feature bagging model and we are interested in assigning to the coordinate i of X an importance score in predicting the overall outlier score $f_{\mathcal{M}_X}(x)$. We define the bagged Shapley values:

Definition 6 (bagged Shapley Value). *Given a set of data points $X \subseteq \mathbb{R}^N$, a set of (sub)models $\mathcal{M}_X$ and a feature bagging model $f_{\mathcal{M}_X}$ defined over $\mathcal{M}_X$, the* bagged Shapley values *are the values*

$$\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) = \sum_{S \subseteq \mathcal{N}, i \notin S, s \geq bag} \frac{N}{N - bag} \frac{s! \cdot (N - s - 1)!)}{N!} \left[f_{M_{X_{S \cup \{i\}}}}(x) - f_{M_{X_S}}(x) \right] \tag{4}$$

This equation removes terms with magnitude $\propto \frac{\text{bag}}{N}$, a necessary step, as defining an ensemble model with less than *bag* features is not possible. Notice that the higher the dimension of the data points in X is, the smaller the difference between $\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i)$ and $\phi_{f_{\mathcal{M}_X}(x)}(i)$. To somewhat correct for this difference, we add a factor $\frac{N}{N - \text{bag}}$ to compensate that we are summing over fewer subsets of $\mathcal{N}$.

5 Theoretical guarantees for the approximation

The main result of our studies regards the chance to express Shapley values with a limited number of selected bagging submodels, thus avoiding the exponential computational costs of Shapley values.

Theorem 1. *The* bagged Shapley values *can be expressed using a selection of submodels involved in the feature bagging ensemble $f_{\mathcal{M}_X}$. In particular, it holds*

$$\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) \propto f_{\mathcal{M}_X}(x) - f_{\mathcal{M}_{X_{\mathcal{N} \setminus i}}}(x).$$

Proof. To increase readability, we use the notation

$$k(S,N) = \frac{N}{N-\text{bag}} \frac{s!(N-s-1)!}{N!}$$

where $s = |S|$ and $N = |\mathcal{N}|$. For abuse of notation and readability, we write S instead of X_S throughout the whole proof.

Now, we can rewrite the *bagged Shapley values* in the following way $b_{S,a|\text{seed}}$ and substitute it with $b_{|\text{seed}} \in \mathcal{M}_S$

$$\begin{aligned}\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) &= \sum_{S\subseteq\mathcal{N}, i\notin S, s\geq\text{bag}} k(S,N)\left[f_{S\cup\{i\}}(x) - f_S(x)\right] \\ &= \lim_{n\to\infty} \sum_{S\subseteq\mathcal{N}, i\notin S, s\geq\text{bag}} k(S,N) \\ &\cdot \left(\frac{\sum_{j=0,\dots,n, b\in\mathcal{M}_{S\cup\{i\}}} b_{|\text{seed}}(x)}{\|\mathcal{M}_{S\cup\{i\}}\|} - \frac{\sum_{j=0,\dots,n, b\in\mathcal{M}_S} b_{|\text{seed}}(x)}{\|\mathcal{M}_S\|}\right)\end{aligned}$$

where $\mathcal{M}_K = \{a \in \mathcal{M}_X \mid a \text{ restricted to features in } K\}$ is the subset of models that contain only features included in K.

From the previous equation, we see that $\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i)$ is a sum over the same bagging models multiple times, as they are part of various subsets. We can simplify the writing to evaluate each model only once but weight them.

$$\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) = \lim_{n\to\infty} \frac{1}{\|\mathcal{M}_X\|} \sum_{b\in\mathcal{M}_X} \alpha_b \cdot b_{|\text{seed}}(x) - \frac{1}{\|\mathcal{M}_{\mathcal{N}\setminus i}\|} \sum_{b\in\mathcal{M}_{\mathcal{N}\setminus i}} \beta_b \cdot b_{|\text{seed}}(x) \quad (5)$$

Noting that we can shuffle our feature labels without changing Equation 5, $\alpha_b = \alpha$ and $\beta_b = \beta$ have to be independent on the specific model $b_{|\text{seed}}$. By the same argument, α and β can not depend on the model outputs $b_{|\text{seed}}(x)$. This allows us to choose any model $b(x)$ to compute them; we pick here

$$b(x) = \begin{cases} 1 \text{ if model } b \text{ considers feature i} \\ 0 \text{ else} \end{cases} . \quad (6)$$

Using the proposed $b(x)$, the β term disappears, thus we can write α as:

$$\begin{aligned}\alpha &= \lim_{n\to\infty} \frac{\sum_{S\subseteq\mathcal{N}, i\notin S, |S|\geq\text{bag}} k(S,N) \frac{\|\mathcal{M}_X\|}{\|\mathcal{M}_{X_{S\cup\{i\}}}\|} \sum_{b\in\mathcal{M}_{X_{S\cup\{i\}}}} b(x)}{\sum_{b\in\mathcal{M}_X} b(x)} \\ &= \lim_{n\to\infty} \frac{\sum_{S\subseteq\mathcal{N}, i\notin S, |S|\geq\text{bag}} k(S,N) \cdot \frac{\text{count}(\mathcal{M}_{X_{S\cup\{i\}}})}{\|\mathcal{M}_{X_{S\cup\{i\}}}\|}}{\frac{\text{count}(\mathcal{M}_X)}{\|\mathcal{M}_X\|}}\end{aligned}$$

where $\text{count}(\mathcal{M}_{X_K})$ is the number of models in $\mathcal{M}_{X_K}$ that contain one specific feature in K. We can use $\lim_{n\to\infty} \frac{\text{count}(\mathcal{M}_{X_K})}{\|\mathcal{M}_{X_K}\|} = \frac{\binom{|K|-1}{\text{bag}-1}}{\binom{|K|}{\text{bag}}} = \frac{\text{bag}}{|K|}$ thus getting

$$\begin{aligned}\alpha =& N\frac{N}{N-\text{bag}}\sum_{s=\text{bag}}^{N-1}\binom{N}{s}\cdot\frac{s!(N-s-1)!}{N!}\cdot\frac{1}{s+1}\\ =&\frac{N}{N-\text{bag}}\sum_{s=\text{bag}}^{N-1}\cdot\frac{1}{s+1}\\ =&\frac{N}{N-\text{bag}}\cdot\left(\psi^0(N+1)-\psi^0(\text{bag}+1)\right)\end{aligned}$$

with the digamma function ψ^0.

When instead of choosing $b(x)$ to be independent of i, we find that $\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) \propto (\alpha-\beta)$. But since the feature is designed not to have any effect, we know that $\tilde{\phi}_{f_{\mathcal{M}_X}(x)}(i) = 0$ and thus $\alpha = \beta$. This concludes the proof.

□

The results not only show that the bagged Shapley value is proportional to the difference of two feature bagging, respectively defined on $\mathcal{M}_X$ and $\mathcal{M}_{X_{\mathcal{N}\setminus i}}$, but also that when using bagging models, we can estimate the bagged Shapley values in polynomial time. This is because for deterministic submodels, instead of using ∞ of them, we only need to train $\binom{N}{bag} < N^{bag}$ submodels.

6 Experiments

We evaluate our approach on various freely available real-world datasets with varying numbers of features [30,8,17]. We conduct experiments on the correctness of the approximation (Section 6.1), the effectiveness (Section 6.2), and the scalability (Section 6.3) of our approach.

6.1 Quality of the Approximation

To fairly investigate the approximation accuracy of the bagged Shapley values, we use a low-dimensional dataset, i.e., the five-dimensional phoneme dataset[30], thus requiring the training of feature bagging ensemble models only 2^5 times. The low-dimensionality of the dataset allows us to compute the non-approximated version of Shapley values without incurring extremely long runtimes. We train isolation trees from [16] with a bagging size of 2 and simplify the anomaly score from [16] to fit our methodology by using the negative average path length over all trees as an indicator of anomalies. We train one million submodels and average the obtained results to guarantee consistent and robust results. The total training takes about 70min of CPU time[2]. The ROC-AUC score is 0.733.

[2] All experiments were performed on Intel Xeon E5 CPUs. In the paper, we stick to CPUs over GPUs also when we use neural network submodels; the choice is justified by the higher amount of parallelization they allow.

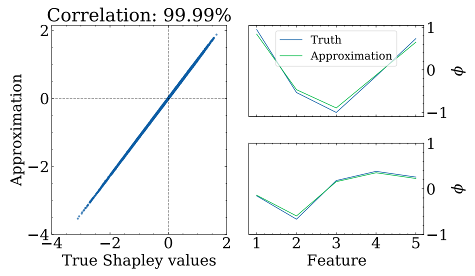

Fig. 1. Left: Plot of the bagged Shapley values against the exact Shapley values for each data sample in the phoneme dataset. Right: Shapley values and their approximation for two example samples. The color-coding of the features is represented in the legend.

We separate the trained models into ensembles for each subset of them and compute the exact Shapley values and the bagged Shapley values. We combine the values obtained into Figure 1. As the mapping lies on the diagonal line, we conclude that the approximation works well on all data points.

6.2 Effectiveness

We can compute the bagged Shapley values for datasets whose dimensions are too high for an exact computation. We focus on the MNIST dataset [8], a collection of images of hand-written digits usually used to train image-recognition models. Following the approach of [22], we consider normal all images representing a handwritten 7, and anomalous the images representing other digits. Each image has a resolution of $28 \cdot 28$, i.e., we handle 784 features in each image. Computing the exact Shapley values for the single pixels requires $2^{784} \approx 10^{237}$ evaluations, a number significantly larger than the computational power available.

For the bagged Shapley values, we use the bagging size bag $= 32$. We train two models: we use DEAN, a deep learning model-based ensemble, and a shallow isolation forest [16]. We choose DEAN [5] because of its inherited feature bagging and relatively low training time per submodel. The training time is significantly longer than using IForest[3]. Note that we do not only train a model on each possible subset, as the number of subsets is still $\binom{768}{32} \approx 4 \cdot 10^{32}$. Instead, we train on random subsets until the result converges. This also helps deal with the random nature of our algorithms.

Figure 2 represents the plots of the Shapley values for five representative samples in the form of heatmaps; bright colors represent high score, i.e., features highly increasing the outlier score. Each heatmap, both for DEAN and IFOR, highlights the changes to the original input that would make it closer to a normal observation by highlighting the erroneous regions. From the left to the right side, the first two input images are labeled as normal; however, they still contain

[3] The isolation forest takes about 220min of CPU time. DEAN requires about 113days; However, the independent ensembles are easy to parallelize, and less accurate results can already be achieved with ten thousand submodels (27hours).

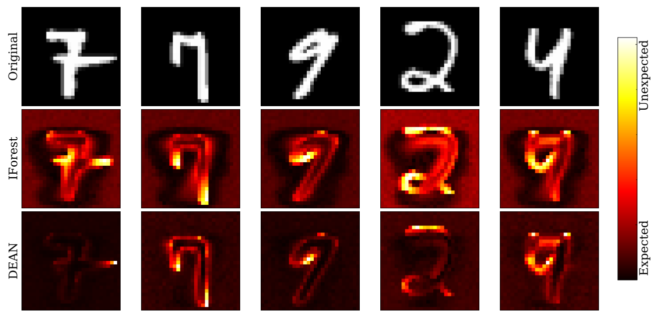

Fig. 2. MNIST dataset. The original images are in the top row. The bottom rows contain the derived *bagged Shapley values* heatmap for ISOR and DEAN. We rescaled the color legend to the upper and lower bound of the Shapley values in each plot.

features that are not expected, e.g., the middle horizontal line in the first image. These unexpected features are highlighted in bright red/yellow. Similarly, the other three images obtain high outlier scores, although they contain typical features for normal input images. These features are also unexpected by the model and thus result in high Shapley values. Examples are represented by the nine and the four; removing the lower line from the circle would make the nine more similar to a normal observation, while adding a horizontal line to the top would make the four more similar to a seven.

Comparing DEAN and IForest, we see how the understanding of the normal concept, i.e., the digit "seven", of the isolation forest is too simple to explain the predictions entirely. In the second column of Figure 2, we see that the isolation forest expects the tail of the seven to bend instead of going straight down. On the other hand, DEAN, based on a deep learning method, has less difficulty in learning a broader concept of seven. This is also reflected in the outlier detection performance: While DEAN reaches a ROC-AUC of 0.9698 on the dataset, the isolation forest only reaches a lower 0.9118 score. We strongly believe that the bagged Shapley images provide useful insights into what the model understood and learned from the training data, additionally to better performance measured by the ROC-AUC metric.

6.3 Scalability

We select the celebA dataset [17] to study how the approach scales to larger datasets. celebA contains images with $218 \cdot 178 = 38804$ pixels, which we convert to grayscale to simplify the plotting later. In the previous section, we showed how complex patterns can overwhelm outlier detection ensembles that struggle to learn a proper schema for normal and abnormal data points. Thus, we aim to maximize the separation between normal and abnormal classes in order to simplify the learning task. We divide the dataset into normal and anomalous instances, where we characterize a normal observation being labeled with the attributes "female", "young", "attractive", and "not bald". The inverse attributes

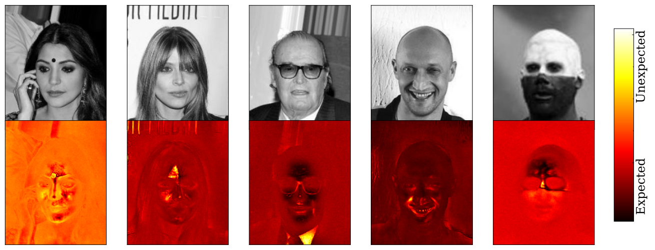

Fig. 3. Analysis on the celebA dataset. Heatmaps show the bagged Shapley value; Brighter colors indicate higher values.

characterize an abnormal observation. Here, the choice of attributes was only guided by the distribution of attributes in the dataset, and similar results would likely have followed any other choices for the anomalous and normal classes. We only trained the DEAN ensemble on the dataset, as the model proved to handle complicated attributed data better. We represent the obtained bagged Shapley values as heatmaps on five different images in Figure 6.3. The first row is the input image, while the second contains the corresponding Shapley values. The images resulting from the bagged Shapley values plotting have high resolution and show some features as more anomalous; However, the designed features do not match the designed separation in normal and abnormal images. This can also be seen in the ROC-AUC score of 0.6184. The most anomalous features seem to be (from left to right) the bindi, the partially covered forehead, the shirt collar, the laugh lines, and the skin paint transition. These are rare features in the images of young women in celebA, thus considered anomalous by the model. Still, the complexity of the separation is likely too big for the available samples (≈ 72000), and thus, the learning, as shown by the ROC-AUC, is inaccurate. Although the features outlined are not the expected ones from our understanding of the separation between the two classes, it is worth noticing how the bagged Shapley value maps can be used to understand and improve the outlier detection models. The runtime of the training procedure for one million DEAN submodels is ≈ 468 days; training 500 submodels at the same time requires about 4 days of CPU time. We use 4 millions of submodels in our training setup, under parallelization assumption, and set up the bagging size to be $bag = 32$. A different bagging size might have achieved more accurate results, but we did not optimize it since, in most contexts, the outlier detection task sets the bagging size.

We finally want to characterize the minimum number of submodels needed for our methodology to perform well. For this, we calculate the bagged Shapley values maps so that each feature is used 10, 100, and 1000 times. The corresponding maps for the center image from Figure 6.3 are shown in Figure 4. While some features are already visible at about 12000 submodels, the noise level being still very high, facial features are undetectable; with about 10, those become visible while extensively the number of submodels to about 100 times more, they have become clear. As a rule of thumb, we suggest training $10 \cdot N$ features to visualize the basic features and to train $10 \cdot N^{\frac{3}{2}}$ for clear images.

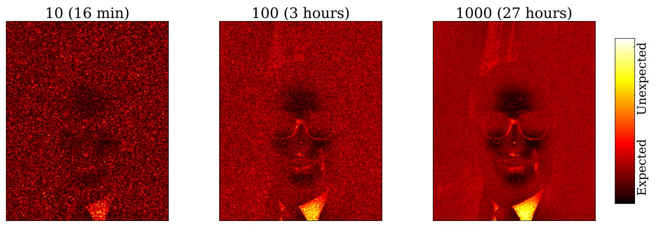

Fig. 4. Shapley value maps for different numbers of submodels. Here we use 12127, 121263 and 1212625 submodels, so that each feature is approximately sampled 10, 100 and 1000 times. The times stated assume a parallelization with 500 CPUs.

7 Conclusions

Detecting and explaining outlier can be highly complicated. Shapley values by their side offer a flexible definition, easily applicable to this context; However, their high computational costs represents an often insormontable downside that makes their exact computation often unfeasible.

We combine Shapley values with ensemble techniques, specifically focusing on feature-bagging ensembles for outlier detection. The *bagged Shapley values* offer an advantageous reduction of the computational costs, giving the chance to compute importance scores for settings with tens of thousands of features. Furthermore, we showed the value of highlighting anomalous features in images to obtain insights into the features learned by the outlier detection method.

We believe that combining Shapley values with ensemble methods can boost the use of Shapley values in the Machine Learning community, showing advantages from a computational and interpretability point of view, as well as lead to better, more reliable, outlier detection models.

References

1. Ali, K.M., Pazzani, M.J.: Error reduction through learning multiple descriptions. Machine learning **24** (1996)
2. Balestra, C., Li, B., Müller, E.: slidshaps – sliding shapley values for correlation-based change detection in time series. In: DSAA (2023)
3. Breiman, L.: Bagging predictors. Machine learning **24** (1996)
4. Burgess, M.A., Chapman, A.C.: Approximating the shapley value using stratified empirical bernstein sampling. In: IJCAI (2021)
5. Böing, B., Klüttermann, S., Müller, E.: Post-robustifying deep anomaly detection ensembles by model selection. In: ICDM (2022)
6. van Campen, T., Hamers, H., Husslage, B., Lindelauf, R.: A new approximation method for the shapley value applied to the wtc 9/11 terrorist attack. Social Network Analysis and Mining **8**, 1–12 (2018)
7. Castro, J., Gómez, D., Tejada, J.: Polynomial calculation of the shapley value based on sampling. Computers & Operations Research **36**(5), 1726–1730 (2009)
8. Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine **29** (2012)

9. Dissanayake, T., Fernando, T., Denman, S., Sridharan, S., Ghaemmaghami, H., Fookes, C.: A robust interpretable deep learning classifier for heart anomaly detection without segmentation. IEEE Journal of Biomedical and Health Informatics **25** (2021)
10. Dong, L., Shulin, L., Zhang, H.: A method of anomaly detection and fault diagnosis with online adaptive learning under small training samples. Pattern Recognition **64** (2017)
11. Han, S., Hu, X., Huang, H., Jiang, M., Zhao, Y.: Adbench: Anomaly detection benchmark. In: NeurIPS (2022)
12. Hilal, W., Gadsden, S.A., Yawney, J.: Financial fraud: a review of anomaly detection techniques and recent advances. Expert systems With applications (2022)
13. Kadir, T., Brady, M.: Saliency, scale and image description. International Journal of Computer Vision **45**(2), 83–105 (2001)
14. Klüttermann, S., Müller, E.: Evaluating and comparing heterogeneous ensemble methods for unsupervised anomaly detection. In: IJCNN (2023)
15. Li, Z., Zhu, Y., Van Leeuwen, M.: A survey on explainable anomaly detection. ACM Transactions on Knowledge Discovery from Data **18** (2023)
16. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: ICDM (2008)
17. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: ICCV (2015)
18. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)
19. Müller, E., Keller, F., Blanc, S., Böhm, K.: Outrules: a framework for outlier descriptions in multiple context spaces. In: ECML PKDD (2012)
20. Park, C.H., Kim, J.: An explainable outlier detection method using region-partition trees. The Journal of Supercomputing **77** (2021)
21. Ribeiro, M.T., Singh, S., Guestrin, C.: " why should i trust you?" explaining the predictions of any classifier. In: KDD (2016)
22. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: ICML (2018)
23. Sandim, M.O.: Using Stacked Generalization for Anomaly Detection. Ph.D. thesis
24. Schapire, R.E., et al.: A brief introduction to boosting. In: IJCAI (1999)
25. Shapley, L.S.: A value for n-person games. Contributions to the Theory of Games (1953)
26. Strumbelj, E., Kononenko, I.: An efficient explanation of individual classifications using game theory. The Journal of Machine Learning Research **11**, 1–18 (2010)
27. Štrumbelj, E., Kononenko, I.: Explaining prediction models and individual predictions with feature contributions. Knowledge and information systems **41**(3), 647–665 (2014)
28. Takahashi, T., Ishiyama, R.: FIBAR: Fingerprint Imaging by Binary Angular Reflection for Individual Identification of Metal Parts. In: EST (2014)
29. Tallón-Ballesteros, A., Chen, C.: Explainable ai: Using shapley value to explain complex anomaly detection ml-based systems. Machine learning and artificial intelligence **332**, 152 (2020)
30. Triguero, I., et al.: Keel 3.0: An open source software for multi-stage analysis in data mining. International Journal of Computational Intelligence Systems **10** (2017)
31. Zimek, A., Campello, R.J., Sander, J.: Ensembles for unsupervised outlier detection: challenges and research questions a position paper. SIGKDD Explorations Newsletter **15** (2014)